Fermi National Accelerator Laboratory

FERMILAB-TM-1747

SSD Module Specifications and As-Built Hardware Descriptions

M. Bowden, R. DeMaatg, M. Fachin, H. Gonzalez,
M. Haldeman, M. Larwill, C. Rotolo and J. Urish

Fermi National Accelerator Laboratory
P.O. Box 500, Batavia, Illinois 60510

April 1990

Operated by Universities Research Association Inc. under contract with the United States Department of Energy

Introduction

This binder contains individual module specifications and as-built hardware description manuals for a Silicon Strip Detector (SSD) Readout System. The modules described herein have been assembled together as designed to provide a readout system for the E771 Silicon Strip Detector. The system was used to acquire data from the SSD during the Fermilab 1991 fixed target run.

Fermi National Accelerator Laboratory

April 11, 1990

TO: Distribution
FROM: Carl Swoboda
SUBJECT: Silicon Strip Detector Readout System

The second edition specifications for all modules are included in this binder. The binder also contains papers that were submitted to the 1989 I.E.E.E. Nuclear Science Symposium. The system paper describes the overall system. Some specific details of the system have been improved since the paper was written but the paper does provide an accurate description of the proposed system implementation. The Preamplifier section contains papers specific to the detector mounted preamplifier system.

If you are interested in the detailed logic design of any of the modules listed in this binder please see Hector Gonzalez for logic schematics.

A partial list of individuals who have or are contributing to this effort are as follows:

Ed Barsotti
Mark Bowden
Dave Christian
John Chramowicz
Robert DeMaat
Miguel Fachin
Hector Gonzalez
Rick Hance
Merle Haldeman
Jim Hoff
Rick Kwarciany
Wolfgang Kowald
Mark Larwill
Garry Moore
Andy Romero
Carmen Rotolo
Matteo Recagni
Panayotis Spentzouris
Robert Trendler
Ken Treptow
John Urish
Don Walsh
Ray Yarema
Tom Zimmerman

1-16-90

A HIGH-RATE FASTBUS SILICON STRIP READOUT SYSTEM *

Carl Swoboda, Ed Barsotti, Mark Bowden, David Christian, Robert DeMaat
Miguel Fachin, Hector Gonzalez, Rick Hance, Merle Haldeman, Jim Hoff, Mark Larwill
Carmen Rotolo, Robert Trendler, Ken Treptow, John Urish, Don Walsh, Ray Yarema, Tom Zimmerman

Fermi National Accelerator Laboratory, P.O. Box 500, Batavia, Ill. 60510

Abstract

This paper describes a synchronous silicon strip readout system capable of zero deadtime readout at average trigger rates in excess of 1 MHz. The system is implemented in FASTBUS, uses pipelining techniques, and includes point-to-point fiberoptic data links to transmit detector digital data. Semi-custom ASIC chips are used to amplify, discriminate, and logically combine track data before encoding. This paper describes the overall system, each major FASTBUS module, and the functional aspects of the ASIC chips.

Introduction

Extracted beams at Fermilab retain the 53 MHz RF structure of the Tevatron accelerator. Events occur in well defined "buckets" of time which are approximately 1.5 ns long and occur every 18.9 ns. We are developing a silicon strip detector readout system capable of associating all of the information from an event with a single RF bucket and completely recovering within one or two buckets. The system operates synchronously with the 53 MHz Tevatron clock. A digital memory is used to provide a trigger delay, which is adjustable in one bucket steps to a maximum of 4.8 μs.

The system consists of amplifiers mounted on the detector[1][2] and a set of FASTBUS modules that discriminate, delay, encode, and readout detector hit information.

System Overview

A block diagram of the system is shown in Figure 1. The detector signals are amplified by pre-amplifiers mounted near the detector. The amplified signals are connected to the FASTBUS data acquisition system by high density, .025 pitch, ribbon cables.

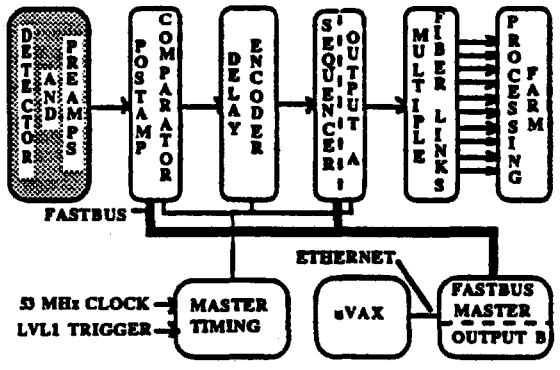


Figure 1: System Block Diagram

The Postamp/Comparator (P/C) modules discriminate the amplified signals. The discriminated signals are synchronously applied to the Delay/Encoder (D/E) modules in parallel where all channel data is delayed pending a Level 1 trigger signal. Upon the occurance of a Level 1 trigger, the hit data propagating through the memories of the D/E modules is input to the encoders. Up to 12 D/E modules simultaneously pass encoded hit data to the Sequencer (SEQ) modules. Data can be readout through a FASTBUS Master in each crate (output B), or be transmitted directly out of the Seqeuencer modules (output A), over fiberoptic cable, to individual input ports of a NEVIS [3] [4] readout and processing system.

System Board Partitioning

The FASTBUS implementation of the system is based on 128 channel increments. Each fully loaded FASTBUS crate can process a maximum of 1536 channels. Each crate can contain 12 Postamp/Comparator modules, 12 Delay/Encoder modules, a FASTBUS Smart Crate Controller, and a Sequencer module. A crate implementation is shown in Figure 2.

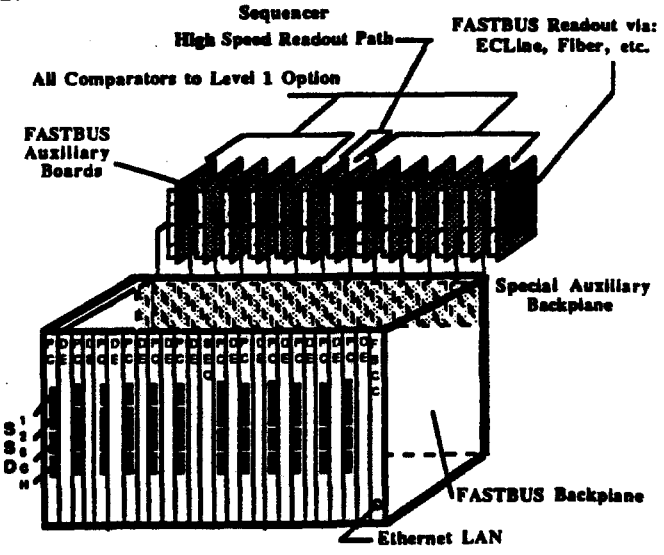


Figure 2: Crate Module Partitioning

Two different readout modes are user selectable. The FASTBUS readout path provides a relatively high bandwith output to be used with FASTBUS, VME, or CAMAC buffers on the receiving end. The Sequencer output is intended to be used to feed optical links. In the fiberoptic link case, each crate transmits data to an independent receiver buffer that is part of a high speed event processor [4]. The Data transmission rate using the fiberoptic FASTBUS Auxiliary port is 40 MBytes/Sec/Crate.

*This work performed under the auspices of the United States Department of Energy.

System Modules

Preamplifier

A silicon strip amplifier which uses a Tektronix semicustom bipolar integrated circuit has been developed [2]. It has a fast impulse response (35 ns base to base) and high gain (17 mv/fc), and is able to directly drive a differential transmission line. The amplifiers are mounted in small custom leadless chip carriers, with 25 mil pitch in order to meet density requirements. Sixteen chip carriers, or 128 channels, are mounted on a double sided printed circuit board measuring 3.5 X 4.5 inches. Each board plugs into an edge connector mounted directly on the silicon strip detector Kapton fanout. Four 64 conductor, 25 mil-pitch flat cables connect each preamplifier card to Postamp/Comparator modules. Due to the high gain-bandwidth of the amplifiers, special attention must be given to grounding and shielding [1].

Postamp/Comparator (P/C)

The P/C module amplifies, discriminates and logically combines 128 channels of silicon strip hit data. A block diagram of this module is shown in Figure 3. The major portion of the analog and digital circuitry required on this board is implemented using bipolar Application Specific Integrated Circuits (ASIC) manufactured by Tektronix. The threshold setting Digital-to-Analog Converter is implemented as a CMOS ASIC.

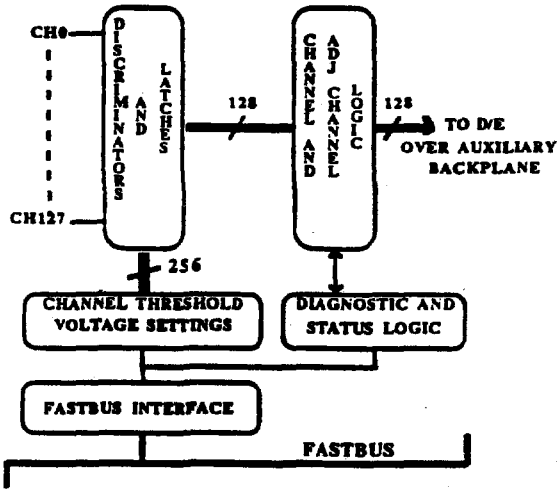


Figure 3: Postamp/Comparator Module

The following is a description of each ASIC contained on the P/C module.

ASIC 01: Two Channel Sum, Quad Latched Discriminator

A block diagram of this chip is shown in Figure 4. This ASIC receives low level linear differential signals from the preamplifiers mounted on the detector and produces analog sums of adjacent strip signals, discriminated digital signals of individual channels and discriminated signals of the analog summed adjacent channels. The **analog sum** block accepts the adjacent channel signals and provides an output proportional to the sum of its two inputs. The purpose of this block is to allow the discrimination of signals produced by particles which transverse the SSD midway between strips causing the signal to be divided between two adjacent channels.

The **discriminator** block provides an output logic level for the duration that the input signal is above a s threshold level. Each of the four discriminators has an individual threshold voltage control that allows individually set thresholds between 10 and 50 mV referred to the discriminator input. There is a built in hysteresis of 10 mV.

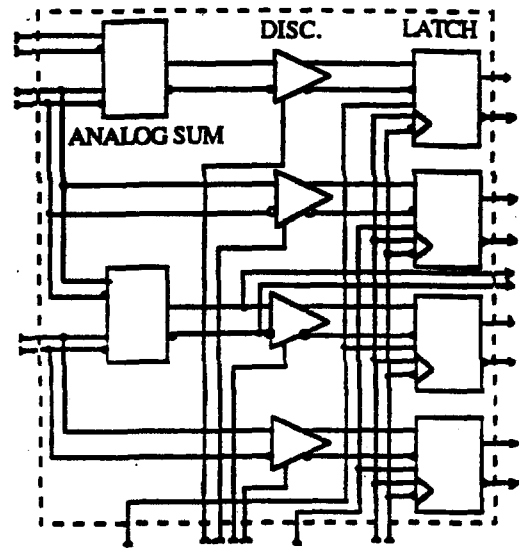


Figure 4: ASIC 01

The output of each discriminator drives a transparer latch, all four of which are controlled by a common 2.5 n. wide latch signal. In one state the latch is in the transparent mode, passing its input to the output. In the other state the output is latched ignoring input changes. There are also two Force Zero inputs, one for the individual signal latches and one for the summed signal latches that force the appropriate latch outputs to logical zeros.

ASIC 04: 5 Channel Logic and Octal NHit

The circuitry of ASIC 04 and ASIC 02 function jointly to logically combine 8 channels of track data. Block diagrams of the two ASICs are shown in Figures 5 and 6 respectively. They accept as inputs the outputs of 4 ASIC 01 chips (Two Channel Sum, Quad Latched Discriminnators). If any individual channel input is above the threshold during the latch transition, a logical high is output on that corresponding channel. If two adjacent channels have signals both of which are below threshold but the sum of the two is above threshold, a logical high is output on those two corresponding channels.

The logic can be bit sliced at any channel boundary. Chip pinout limitations on the Tektronix linear arrays dictated that of a given set of 8 channels, the lower five would be processed by ASIC 04 and the upper three by ASIC 02. Thus 16 ASIC 04/ASIC 02 pairs are capable of processing all 128 channels as one logical unit.

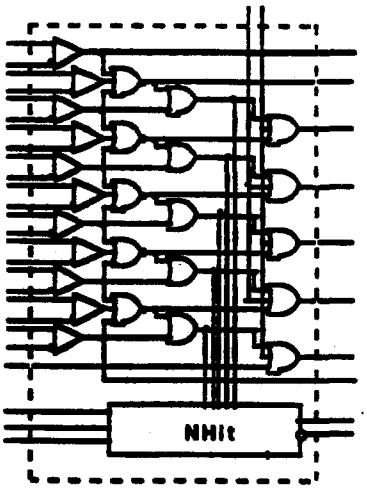


Figure 5: ASIC 04

The **NHit** circuitry provides a current output dependent on the number of channels hit. The output is approximately 100μA per strip hit. In the situation described above where individual signals are below threshold but the sum of the two is above, only 100 μA is provided thereby treating it as a single hit.

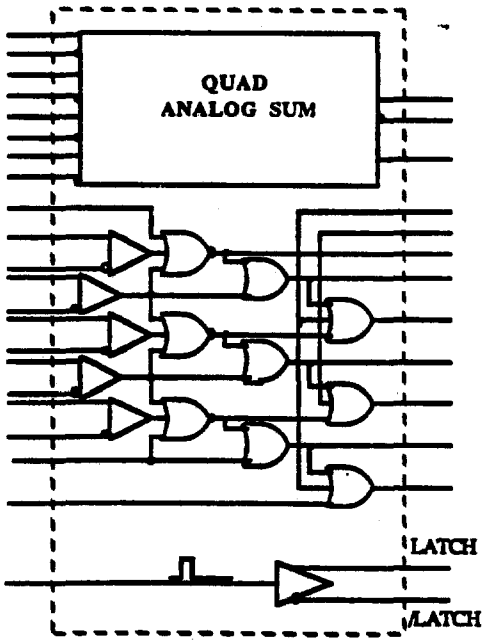


Figure 6: ASIC 02

ASIC 02: Quad Analog Sum and Latch Driver

The **Quad Analog Sum** circuit on ASIC 02 gets its inputs from analog sum outputs on ASIC 01 (two input sum circuits), thus producing an analog sum of eight adjacent channels.

The **Latch Driver** takes an ECL 2.5 ns pulse, converts it to a differential pulse which will then be used to drive the latch inputs of the four corresponding ASIC 01's.

ASIC 03: Digital to Analog Threshold Set and Readback

The chip shown in Figure 7 is a CMOS implementation of 4 D/A output channels and a single A/D readback channel. Writing and reading of this chip is accomplished through the FASTBUS system. To set a threshold for each channel and adjacent channel sum discriminator, 256 D/A outputs are required per 128 channel P/C module. 64 D/A chips are mounted on the P/C module in this case. It is optionally possible to feed more than a single channel discriminator with the same threshold level. Implementing this option minimally requires a single D/A chip with each output feeding 64 discriminators.

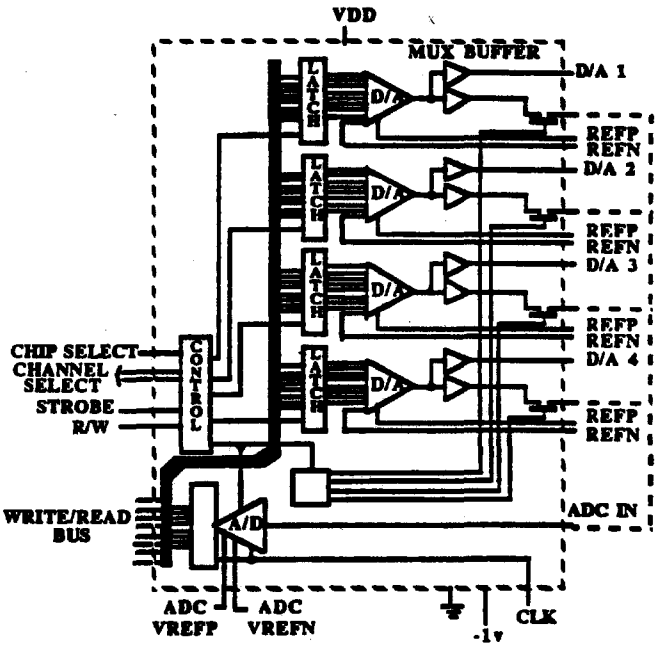


Figure 7: ASIC 03

Delay/Encoder (D/E)

The D/E module (Figure 8) accepts 128 channels of parallel data from the P/C every 18.9 ns and stores the individual channel data for up to 4.8 μsec, while a level 1 trigger decision is made. The delay is implemented using high-speed ECL memory (10422-5). The 18.9 ns time between events is split into a write cycle and read cycle.

During each 18.9 ns RF cycle, a write address counter is incremented and data is written into the delay memories. During any cycle, stored data may be read from any memory location. No detector deadtime is generated by this readout.

When a level 1 trigger occurs, the MTC generates an address that corresponds to the location in the D/E memories holding data for the RF bucket in which the triggering event took place. This address is fanned out through the SEQ modules to all of the D/E's in the readout system. Data from the RF bucket containing the triggering event and data from the previous RF bucket in time are input to a pipelined priority encoder. It takes seven clock cycles to load the encoder pipeline. After this delay, the D/E outputs an ordered list of hit strips. The address of each hit strip is encoded in a seven bit number. An adddtional bit is used to flag strips which were hit in the RF bucket previous in time to the

triggering bucket. This bit identifies hits that are likely to have been caused by an event other than the triggering event.

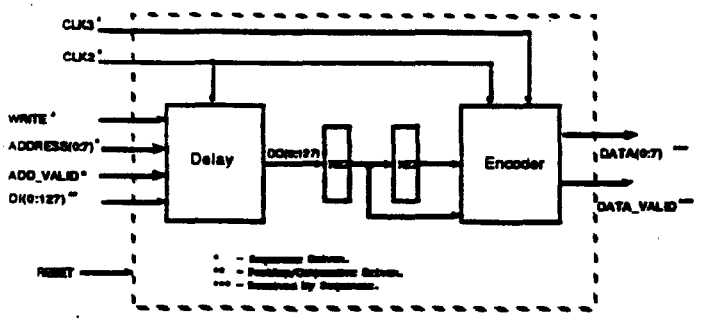


Figure 8: Delay/Encoder

Encoded data is transferred synchronously to the SEQ module over point-to-point connections on the FASTBUS auxiliary backplane. This transfer is data driven and occurs in parallel for all D/E's.

Sequencer (SEQ)

A block diagram of the SEQ is shown in Figure 9.

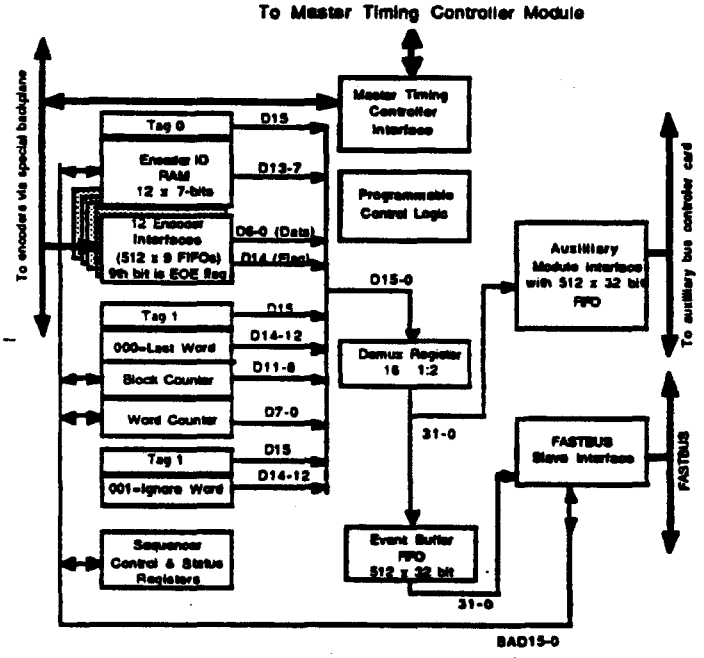


Figure 9: Sequencer Module

The SEQ module fans out triggers received from the MTC to the D/Es in the form of a hit address in the delay memories. It also accepts the conditioned 53 MHz clock from the MTC, delays it by a programmable amount, and fans out two versions of the 53 MHz clock and a 26.5 MHz clock derived from it, onto the FASTBUS auxiliary backplane. These clocks are used by the P/C to latch data, by the D/E to clock the high speed memory and the encoder pipeline, and to synchronize the data transfer from the D/E modules to the SEQ. The SEQ accepts data from the 12 D/E's in parallel and places it in 12 FIFO's. When a trigger is received, the SEQ deasserts an "Encoder Ready" signal which is input to the MTC. As soon as all D/E data has been transferred to Seqeuncer input FIFO's, the "Encoder Ready" is asserted again and the readout system is ready to process another trigger.

The Sequencer reads data out of its input FIFO's in a fixed order. The 8 bits of data from each D/E are combined with 7 bits of D/E address and stored in a FASTBUS accessible memory. A high order zero is added to distinguish data from control information. The end of event is marked by control word containing a count of the number of data words and a 7 bit "event number' that may be used to insure synchronization across the entire readout system. The sequenced data is moved two 16 bit words every 73.4 ns into a "Event Buffer FIFO" that can be accessed from FASTBUS. The data is simultaneously output through the auxiliary connector to an optional high-speed fiberoptic link. When the fiberoptic port is used, the data may still be readout of the "Event Buffer FIFO" over FASTBUS and used to compile high statistics histograms in the FSCC without reducing the readout bandwidth.

FASTBUS Smart Crate Controller (FSCC)

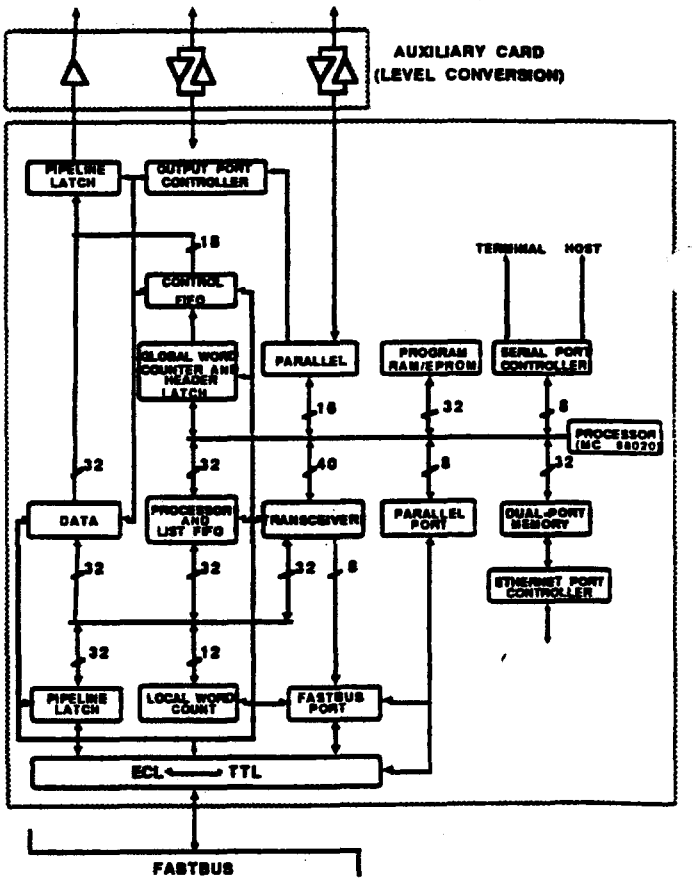


Figure 10: FSCC

The FSCC, shown in block form in Figure 10, is designed as a limited general-purpose readout device for low-occupancy front-end systems. It can execute most FASTBUS master operations directly from higher level software at norm 68020 processor speeds, typically a few hundred nsec p

instruction, and also supports faster operation through microcoded list operations. The FSCC is dual-ported from FASTBUS to the auxiliary connector to allow 100 nsec block transfers on both ports through an intermediate FIFO buffer. This buffer is designed to store the data from a single crate of front-end modules and insert a leading word count or header. The output port connects to several standard formats using an auxiliary card level adaptor. In addition to the data ports, two RS 232 lines, an Ethernet interface and trigger I/O ports are provided. The FSCC runs the PSOS real-time OS kernal and is programmable in any combination of C, assembler or microcode depending on performance requirements.

Master Timing Controller (MTC)

The Master Timing Controller generates the system clock, controls system synchronization, and generates delay memory addresses upon the receipt of Level 1 triggers from the experiments Trigger system. A block diagram of the MTC is shown in Figure 11. The MTC receives the 53 MHz Tevatron

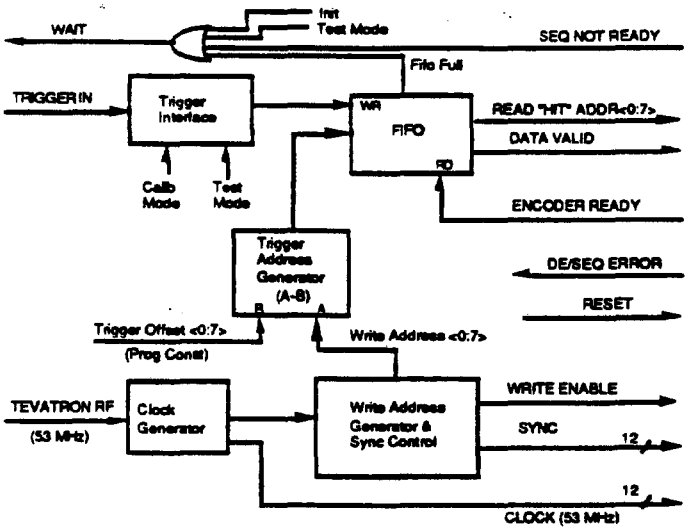


Figure 11: MTC

RF clock and establishes a 50% duty cycle clock whose phase is adjustable relative to the incoming RF. This CLOCK along with a SYNC pulse are distributed to each SEQ and eventually to all D/E's. D/E's use this clock and sync to determine write addresses for incoming data. Being synchronized, the MTC knows the current D/E write address and generates a read or "hit" address when a trigger is received. The hit address generated is offset from the write address based upon a calibration of the Trigger decision time. Hit addresses are placed into a high speed FIFO queuing up to eight trigger requests and can be accepted on successive RF buckets. The read or "hit" address output from the FIFO is broadcast to all D/E modules at a rate determined by the ENCODER READY signal summed from all D/E modules.The ability to pipeline triggers makes the system truly deadtimless at trigger rates up to the readout bandwidth. With knowledge of the read address and the D/E's current write address, the MTC detects fatal DE memory overwrite errors. However, to prevent such a condition, the system is throttled by sending a WAIT signal to the Trigger system under appropriate conditions.

Project Status

All system module logical design is complete. The FSCC is in the second prototype stage. The P/C module board layout is nearing completion. The D/E and the MTC module prototype boards are due in February 1990. The SEQ module board design has started and will be completed in February. Prototype printed circuit boards of each of the system modules are expected to be available in early March for initital integrated testing.

Acknowledgements

The authors would like to thank Garry Moore, Betty Conrin, Rick Kwarciany, John Chramowicz, Rick Van Conant, Scott Holm, Bruce Merkel, and our tireless cooperative education student, Mike Kodner for the effort they expended on this project. Ruth Pordes and Mark Bernett of the Computing Division were instrumental in developing the software needed to make the FSCC a usable product.

Bibliography

[1] Ray Yarema, Tom Zimmerman, A High Speed, High Gain Preamplifier for Silicon Strip Detectors, 1989 Nuclear Science Symposium Proceedings

[2] Tom Zimmerman, A High-Speed, Low Noise ASIC Preamplifier for Silicon Strip Detectors, 1989 Nuclear Science Symposium Proceedings

[3] J.A. Crittenden et al., A Data Acquisition System for Elementary Particle Physics, IEEE Transactions on Nuclear Science, Vol. NS-31, No. 5, October 1984

[4] W. Sippach, G. Benenson, B. Knapp, Real Time Processing of Detector Data, IEEE Transactions on Nuclear Science, Vol. NS-27, No.1, February 1980

A High Speed, High Gain Preamplifier System for Silicon Strip Detectors

R. J. Yarema, T. Zimmerman
Fermi National Accelerator Laboratory*
P. O. Box 500, Batavia, IL 60510

Abstract

A high speed, high gain amplifier system has been designed for silicon strip detectors. The amplifier has been designed using a semicustom bipolar linear array. This paper focuses on a practical integration of this amplifier into a working system. To maximize board density and reduce cost a new, relatively inexpensive custom chip carrier was designed. The design approach could be useful for other custom or semicustom chip designs. Insight into the design of a 128 channel preamplifier circuit board for low noise and low crosstalk using the new carrier is presented.

High channel density presents challenges in cabling. A relatively new high density cable with mass termination capability was used for transmitting signals from the preamp to the discriminator boards. As a part of the overall design, the approach taken for shielding of the detector, preamplifier cards, and output cables is discussed.

Introduction

The amplifier described in this paper is part of a high speed silicon strip detector readout system being developed at Fermilab [1]. This system will be used by two experiments, E-771 and E-789, in the 1990 fixed target run. Both of these experiments will operate at very high interaction rates with the goal of accumulating for study large samples of B particles. Each of these experiments will contain more than 10000 channels of silicon strip electronics. Although this paper is generally applicable to both experiments, the requirements for E-771 will be used as a specific example.

Front End System

The silicon strip detector for E-771 is comprised of 24 planes of silicon, each with 688 strips. Plane to plane spacing averages 0.5 inches. Signals from a silicon plane are fanned out over fine pitch circuitry on flexible Kapton to six high density connectors for preamplifier cards. Each preamplifier card has provision for 128 channels and has differential outputs for each of the channels. Thus there are 256 output signals per card. Outputs from a preamp card are transmitted over four fine pitch ribbon cables which are about 20 feet long to discriminator circuits housed in FASTBUS crates.

*Operated by the Universities Research Association under a contract from the U. S. Department of Energy.

The preamplifier packaging design was driven by a need to minimize the Kapton length and reduce the associated capacitance in order to improve the signal to noise performance of the system. Since the silicon planes are physically close together, a circuit board which could be mounted on nearly the same pitch as the silicon planes was designed. The size of the circuit board, the pitch of the pins on the preamplifier chips, the choice of input and output connectors, and choice of output cable were all driven by the desire to place the preamplifiers close to the silicon planes.

Another design criterion was that preamplifier cards be easily replaceable and repairable. These factors influence the type of input and output connectors used and the construction of the preamplifier board.

To achieve the desired goals, an integrated design of the front end electronics was implemented. The integrated design in this context means considering the chip design, chip packaging design, board design, heat dissipation, and shielding design all at the same time.

Preamplifier

The preamplifier response for a charge impulse is required to be 20 to 40 ns baseline to baseline and have an equivalent input noise of less than 1800 electrons RMS. To achieve the design in a physically small package with reasonably fast construction time, a semicustom, bipolar chip was designed using the Tektronix Quickchip 2S linear array. Four channels of the preamplifier are designed on a single die. The design and performance of this preamplifier chip is covered in another paper presented at the 1989 Nuclear Science Symposium [2].

The preamplifier chip was designed to maximize the isolation between input and output pads by locating the pads far apart and to make power supply busing across many chips easy. Figure 1 is a bonding pad diagram which shows that the inputs and outputs are on opposite ends of the chip to reduce feedback. Power supply connections are made on the sides of the chip to permit power busing under the chip. Identical power connections are made on each side of the die to facilitate the mounting of the chips in the chip carrier package which was chosen.

Chip Packaging

Each circuit board is to have 128 channels and thus 32 four channel dies mounted on it. To fit the chips onto the preamplifier board, the chips had to be mounted relatively

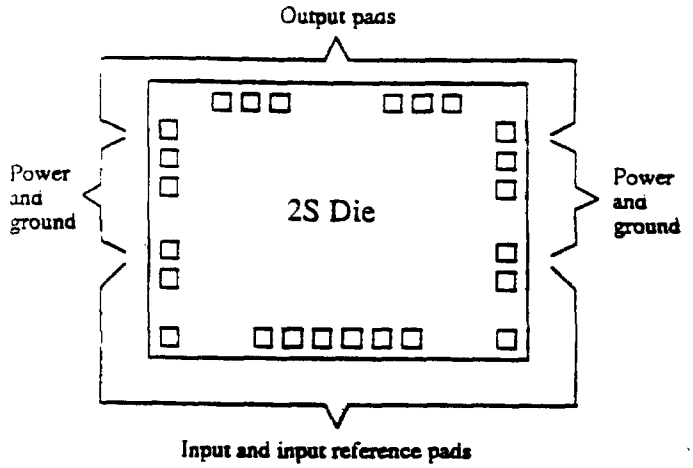


Figure 1 - Preamplifier Die Bonding Pads

close to each other. Chip on board (COB) mounting of the bare dies was considered to be impractical because of the large number of dies to be mounted. Secondly, COB does not readily permit replacement of a bad die after assembly. Thus a chip carrier was desired which would facilitate replacement of bad chips. To fit the required number of channels onto a board, a leadless surface mount chip carrier with a pitch of 25 mils was necessary. Standard packages which would meet the space requirements were not available. Therefore a custom package was designed.

The type of package designed is manufactured by Tectonic, Ltd in England and sold under the trade name of EPIC [3] by S A Communications of San Diego, CA. Advantages of the package in addition to full choice of package dimensions include ease of design, fast delivery, and relatively low cost. The primary disadvantages are that the package is not well sealed and replacement of a faulty device requires special care.

The EPIC chip carrier is made of B-T (bismaleimide-triazine) which is similar to the FR-4 material that is commonly used to make printed circuit boards. Thus, there is no temperature coefficient mismatch between the package and the printed circuit board to cause solder joint

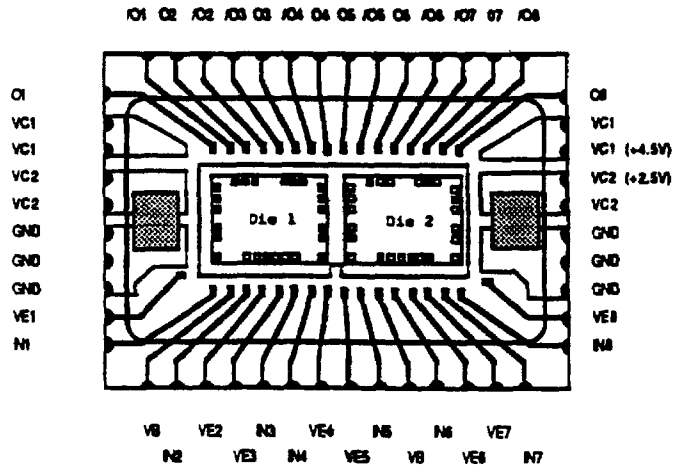


Figure 2 - Custom EPIC Chip Carrier Layout

failure problems. The glass transition temperature for B-T, however, is about 210°C, which makes it more suitable for gold ball bonding and surface mount reflow soldering.

To make most effective use of the space on the circuit board and reduce the packaging cost, the package was designed to have space for two dies (eight channels of preamplifier). The package was ordered by sending the layout shown in Figure 2 to the manufacturer's representative in the USA. Chip carriers were normally returned in about 4 weeks. As can be seen, provision was made inside the chip carrier for 2 internal bypass capacitors for critical power supply voltages. This feature was very helpful in preventing the output drivers on the chip from oscillating. Effectively, the assembled chip carrier becomes a mini hybrid with surface mount provisions. Figure 3 shows an assembled chip carrier with the lid removed.

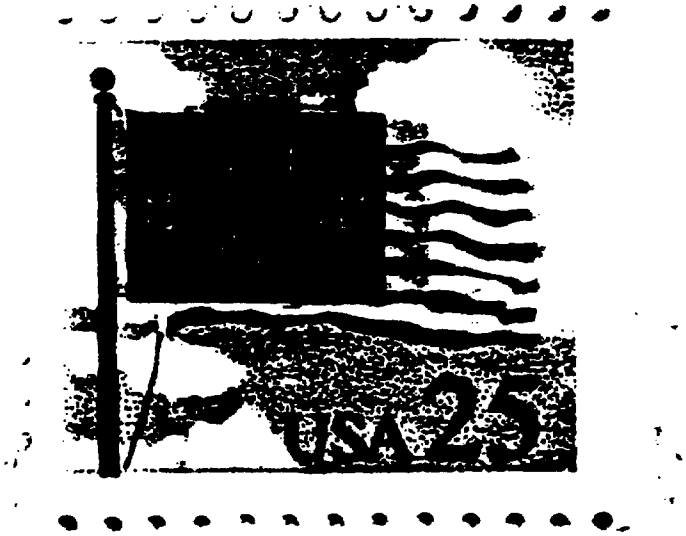


Figure 3 - Assembled EPIC Chip Carrier with Lid Removed to show 8 Preamplifier Channels.

The EPIC type of package has received only limited attention in the USA. However, the package has been extensively tested and used by British Telecon in the United Kingdom. Accelerated life tests performed by the manufacturer show the package to be highly reliable.

A preliminary measurement of the thermal impedance of the EPIC package to the circuit board described in the next section is about 80°C/watt. The preamplifiers dissipate about 350 mw per EPIC package.

Due to the inherent construction of the EPIC packages, they are not well sealed against moisture. To provide adequate protection, the chip surfaces must be coated or sealed to prevent contamination of the bonding pads. Two different materials have been tested on the preamplifiers. One is a polyimide surface coating (Rely-imide #210D) made by M & T Chemicals and the other is a glob top encapsulant (FP4323) made by Hysol for COB applications. Packages with both types of coating were thermally cycled. Packages with heavy coats of the polyimide material that covered the bonding wires (not recommended by the manufacturer) showed damage to the bonding wires after several days of thermal cycling. Similar

tests performed with thin coatings, 1-2 mils, and no coating of the bond wires showed no damage. The encapsulant material has shown no problems. Both materials when properly applied appear to be acceptable solutions to the moisture problem.

Initially, poor wire bonds to the package were encountered when using a gold wire bonder. This was apparently due to thin gold plating (8 to 15 microinches) inside the carriers. After ordering packages with a minimum of 32 microinches of gold, the problem disappeared.

Solder mounting and replacement of the EPIC packages requires special attention. Production assembly was accomplished with IR solder reflow techniques. Because of the fine pitch components, stencils rather than screens were used to deposit the solder paste. Special fine mesh solderpaste with low slump characteristics was necessary. Replacement of a single EPIC package by an experienced person requires about 10 minutes.

The EPIC package is relatively inexpensive for a custom package. Nonreccuring engineering charges are about $1500 and the cost per package is $1.06 for a few thousand pieces.

Board Layout

As has been mentioned, the printed circuit board and the chip carrier are an integrated design. Each printed circuit board which is 3.5 inches by 4.5 inches has 128 channels of preamplifiers. Seventy percent of the board space is devoted to input and output connectors. Eight chip carriers are mounted in a row on each side of the board. Power is brought into the center of the board with a cable and then distributed under the chips on each side of the board to minimize voltage drop. Power supply bypassing is done as close to the chip carriers as possible.

The circuit board has four layers. The center two layers are primarily ground plane, and act as a low impedance amplifier reference. They also provide impedance control for the output traces. Each channel of amplifier has a dedicated reference, or Vee, connection. Each four channel chip has a common ground connection for the output driver pulldown resistors on the chip. It is important that these all connect directly to the same low impedance reference, or ground plane, to reduce crosstalk. In fact, the common pulldown resistor ground connection is made through three parallel package pins to insure a low inductance connection to the board. Multiple vias are used to connect the chip ground pins to the ground planes in the center of the board. All bypassing is as tight as possible to the ground plane.

The ground plane layers also distribute the heat from the chips to the entire board. The board thus tends to act as a cooling fin. There are 144 circuit boards distributed over four sides of the detector. Forced air is used to cool the circuit boards.

Figure 4 shows an assembled card. Input signals are brought onto the card from the detector via an inexpensive, 132 pin 50 mil pitch card edge connector. This connector permits easy insertion and extraction of 128 channels of electronics with less than 10 pounds of force. Two 64 pin, high density connectors on each side of the card are used to send signals to the discriminator boards.

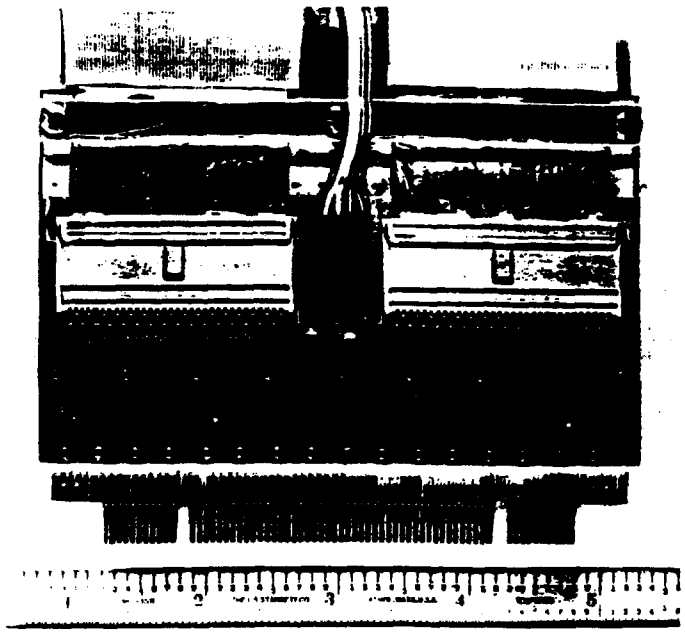


Figure 4 -128 Channel Preamplifier Card With Connectors.

Shielding Design

A double RF shield is employed around the detector as shown in Figure 5. The inner shield encloses all of the silicon strip detector planes. Penetrations are made through the shield for the preamplifier cards to pass through and plug into connectors mounted on the detectors inside. Small beryllium copper RF gasketing strips are used to make contact from the shield to the exposed ground plane on the two sides of the preamplifier card near the card edge connector. See Figure 4. The RF shield is carried through the circuit card by means of a series of via holes located where the gasketing contacts the board. This arrangement maintains RF shield integrity which would have been reduced by the card slots. Also, this allows a very low impedance connection between the RF shield and the amplifier grounds. This was found to be absolutely essential in eliminating coupling between the preamplifier input leads coming from the detector and the higher signal level preamplifier outputs.

A second RF shield is placed around all of the preamplifier cards and the inner RF shield. Power supply leads for the preamplifiers are brought through the outer shield in four locations and then distributed. The outer shield eliminates pick up on the power distribution cables.

Preamplifier output cable bundles passing through the outer RF shield are shielded to reduce pick up on the cables. The bundle shields are terminated to the outer RF shield at the detector to effectively extend the outer shield to include the cables. The bundle shields are also connected at the receiving end to the FASTBUS crate racks.

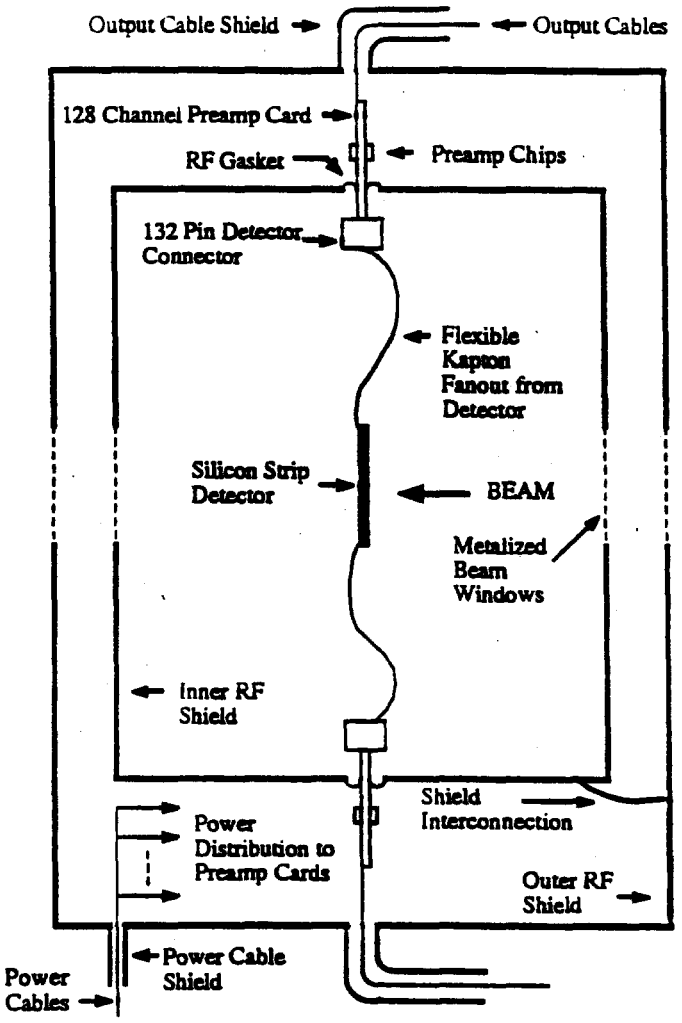


Figure 5 - Silicon Strip Detector and Preamplifier Card Shielding

Output Cables

A mass terminable, flexible, 64 conductor, 25 mil pitch ribbon cable was chosen to transmit the large number of signals at the output of each preamplifier card to the discriminator cards. The cable which is made by Spectra-Strip (#133-3013-064) has an embedded ground plane for impedance control and cross talk reduction. All wires in the cable are used as signal conductors.

Four of the cables are connected to each preamplifier card, for a total of 576 cables in the system. Each cable is relatively light weight, flexible, and stacks well into bundles for shielding.

For this application, pairs of conductors are driven differentially. The differential impedance of the cable is 100 ohms. Although the output signals are differential, the ground plane of the cable is connected to the preamplifier ground plane to provide a return path for imbalanced signals. Also, fixing each cable ground plane to the same reference reduces coupling which can cause oscillations. To make the connection to the preamp ground plane, the ground plane on the cable is folded back and wrapped with conductive adhesive copper tape and then clamped to the preamplifier ground plane as shown in Figure 4. In addition to making the ground connection to the circuit board. the clamp also doubles as a cable strain relief.

At the receiving end, each signal conductor is AC coupled and terminated with 50 ohms to a common po which is connected to ground. Forward cross talk between adjacent conductor pairs in the cable is about 0.5% for a ten foot cable with 5 ns rise time pulse. This is quite acceptable for the silicon strip application.

Preamplifier Power

A "quiet power" source is provided for the preamplifier cards to eliminate noise injection from the power source. A separate double shielded transformer is used to provide AC power to the preamp power supplies. Only linear power supplies are used for the preamps. Power cables from the power supplies to the detector are shielded outside the RF shield and bypassed just inside the outer RF shield to stop any noise transmitted down the cables from entering the RF enclosure.

Once the Power supply voltages are brought inside the outer shield, they are distributed to the individual preamplifier cards by means of short ribbon cables. Each of the two power supply voltages on the ribbon cable to each card is fused. The fuses are located inside the outer RF shield. Easy access doors to the fuses are provided for maintenance purposes.

Summary

The front end electronics for the silicon strip detector i packaging problem as well as an electrical design problem. Both factors needed to be considered in the early design stages of the project to build a system which not only meets the electrical design criteria, but is manufactureable and readily repairable. In completing the design, packaging ideas which should be applicable to other systems were developed.

Acknowledgments

The authors are particularly grateful for the help which Dan Graupman of the Research Facilities Group at Fermilab provided in laying out the preamplifier and various test boards. We would also like to thank Ted Bohn for his valuable assistance in assembling and testing various parts of the system.

References

1) C. Swoboda, et al., "Silicon Strip High-Rate FASTBUS Readout System", Presented at the 1989 Nuclear Science Symposium.

2) T. Zimmerman, "A High Speed, Low Noise ASIC Preamplifier for Silicon Strip Detectors", Presented at the 1989 Nuclear Science Symposium.

3) N. Sinnadurai, "EPIC: A Cost-Effective Plastic C. Carrier for VLSI Packaging", IEEE Transactions on Components, Hybrids, and Manufacturing Technology, Vol. CHMT-8, No 3, September 1985.

A High Speed, Low Noise ASIC Preamplifier for Silicon Strip Detectors

T. Zimmerman
Fermi National Accelerator Laboratory*
P. O. Box 500, Batavia, IL 60510

Abstract

A first version Tektronix Quickchip semicustom ASIC preamplifier for silicon strip detectors was reported in Oct., 1988[1]. An improved version, QPA02, has been designed which incorporates laser trimming of nichrome resistors as a means of compensating for chip resistance process variations, which affected the response of the first version. This allows chips with randomly varying resistance values to be trimmed to have a standard output pulse shape. The QPA02 also has improved performance specifications. This paper describes the design philosophy and specifications of the QPA02, and the test results.

Introduction

The QPA02 is a high speed bipolar transimpedance amplifier built using a Tektronix "Quickchip 2S" semicustom linear array. It converts an impulse of charge to a fast voltage pulse. It was designed for use as a silicon strip amplifier for Fermilab E-771[2], but will be used in E-789 and may have other applications as well. E-771 requires an amplifier with a fast response to a charge impulse input, since the time between beam buckets is about 19 ns. The QPA02 has an impulse response that returns to baseline in less than 38 ns, or two beam buckets, for a detector capacitance of 20 pf. A high gain amplifier was designed (of order 15 mv/fc) so that output signals traveling over long cables could directly drive discriminators. This also reduces sensitivity to external noise pickup. A buffered differential output was necessary to drive flat cable with maximum noise immunity. E-771 space constraints forced a high channel density, which motivated an integrated circuit design.

The Tektronix Quickchip linear array was chosen for this design over other available high performance bipolar linear arrays because Tektronix markets its own design tools which facilitate accurate simulation and layout, and because custom laser trimmable nichrome resistors are available on Quickchip. A first prototype amplifier, QPA01, was designed, produced, and tested[1]. The response of the amplifier differed somewhat from the nominal simulations due to process variations in on-chip resistor values. This emphasizes one of the major challenges in analog integrated circuit design. Chip resistor values can vary widely from run to run and are unpredictable. However, the designer can make use of the fact that on any given chip the resistors are matched to within 1%. The second version amplifier, QPA02, is designed to use a nichrome resistor laser trimming scheme to partially compensate for these resistance variations, resulting in a more standardized output pulse response.

*Operated by the Universities Research Association under a contract from the U. S. Department of Energy.

The Quickchip Design Process

Designing with Quickchips is a well defined process. Normally a design is simulated and its performance understood by using TSPICE (a Tektronix proprietary enhanced version of SPICE 2G) before commencing layout. This is done using Tektronix supplied component libraries. Then a die selection is made from several available types. This is usually based on the number of components and I/O pads needed per chip. Layout is then performed using the QUICKIC (Tektronix version of KIC) layout editor. Two levels of metal are available to interconnect components. Sometimes layout constraints force component changes. This necessitates a return to TSPICE to verify any design changes. A design review at Tektronix checks the integrity of the design. Normally a prototype run of chips is made and performance checked before ordering production quantities.

Amplifier Design

The QPA02 is a two stage amplifer. A schematic is shown in Fig. 1. The first stage, or preamp, is a transimpedance feedback amplifier in the common emitter configuration to convert a charge input to a voltage output. The second stage is a differential voltage amplifier. Its function is to boost the gain, shape the preamp output, and provide differential outputs to drive a transmission line. A "reference" preamp stage, similar to the input stage, is used to provide DC tracking to the second stage. This is necessary since DC bias levels are somewhat uncertain due to process variations and temperature effects. This reference stage has no input capacitance and is bandwidth limited with a relatively large on-chip capacitor. Therefore it does not contribute significantly to the output noise.

Preamp

The preamp input transistor Q1 is a large area transistor to minimize noise due to base resistance. A cascode transistor, Q2, limits Miller capacitance at the input. The cascode base

bias voltage is supplied by a simple diode string. An emitter follower, Q3, buffers the cascode collector to the second stage. A feedback resistor is connected from a variable attenuator (driven by the follower) back to the input.

This stage has two important open loop poles, one significantly higher than the other. The feedback resistor value is selected to give a slight overshoot in the impulse response. (This overshoot is smoothed out by the second stage shaper and causes the output to have a quicker return to baseline). The dominant open loop pole is formed by the capacitance at the input node (dominated by the detector capacitance C_d) and the input resistance R_π. This can be written as

$$f_1 = 1/2\pi R_\pi C_d = g_m/2\pi\beta C_d.$$

Since g_m is inversely proportional to I_c, and I_c is nearly proportional to the collector resistance R_c, f_1 is then proportional to $1/R_cC_d$, and is nominally around 2.5 MHz. The second pole is formed by the cascode collector node capacitance, C_c, and the collector resistor, and can be written as $f_2 = 1/2\pi R_cC_c$. Its nominal value is 65 MHz. The DC open loop transresistance is approximately βR_c. Without the cascode transistor, the amplifier would have only one important pole, formed by C_d added to the Miller capacitance of Q1. Such a configuration would be more sensitive to process variations in input transistor parasitic capacitance, and would have a longer impulse response fall time. By using the cascode, however, the response becomes more sensitive to variations in input capacitance. Since detector capacitances should be relatively uniform channel to channel compromise is acceptable. The nominal design assumes ... pf input capacitance, but other capacitances can be used. Supply voltage can then be used to adjust the impulse response if desired.

Since the two poles are widely separated, and the feedback is set such that the loop gain falls to one at about 25 MHz, the preamp closed loop bandwidth and phase margin are determined mainly by f_2 and R_{fb}. Thus process variations in resistance will affect the amplifier's response. Assuming fixed C_d, the value of R_c has little effect on the open loop gain at f_2, since a change in R_c moves f_1 and the DC open loop gain in opposite directions, preserving gain bandwidth. In order to stabilize the response in the presence of process variations, the second pole frequency and the amount of feedback must be held constant. A scheme in which R_c and R_{fb} were formed from nichrome and then laser trimmed was considered. However, the values required were too large to be practically designed as nichrome trims. Also, there is no easy way of monitoring the resistance values while trimming to know when the correct value has been reached. Therefore, a method was developed which uses small value nichrome resistors as "fuses" which can be left in or cut out by laser. There are five different discrete configurations to which the amplifier can be "trimmed" by cutting different combinations of fuses. These configurations are assigned to evenly spaced portions range of possible chip resistance variation. A test re...or

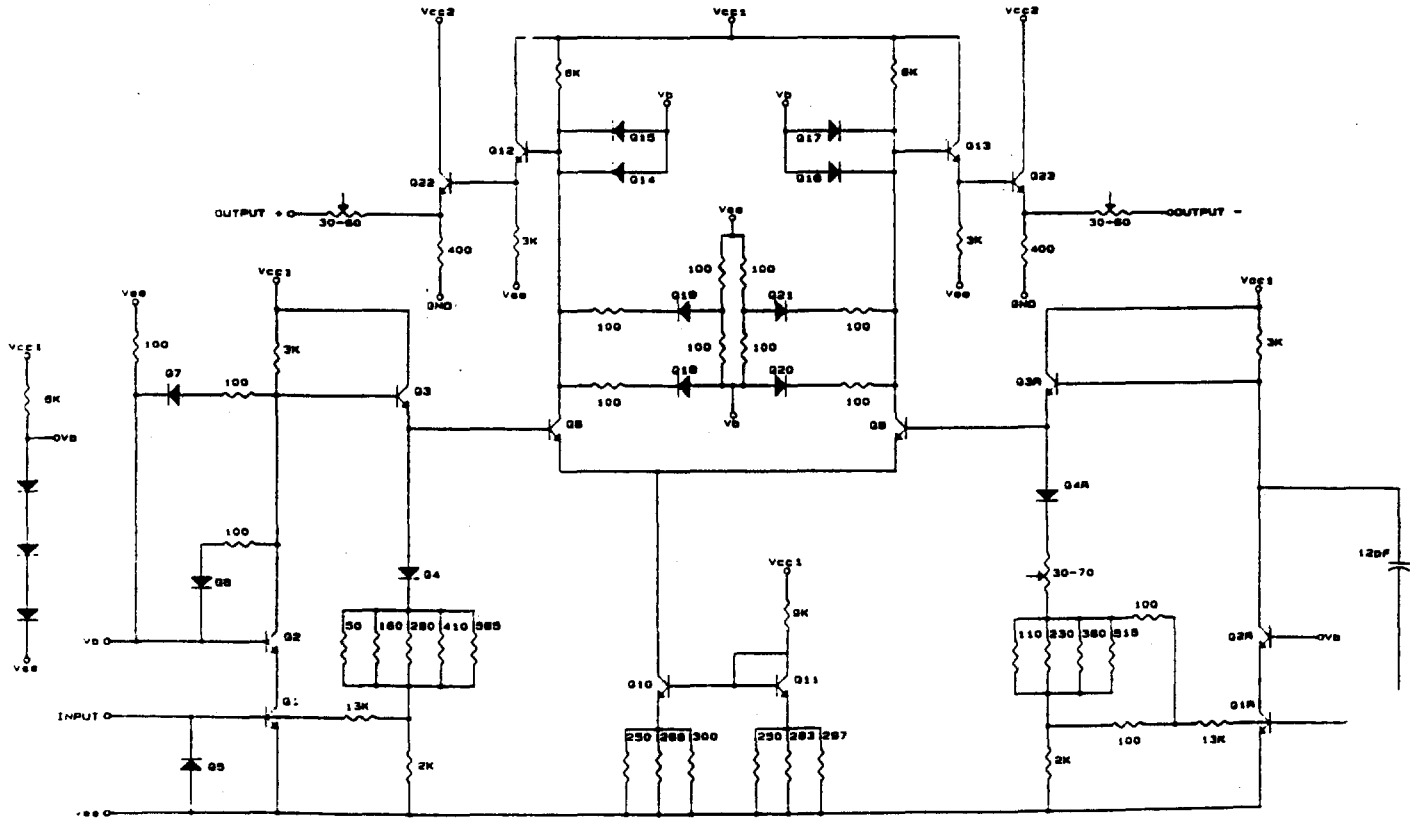


Fig. 1 - QPA02 Schematic

measurement determines which configuration is appropriate for any given chip. Nichrome laser cuts can then be made before the chip is powered up.

The frequency of the second pole is held at 65 MHz by changing the capacitance on the cascode collector node, based on the test resistor measurement. Two small capacitors in the form of back biased junctions are connected to this node through nichrome fuses. The value of one of these junction capacitors can be set to one of two levels by changing the value of back bias voltage via nichrome fuses. Thus a total of five different values of capacitance are available at this node. The amount of feedback is held at a constant value by trimming an attenuator which drives the feedback resistor (again, in discrete steps). This is done by cutting out four of five parallel nichrome resistors, based on the test resistor measurement. The response and phase margin of the preamp have been stabilized against chip resistance variation by using this technique.

The reference stage used for DC balance contains similar feedback attenuator trims, but the other trims are not necessary. A continuously trimmable nichrome resistor is added in order to remove any remaining DC imbalance due to transistor Vbe or Beta mismatch. This is an active trim done by monitoring the DC output voltage while the device is powered.

Amplifier/Shaper

The second stage is a differential pair, Q8 and Q9, which is biased by a degenerated current mirror, Q10 and Q11. This stage provides an amplified differential signal at the collectors of the differential pair. Each collector is buffered in order to drive a transmission line. The output is bandwidth limited by adding capacitance to the collector nodes in the form of back biased junctions. This shapes the signal and limits the noise. The nominal shaped signal peaks at about 10 ns and is less than 35 ns base to base. The first stage overshoot helps speed up the fall time at the output. Since the shaping depends on the RC time constant at the collector, the chip resistance value affects this. Therefore, a nichrome trim scheme similar to that used in the first stage is implemented here to trim the capacitance to one of five discrete values. Thus the shaping time constant will remain relatively constant.

The preceding scheme allows a standard output pulse shape over a range of chip resistance values. However, amplitude differences exist for the different cases since the resistance affects the gain. Therefore, the current mirror degeneration resistance in the second stage is formed from parallel nichromes of different values that were determined empirically using TSPICE. By cutting out two of the three resistors in each leg, the gain can be set such that the pulse height will be at a fixed value.

The output section of the second stage consists of small area emitter followers, Q12 and Q13, to buffer the collectors and drive larger output transistors, Q22 and Q23. An internal pulldown resistor is used to bias each output transistor at about 4 ma. An external pulldown may be added to increase the dynamic range if desired. A laser trimmable back termination is provided in series with each output.

QPA02 Layout

The Quickchip 2S die, the smallest available, was chosen for the layout since it contained the correct number of pads for four channels, and because the component layout was fortuitous. The pad assignments are shown in Fig. 2. The channels are laid out in columns, with input pads on the bottom and output pads on the top. Power supply pads are on either side. The preamplifier stages are on the bottom half of the chip, and the amplifier/shaper stages are on the top half. This arrangement maximizes isolation between inputs and outputs. Fig. 3 shows two chips bonded into a custom chip

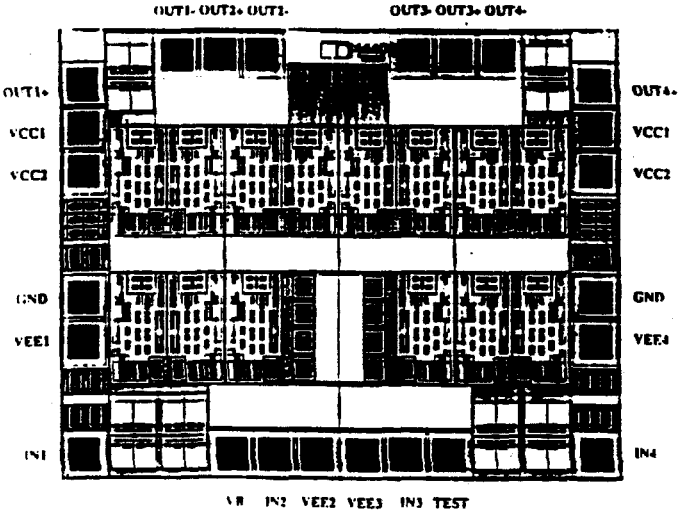


Fig. 2 - Quickchip 2S die with pad assignments

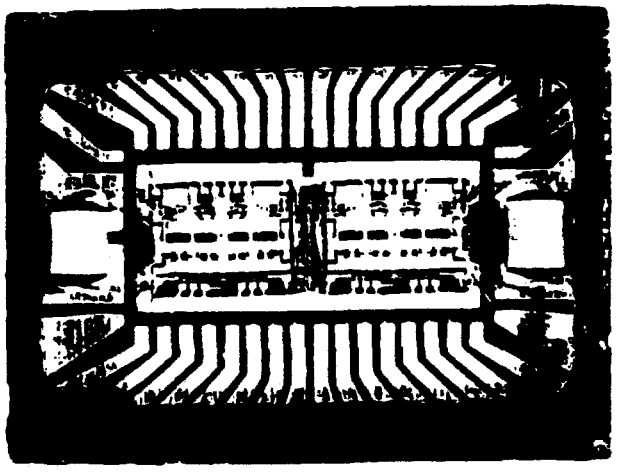


Fig. 3 - Two QPA02's in custom chip carrier

carrier. Small bypass capacitors for the output driver supply are also mounted inside the carrier.

A separate reference (VEE) pad is necessary for each channel to avoid common impedance crosstalk. Output driver pulldown resistors are included on the chip to avoid DC currents in output cables. These are all connected to a common point (GND). When using QPA02, GND and VEE pads should all be referenced to the same low impedance ground plane.

The output driver transistors are connected to a separate power pad, VCC2. This avoids the problem of high output currents coupling to the input circuitry. Also, this allows VCC2 to be independently adjusted. VCC2 should not be more than 2.0 volts lower than VCC1. For ease of use, VCC2 may be run at the same supply voltage as VCC1. However, this will result in increased power dissipation.

The cascode bias, VB, is shared between all channels on a chip. It is connected to a pad so that it can be externally bypassed. This is necessary to reduce crosstalk and to limit random noise components.

Since this amplifier has a large gain-bandwidth, it requires special care in grounding and shielding for practical use. This is the subject of another paper presented at these proceedings[3].

Performance and Test Results

Prototype chips were supplied by Tektronix for evaluation. Fig. 4 is a diagram of the test board used. For all tests, VCC1 was set at 4.5 volts and VCC2 at 2.5 volts. Response and noise measurements were made for a variety of input capacitances and compared to TSPICE predicted values. In all cases, the measured results closely matched the TSPICE values, proving the value of design simulation.

Fig. 5 is the measured impulse response for a charge input of 4 fc, taken at one output. This translates to a differential impulse gain of 17 mv/fc for Cin=20 pf. Fig. 6 illustrates how VCC1 can be varied to adjust the impulse response. In this case, VCC1 is used to keep the same pulse width for a variety of input capacitances. The QPA02 has a dynamic range of approximately 30 fc at the input before the

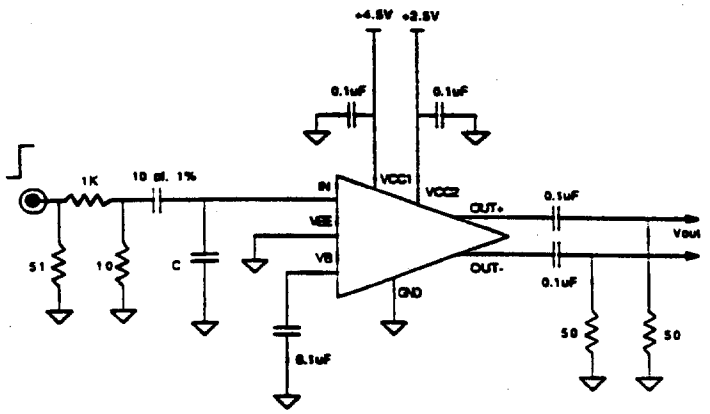


Fig. 4 - QPA02 test circuit diagram

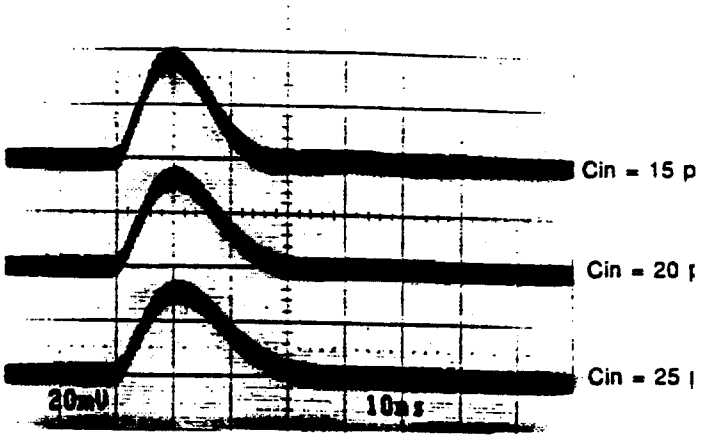


Fig. 5 - Impulse response

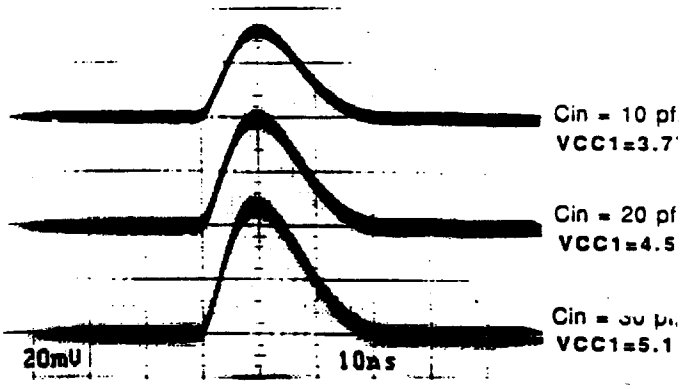


Fig. 6 - Impulse response adjusted with VCC1

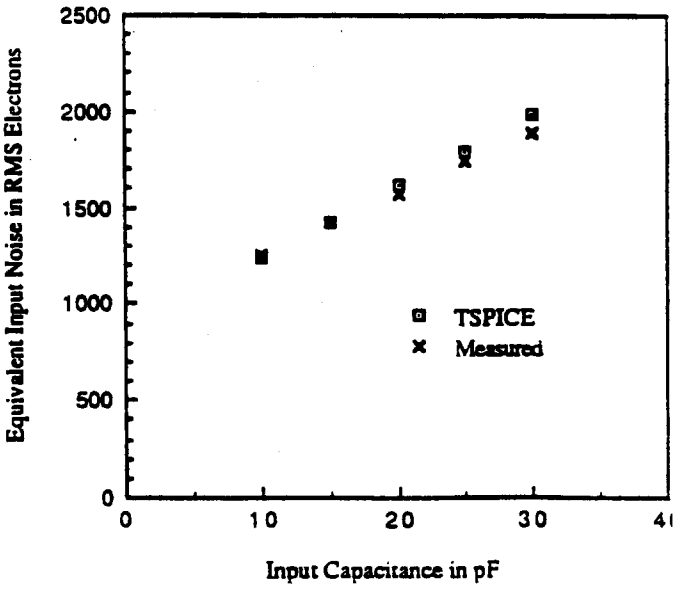


Fig. 7

outputs begin saturating. However, linearity is degraded above 10 fc.

The noise performance was measured using a LeCroy 620AL discriminator[4] and is shown in Fig. 7. The input noise is approximately 1570 electrons for Cin=20 pf.

The power consumption of the QPA02 is approximately 45 mw per channel. Channel to channel crosstalk is measured at typically 0.5%. The input impedance is approximately 200 ohms.

Conclusions

Semicustom linear arrays can be a fast and reliable way to produce integrated amplifiers for high energy physics. The QPA02 has been tested and demonstrated to be an effective silicon strip amplifier for high rate applications, performing as modeled. Other applications may exist which can use this amplifier or a modified version of this amplifier.

Acknowledgements

The author would like to thank Ray Yarema and Dave Christian for their frequent and sound advice. Mitch Newcomer of the University of Pennsylvania provided the foundation for the initial work. Also, Wink Gross of Tektronix provided helpful suggestions.

References

[1] D. Christian, et. al., "The Development of Two ASIC's for a Fast Silicon Strip Detector Readout System", IEEE Trans. on Nucl. Sci. NS-36 (1988) 507-511

[2] C. Swoboda, et al., "A High-Rate FASTBUS Silicon Strip Readout System", presented at the 1989 Nuclear Science Symposium.

[3] R.J. Yarema, T. Zimmerman, "A High Speed, High Gain Preamplifier System for Silicon Strip Detectors", presented at the 1989 Nuclear Science Symposium.

[4] This method is described by P. Jarron and M. Goyot, NIM 226 (1984) 156-162

THE DEVELOPMENT OF TWO ASIC'S FOR A FAST SILICON STRIP DETECTOR READOUT SYSTEM

D. Christian, M. Haldeman, R. Yarema, and T. Zimmerman
Fermilab*
P.O. Box 500 Batavia, IL 60510

F.M. Newcomer and R. Van Berg
University of Pennsylvania
Philadelphia, PA 19104

Introduction

A high speed, low noise readout system for silicon strip detectors is being developed for Fermilab E771, which will begin taking data in 1989. E771 is a fixed target experiment designed to study the production of B hadrons by an 800 GeV/c proton beam. The initial focus will be on events in which a B particle decays into a final state including a J/ψ, which subsequently decays into $\mu^+\mu^-$, and on events containing a single high transverse momentum muon. In order to accumulate the largest possible sample of B particles, the experiment will operate at a rate of up to 2×10^8 beam protons per second and 10^7 interactions per second. The experimental apparatus will consist of an open geometry magnetic spectrometer featuring good muon and electron identification and a 16000 channel silicon microstrip vertex detector. This paper will review the design and prototyping of two application specific integrated circuits (ASIC's), an amplifier and a discriminator, which are being produced for the silicon strip detector readout system.

A high rate system must be able to distinguish information corresponding to a desired event from information coming from other events occurring nearby in time. In order to be efficient, it must also recover from a measurement quickly. Extracted beams at Fermilab retain the 53 MHz RF structure of the Tevatron accelerator. This means that events occur in well defined "buckets" of time which are approximately 1.5 ns long and occur every 18.9 ns. The silicon strip detectors which will be used in E771 are 300μ thick and will operate with a bias voltage high enough so that most of the signal will be induced in less than 10 ns, and all of the current will be induced in less than 20 ns. In order to preserve this resolution and memory time, an amplifier must be built which not only has high gain and low noise (the induced charge totals only 24000 electrons), but also has high bandwidth.

Technology

Recent developments in the micro-electronics industry have made it reasonable for the designers of large scale detector systems to consider the use of custom integrated circuits. A software base including reliable modeling and layout tools has been developed which eases the process of going from block diagrams and schematics to silicon. Fabrication costs have been reduced to the point that designers of systems with as few as 10000 channels may find the total cost of an ASIC less than that of a comparable discrete design. Meanwhile process technologies have been improved to the point that transistors fabricated on monolithic substrates can perform nearly as well as the best available discrete components. The obvious advantages of integrated circuit designs due to the reduction in interconnect capacitance and higher density packaging are being exploited in the development of dense signal processing systems that physically could not be realized with discrete component designs.

Process Selection

The first choice a designer must make is what type of process to use. The readily available processes include CMOS, NMOS and silicon bipolar. Hybrid families are becoming available combining CMOS and bipolar but these are expensive and not yet accessible to small volume users. Our choice was based on an evaluation of the inherent noise performance, f_t, and power requirements of each family. A good guideline[1] is that if the desired signal rise time is greater than 50ns, MOS processes will be preferred over bipolar. This is due mostly to the fact that bipolar transistors require a bias current at the base which contributes a random noise proportional to the measurement time. MOS devices lose their advantage at high frequency where they require large terminal capacitance or large quiescent currents to equal the noise performance of their bipolar counterparts. We quickly decided to focus on bipolar processes for use in the E771 silicon system.

Since advanced bipolar processes require as many as 19 masks, full custom designs entail very large "non-recurring engineering" (NRE) costs. However, several companies now market arrays of uncommitted transistors and passive components. Most of the processing steps can be completed on an unlimited number of wafers using the same masks. The user customizes the array by specifying metal interconnect layers. This dramatically reduces the NRE costs and provides the user with a well characterized parts list that may be interconnected at will. The disadvantage is that the committed placement of components results in designs that have slightly higher stray capacitance and are much less efficient at the utilization of silicon area. The companies which market high performance bipolar linear arrays are AT&T, Gennum, Tektronix, and VTC. After consideration of the properties of the various arrays, and of the CAE & CAD tools available for each one, we decided to use the Tektronix Quickchip 2 for our ASIC's. Tektronix is the only vendor of the four listed that markets its own design tools. The Tektronix arrays are also the only ones which provide the option of customizing laser trimmable nichrome thin film resistors.

Amplifier

The configuration chosen for the E771 silicon strip amplifier is a common emitter transimpedance amplifier. The common base configuration was also considered, but was rejected for the following reason. The noise can be made approximately the same for both configurations if the standing current in the input transistor is the same[2]. However, in the common base configuration the standing current must pass through both a collector resistor and an emitter

* Fermilab is operated by Universities Research Association under a contract with the U.S. Department of Energy.

resistor. and both must have a large value to minimize noise. The result is that the common base configuration dissipates more power and uses more large resistors. which are in relatively short supply on the Quickchip arrays.

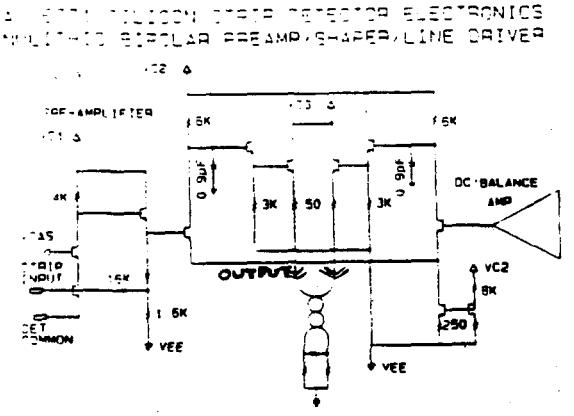


figure 1

A schematic of the E771 amplifier is given in figure 1. The first stage is a common emitter with cascode, buffered by an emitter follower. The open loop voltage gain, $g_m R_c$, is approximately 80. The cascode transistor eliminates Miller capacitance seen at the input, raising the frequency of the open loop dominant pole. This pole position is then determined mainly by the input (detector) capacitance and input impedance. For 20 pf input capacitance, the dominant open loop pole is at about 2 MHz. The cascode transistor introduces a second open loop pole at about 50 MHz. The value of feedback resistance was chosen to give a slight overshoot at the first stage output for an impulse input with an input capacitance of 20 pf. The first stage has a resultant closed loop bandwidth of about 50 MHz, with a 12 db/octave rolloff. The input impedance, given by $R_{rb}/g_m R_c$, should be about 200 ohms. Computer simulation of the impedance, shown in figure 2. predicts a well behaved impedance of approximately 200 ohms, which falls off rapidly above the amplifier bandwidth frequency. The first stage is followed by a second gain stage which drives a differential output. The bandwidth of the second stage is limited with an RC pole at about 25 MHz to shape the pulse and reduce the output noise. The first stage overshoot causes the output of the second stage to return to baseline quickly. Since the response of the input stage is directly dependent on the value of the input capacitance, both stages are optimized for the 20 pf expected for the E771 silicon strip detectors plus fanout. The power dissipation of the full amplifier is less than 35 mw per channel.

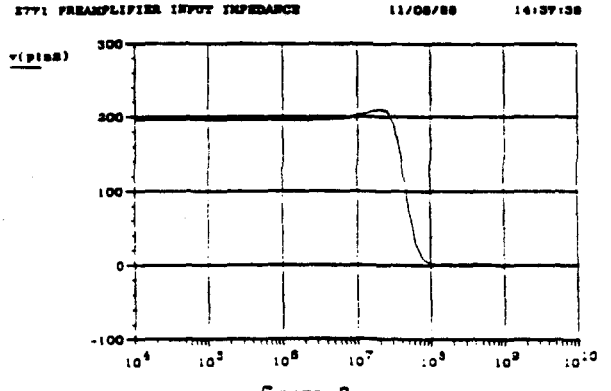


figure 2

In discrete or hybrid designs, key components can be specified with very tight tolerances. However, in integrated circuit design, this is generally not possible. Process variations cause large uncertainties in component values. For example, on a *Quickchip 2* the value of active base implant (AB) resistance can vary by plus or minus 25% from nominal. Also, the temperature coefficient is quite high (1400 ppm/degree C). MOS capacitance can vary by 10% and MIM (Metal Insulator Metal) capacitance can vary by 30%. NPN β can vary by 50%. Most of these variations are uncorrelated with one another. However, these values represent maximums to be expected when comparing wafers processed in separate runs. Within a run the variations wafer to wafer are smaller. On a single wafer the variations are even smaller, and on a single chip the maximum variation is of order 1%. A good integrated circuit design must make use of this fact and rely on matching and differential configurations as much as possible. Laser trimmable nichrome resistors provide the option of specifying exact resistance values and are valuable in some circumstances.

Process variations have a fairly large impact on the first stage of the E771 amplifier. The pulse shape of its output depends on the value of the feedback resistor and of the resistor connected to the collector of the cascode transistor. The use of laser trimmable nichrome resistors was considered as a way of insuring a consistent response, but there are several problems with this approach. In general, the Tektronix nichrome resistors are best suited for use as small value (10-1000 ohm) resistors. A large value resistor must be split into a fixed resistance (with a large uncertainty) and a "trim tab." In order to cover the necessarily large trim range, the trim tab must be physically large. Even if there is enough free area on the chip to place the resistor, the large stray capacitance to the substrate associated with the large trim tab can become a problem. For example, a nichrome resistor trimmable to within 1% of 12K ohms requires a trim tab of about 22000 square microns, which has a stray capacitance of about 0.8 pf. The trimmed resistance of the tab may be very different for chips on different wafers. This means that the RC product associated with the fixed portion of the resistor and with the trim tab may also be different chip to chip. In addition, the parasitic capacitance also can vary significantly, and this is not correlated to the degree of trim required.

An acceptable solution for the amplifier first stage was arrived at which makes use of device matching. Active base resistors were used for both the feedback resistor and the collector resistor. If the collector resistor is lower than nominal, the second pole is higher in frequency. But the feedback resistor is also lower, changing the feedback and maintaining an approximately constant phase margin. The amount of first stage overshoot, and thus the output pulse shape, remains similar. The bandwidth of the amplifier does change and this results in a change in the output pulse width base to base. The overall pulse height can be maintained by trimming the gain of the second stage.

After studying the effects of process and temperature variation, it was evident that a differential second stage was required, with a "dummy" input amplifier to balance the DC output variations of the real input first stage. The dummy stage has no input capacitance and it also has a 6 pf feedback capacitor which kills its frequency response. Consequently, it does not contribute to the output noise. An added advantage of a differential second stage is that a

differential output is naturally provided. The configuration is basically a differential pair with resistive collector loads that are buffered by emitter followers to the outside world. A trimmable nichrome back termination resistor is placed in series with each output. A current mirror is used to bias the pair. The voltage gain of this configuration is relatively insensitive to process variations since it is proportional to the ratio of the collector resistance to the current setting resistance, and these are both the same type. A small nichrome degeneration resistor exists in each leg of the current mirror. These can be used to trim the voltage gain if desired. Shaping is performed by adding capacitance to each collector of the differential pair. An MOS capacitor is used, with a maximum process variation of +-10%, a relatively small effect. Resistance variations do change the shaping time, and this effect tracks the change in the first stage output rather than cancelling it. For example, narrower pulses due to lower than nominal resistance have a smaller shaping time constant applied to them.

Measurements on the Prototype Amplifier

One of the major questions we had before our first experience with a Quickchip design was how good the computer models were. When prototypes were received, the AB resistance was measured (via an on chip test resistor connected between two power pins) and found to be an average of about 11% below nominal. TSPICE* simulations were run at this resistance and the results compared to actual measured results. Figure 3 is the TSPICE predicted amplifier response to a 4fc impulse input for input capacitances of 15 pf, 20 pf, and 25 pf. Figure 4 is the actual measured response of a prototype amplifier with 20 pf. Within measurement accuracy, the simulated and measured responses are identical.

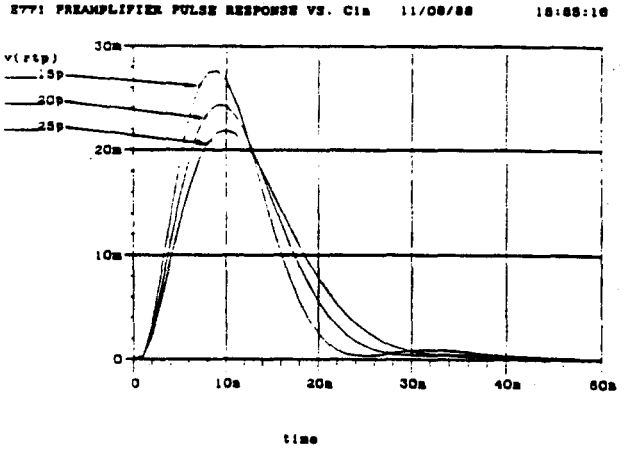


figure 3

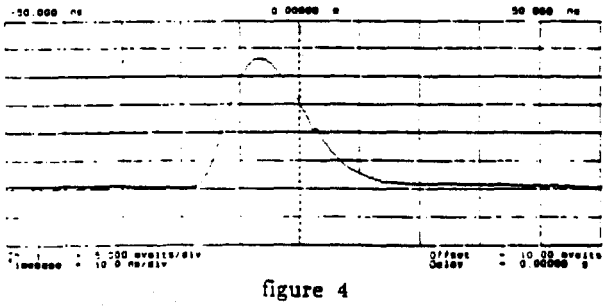


figure 4

* TSPICE is a Tektronix proprietary version of SPICE 2G. We found no significant differences between SPICE and TSPICE results.

The TSPICE predicted noise of the preamp was obtained by integrating the noise density over the full frequency range of the amplifier and then taking the square root. This gives an rms output noise voltage. This can then be referred to the input by dividing by the amplifier gain (mv/fc). The actual noise of the amplifier was measured using the technique shown in figure 5[3], and results are shown in figure 6 along with predicted values. Again, the two cases are almost identical. The noise at 20 pf is .27 fc.

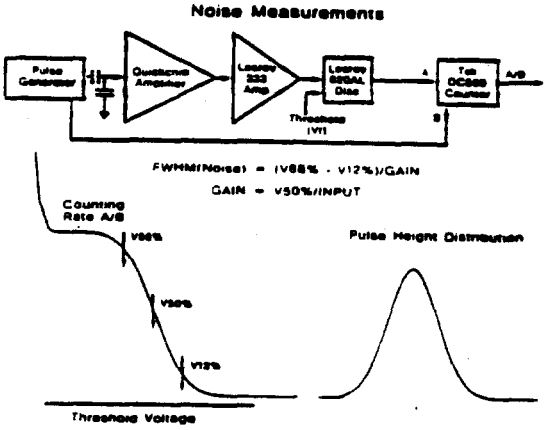


figure 5

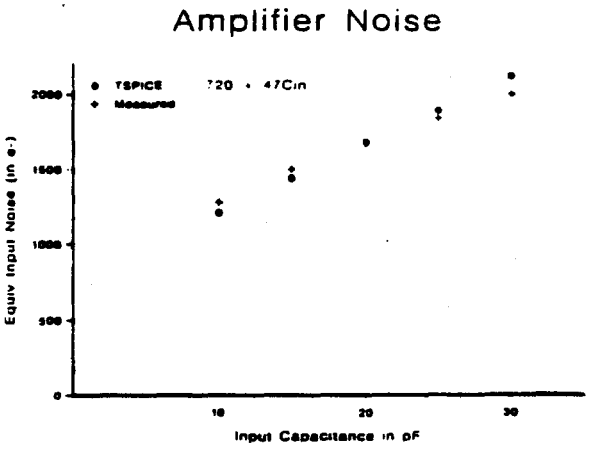


figure 6

Radiation Damage Measurements

Two prototype Quickchip amplifiers were exposed to a Co^{60} source in several steps at a dose rate of about 200 KRad per hour, for a total dose of 5.6 MRad. During irradiation normal bias voltages were applied to the amplifiers. The on chip test resistor (active base type) increased in value at a rate of about 2% per MRad. Nichrome resistors did not change in value. The only observed change in pulse shape was that which would be expected from the increase in the AB resistance. Transistor β decreased about 15%. Base-emitter voltage changes were not observed. Some channels developed a DC offset at the output, but none was large enough to change the pulse response. No significant change in the measured noise was observed.

Discriminator

The shaped preamplifier pulses will be sent over approximately 4 meters of ribbon cable, through a passive termination network, and into a second ASIC. This integrated circuit will provide a latched discriminator output for each strip and also a latched interpolation output derived by feeding the analog sum of two adjacent strips into a separate discriminator with an increased threshold. A fully differential prototype discriminator has been designed and fabricated using a Tektronix Quickchip 2S linear array. The circuit consists of a two stage differential amplifier/comparator with fully balanced hysteresis and voltage programmable threshold. The outputs are CML compatible or suitable for differential logic such as that planned for the latch. The present version requires about 15mw of power per channel (excluding output drivers that were added to the prototype) between supply rails of plus and minus 3 volts.

Circuit Description (see figure 7)

The input stage serves to provide signal gain and positive feedback nodes for threshold and hysteresis. This is accomplished using a cascoded differential input stage consisting of Q1-Q4. The approximate voltage gain of this stage is

$$A=IR_c/2u_t \quad (u_t=kt/q;\ I=\text{current in Q5})$$

The emitters of Q3 and Q4 have low input impedance and provide good feedback summing points without significantly reducing the bandwidth of the first stage. Using the expression for the gain given above, the effective input voltage due to a difference I_d in the currents in the collector resistors of Q3 and Q4 may be written as:

$$V_i=2I_du_t/I$$

The operation of both the threshold and the threshold hysteresis can be understood using this relationship. The output of the first stage is buffered with emitter followers and fed into the comparator stage formed by Q6 and Q7. The collector resistors in this stage are split into equal values and buffered with two sets of emitter followers. Q8 and Q9 provide a voltage divided output suitable for CML or differential logic stages. Q10 and Q11 feed a passive voltage translator to provide the hysteresis switch formed by Q12 and Q13 with suitable logic levels.

The function of the hysteresis is to reduce the effective input threshold during the time between when the input goes over threshold and the time when it falls below the lowered threshold value. This helps to reduce the chance of repetitive triggering due to noise in the signal and has the beneficial effect of providing increased gain near the comparator switching point. Hysteresis has been implemented using a two step approach. A fast message that the output has changed state is sent to the input in the form of a charge. This is accomplished by the capacitor between the emitters of Q4 and Q10 and between Q3 and Q11. Switching logic levels across the capacitance C inserts charge into the summing node. This is almost immediately transferred to the collectors of Q3 and Q4 causing a voltage change proportional to the stray capacitance at that node. The only drawback to this form of feedback is the fact that the MIM capacitors used are subject to +-30% process variations. This fast part of the hysteresis feedback is drained away by the resistors in the collectors of Q3 and Q4 in a few nanoseconds. Long term hysteresis is provided by switching a current controlled by Q12 and Q13. Since only 20μa of current in the collectors of Q3 and Q4 is required to shift the apparent input offset by 3mv, an unusually small current source needs to be established. This has been accomplished using a normal current source transistor with the normal small valued P+ type emitter resistor replaced by a large valued AB resistor. This can lead to a 50% variation in effective hysteresis if AB and P+ resistor types go to opposite three sigma limits. SPICE simulation indicates that worst case variations in this current change the effective input threshold by 1.5 mv.

The threshold is set by adjusting the voltage difference between the base of Q14, referenced to ground and Q15, which is attached to a voltage divider network. The emitters of Q14 and Q15 connect to a common current source through equal valued resistors. A drop across these resistors created by the threshold voltage steers current from the emitter of Q14 to Q15. This differential current is referred to the input summing node through the collectors of Q14 and Q15. The effective input threshold may be approximated as:

$$THR=.67u_t(V_{th}/V_{ca})$$

where V_{th} is the input threshold setting and V_{ca} is the voltage across the current setting resistor. It should be noted that that there are no process dependent terms to first order. On the prototype discriminator, a one volt change in applied threshold setting changes the effective threshold by about 16 mv.

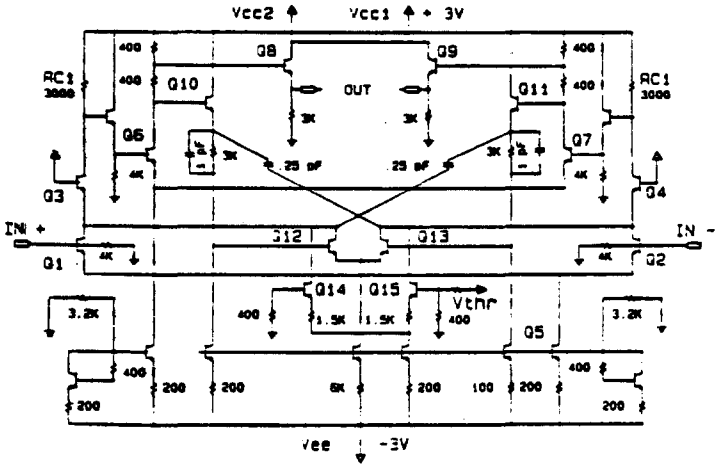


figure 7

Measurements on the Prototype Discriminator

The output of a prototype Quickchip amplifier was connected through eight meters of cable to the input of a prototype discriminator. Figure 8 shows a typical discriminator output pulse for a near threshold input pulse from the amplifier. We have set thresholds as low as eight millivolts and seen only random firing due amplifier noise. There is no hint of retriggering indicating that the hysteresis performs well. Figure 9 shows a snapshot in time taken of the discriminator output firing on amplifier noise at a threshold of about eight millivolts.

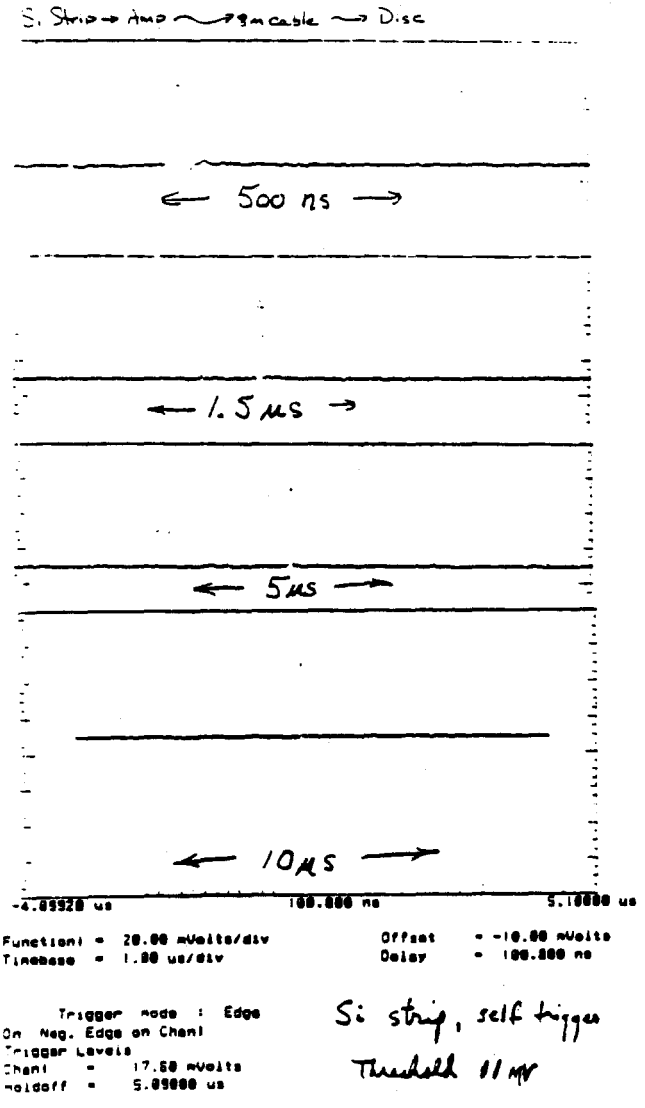


figure 8

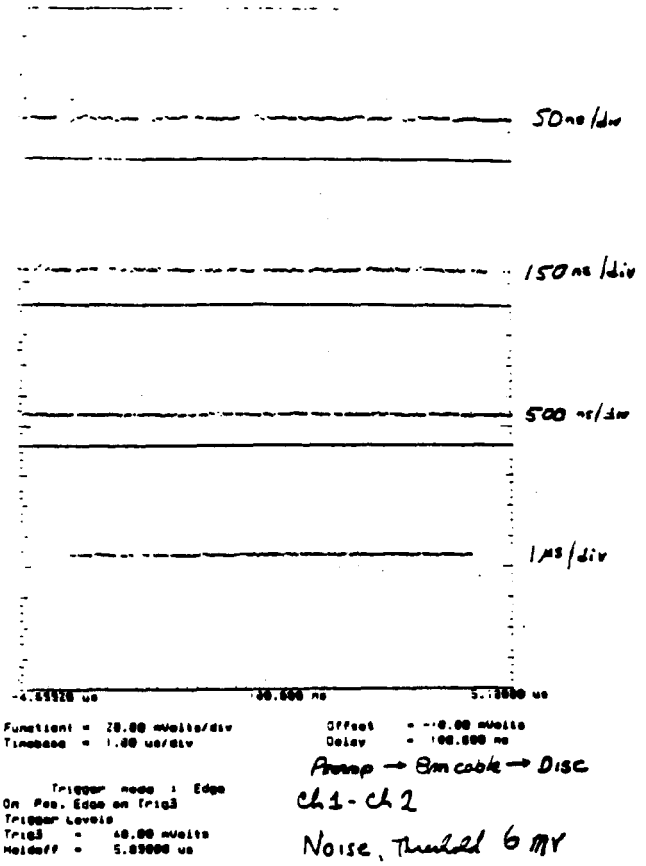


figure 9

Conclusion

We have designed and prototyped two bipolar ASIC's using Tektronix Quickchip 2 linear arrays. Measurements of the performance of the prototype devices agree very well with SPICE simulations.

References

[1] V. Radeka, Annual Review of Nuclear and Particle Science, 1988
[2] P. D'Angelo et al., NIM 193 (1982) 533-538
[3] This method is described by P. Jarron and M. Goyot, NIM 226 (1984) 156-162

Acknowledgements

We would like to thank Jim McCreary of the Argonne National Laboratory Chemistry Department for permission to use the Cobolt source at Argonne to perform our radiation damage studies.

Postamp/Comparator

Hardware Description

HN100

David Christian, Merle Haldeman
Scott Holm & Bruce Merkel

Version 3.0

Revised 5-8-91

Word 4.0

TABLE OF CONTENTS

LIST OF FIGURES

1.0 GENERAL INFORMATION

1.1 SSD READOUT SYSTEM (Overview)

The **Postamp/Comparator (P/C)** module was designed to work in conjunction with two other modules in the front end electronics for experiment E-771; a fixed target experiment. The two other modules; the **Sequencer (Seq)** and the **Delay Encoder (DE)** operate together with the **P/C** in a synchronous manner taking information from preamplifiers, mounted within a few inches of the **Silicon Strip Detectors (SSD)**, repackaging it and sending it on for further repackaging and analysis.

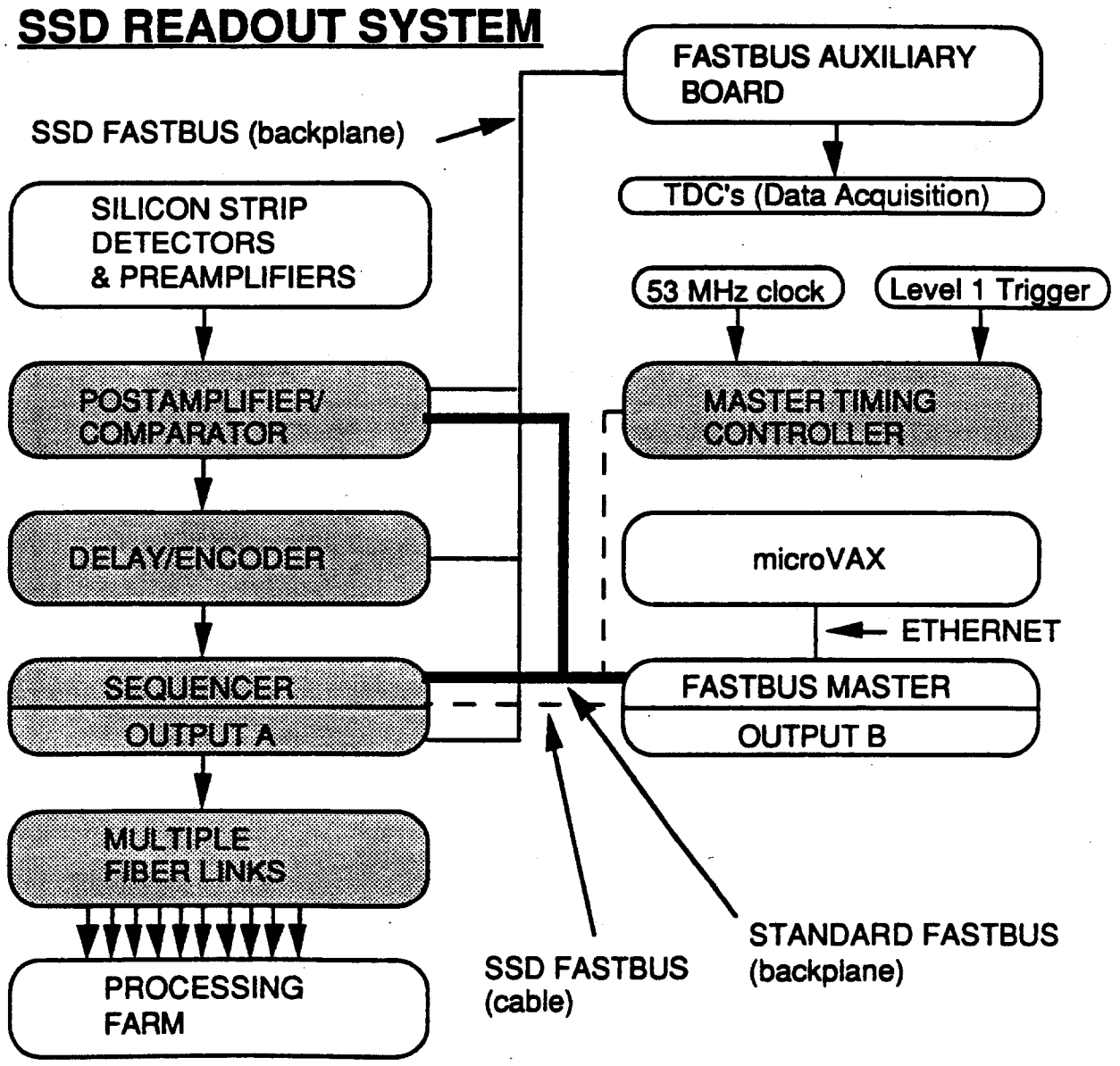


Figure 1: SSD READOUT SYSTEM

The purpose of the **P/C** module is to compare the voltage amplitude of it's input pulses, to a programmable reference voltage (the threshold voltage) and to latch an ECL logic level output for each input pulse greater than it's reference voltage. The latching of the logic levels is synchronized with a 53 MHz clock provided by the **Seq** module.

Differential input signals are received by each P/C from 128 SSD preamplifiers through four 64-conductor ribbon cable connectors mounted on the front portion of the P/C module. 128 ECL level output signals are sent, via a custom FASTBUS auxiliary backplane, to the DE module located in an adjacent slot in the FASTBUS crate. Eight fast analog sums and eight analog encoded digital sums intended for use by the prompt trigger logic, are constructed and output through FASTBUS auxiliary cards.

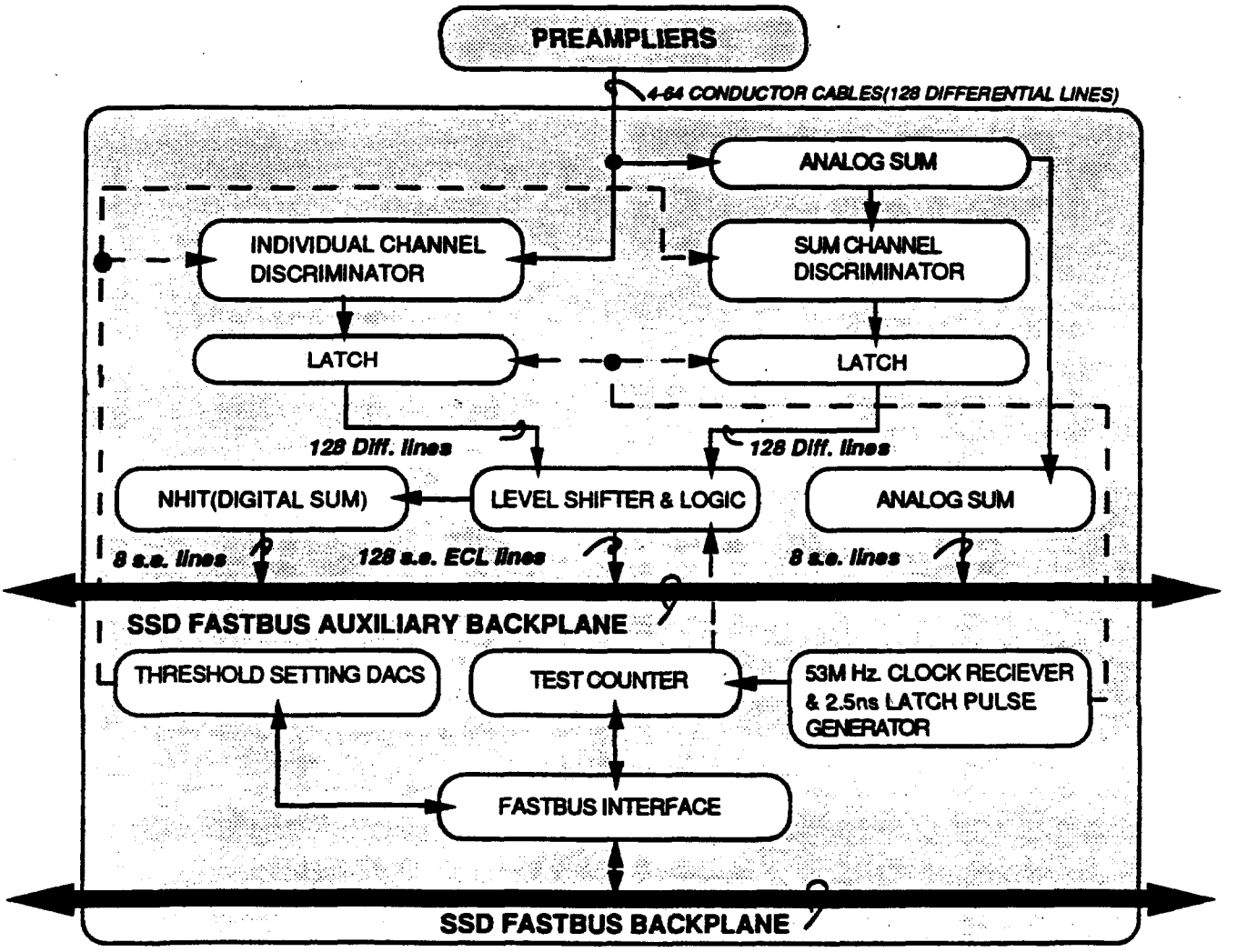


Figure 2: SSD POSTAMP/COMPARATOR

1.1.1 Standard Bus System

The P/C module is a FASTBUS slave module designed to be used together with a DE module in a FASTBUS crate equipped with a special SSD Readout auxiliary backplane. A diagram of an SSD crate with top, bottom and sides removed is shown below.

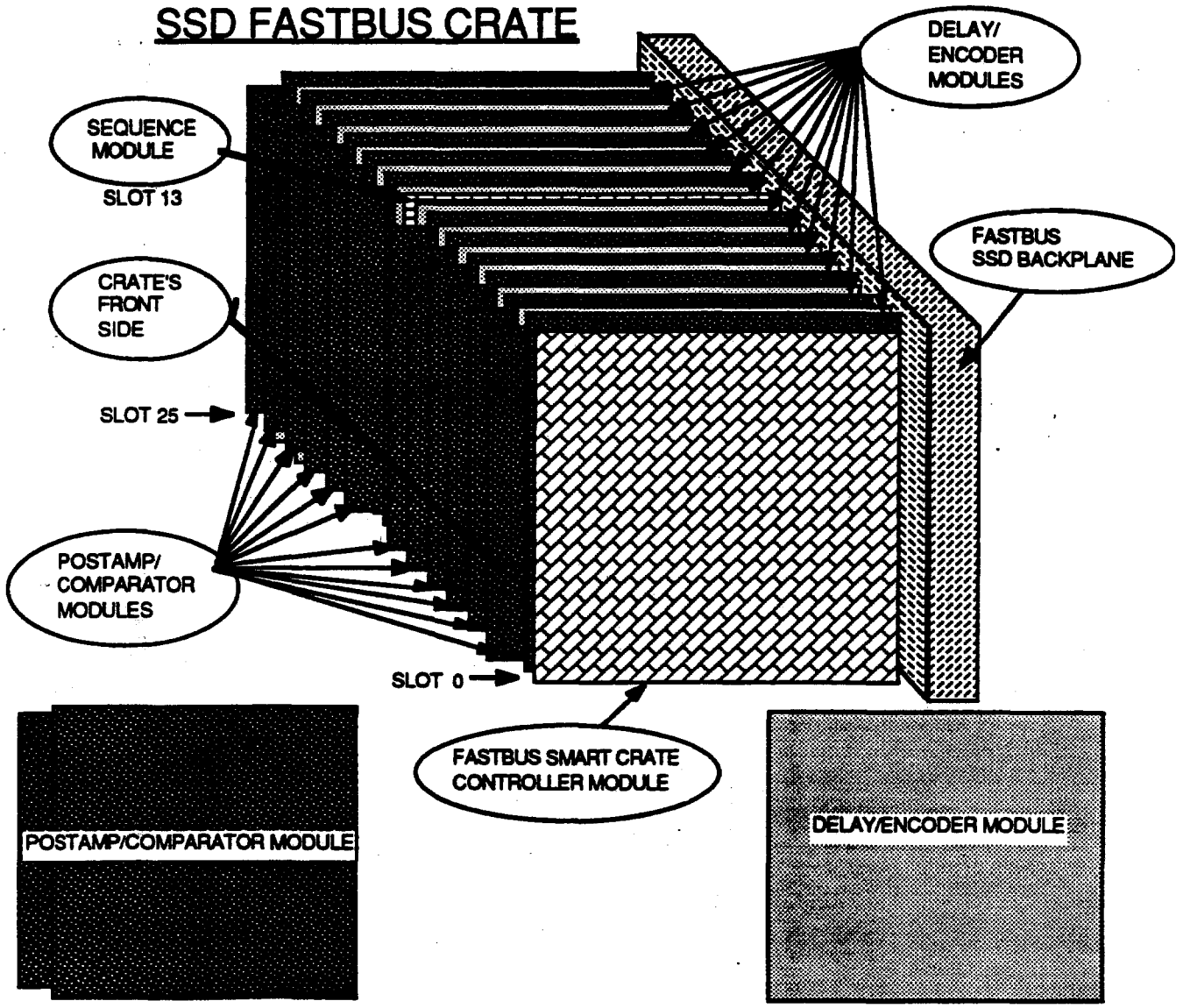


Figure 3: SSD FASTBUS CRATE

2.0 P/C MODULE - GENERAL INFORMATION

2.1 APPLICATION

This module was designed for use in the fast Silicon Strip Detector (SSD) readout system developed at FNAL for use by E-771. It is directly usable only in the context of this readout system.

2.2 PACKAGING

This board is a standard single width FASTBUS module(see ANSI/IEEE Std 960-1986) with the exception that the circuit board is extended through the front panel 3-7/8" and the FASTBUS segment connector uses pins B02 and B03 for -3.5V and pins B04, B05, and B40 for +3.5V.

2.2.1 Physical Size

The maximum board dimensions are 14.436 inches high by 19.747 inches deep, typical board dimensions are illustrated below. The board thickness is between 0.086 inches and 0.100 inches.

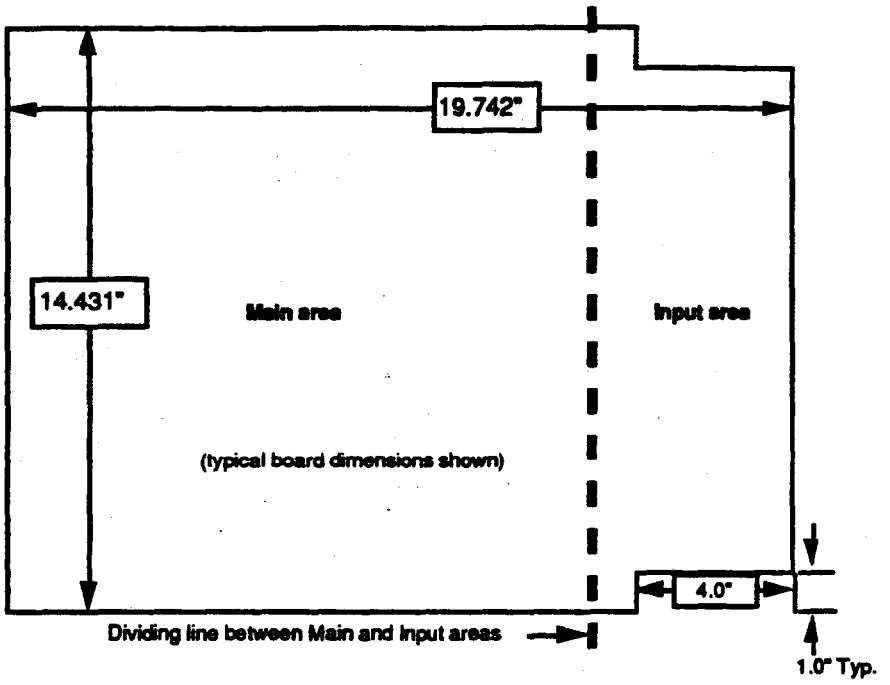


Figure 4: P/C Board Size

2.2.2 Layup

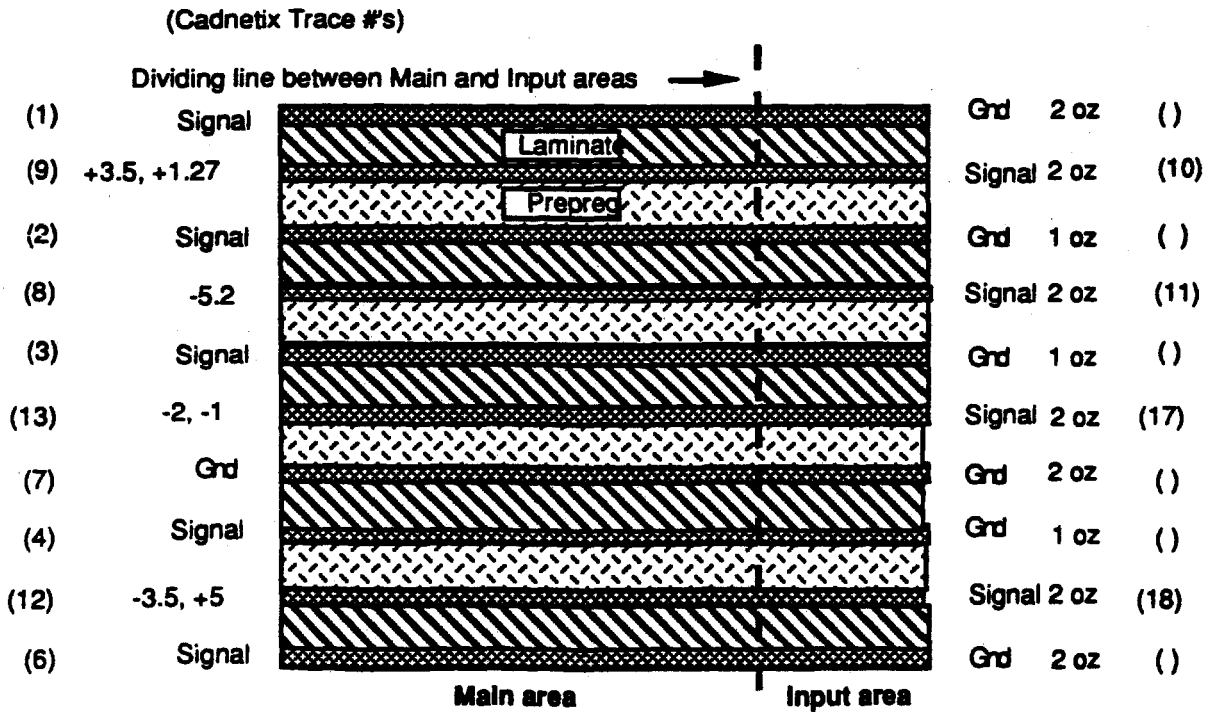


Figure 5: P/C Board Layup

2.2.3 Layout

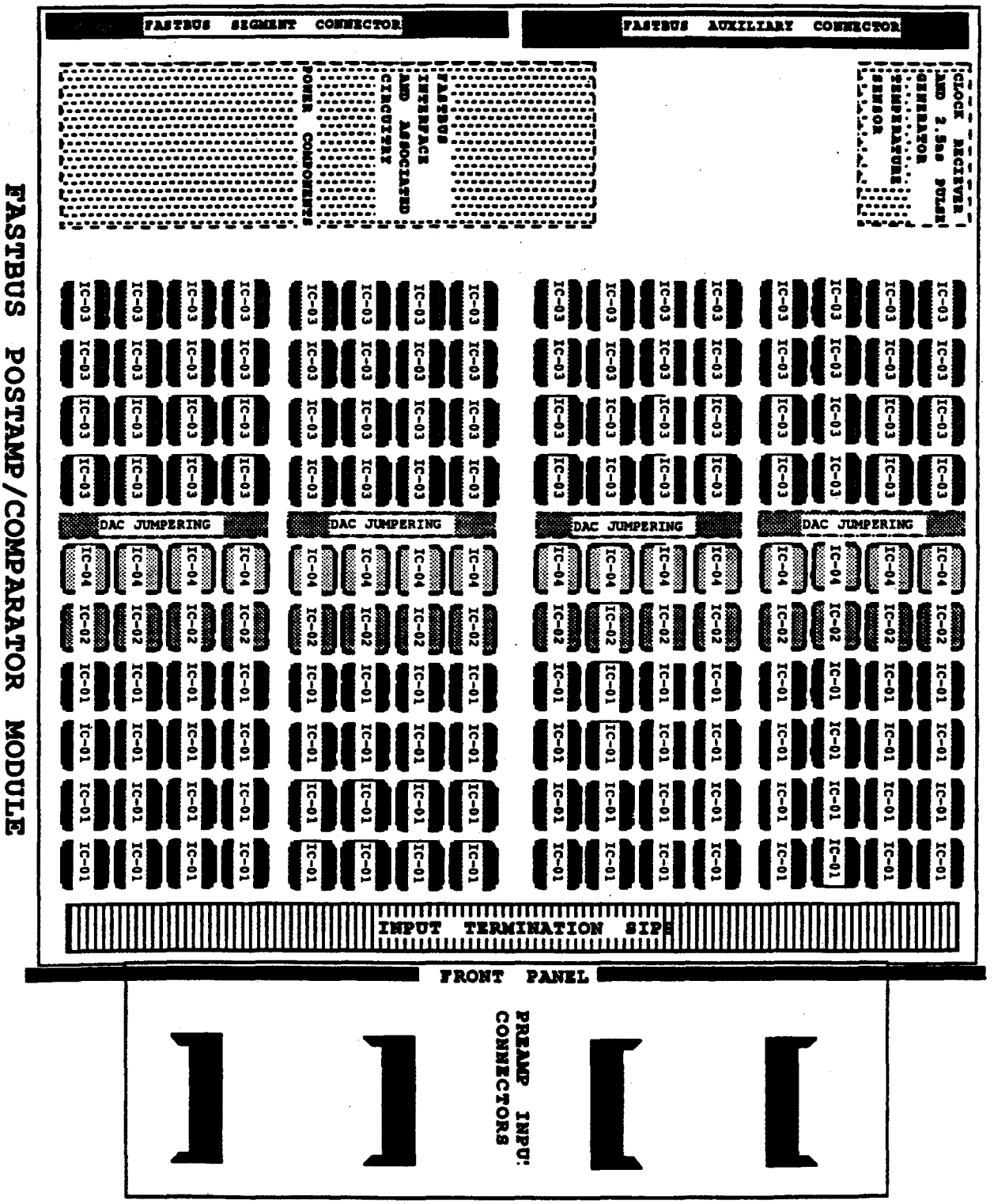
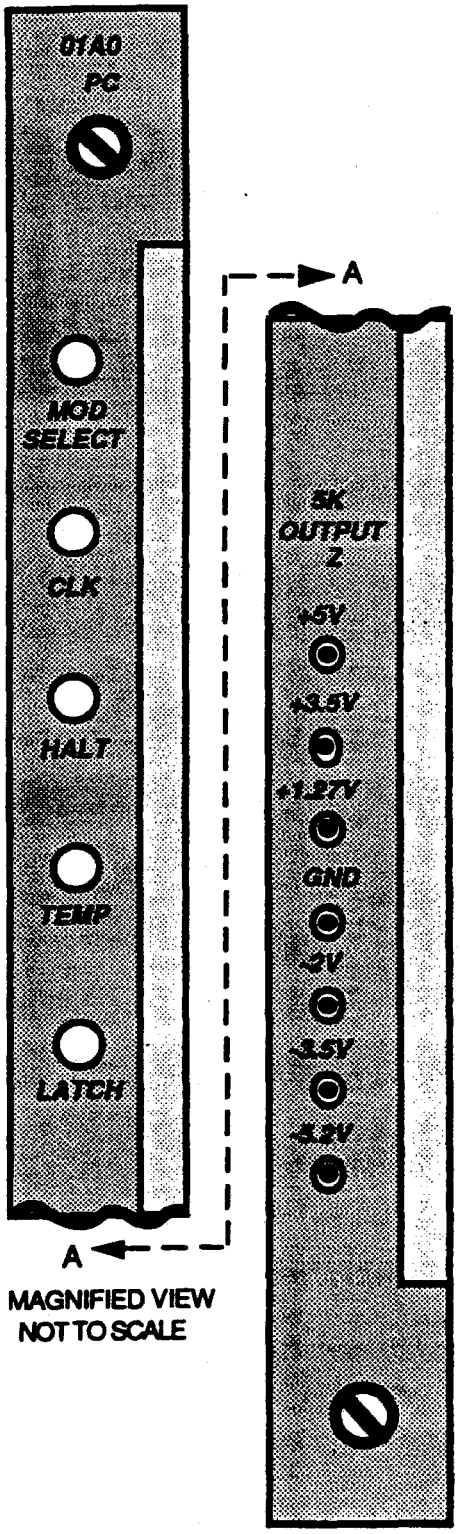


Figure 6: P/C Board Layout

2.2.4 Front Panel



LED COLOR AND ENERGIZED MEANING

MOD SELECT	Yellow	Module is being accessed by FASTBUS
CLK	Yellow	CLOCK (53M Hz.) is present
HALT	Red	Module is in TEST mode, Latches are in transperent mode.
TEMP	Red	Module temperature is >50 degrees celcius at the module top.
LATCH	Yellow	Module is in RUN mode, Latches are in latch mode.

VOLTAGE TEST POINTS

Voltage test points are in series with 5.1k ohm resistor.

Refer to Appendix G - Module Drawings

#2563.000-ED-215752 FRONT PANEL & FRONT PANEL AIR GAP DETAILS
#2563.000-ED-215753 FRONT PANEL SILKSCREEN

Figure 7: P/C Module Front Panel

2.2.5 Component List

#	MANUFACTURER	PART NUMBER	PART NAME	PART DESCRIPTION	QNTY PER BD
1	MOTOROLA	MC10109P	IC	DUAL 4-5-INPUT OR/NOR GATE	2
2	MOTOROLA	MC10124P	IC	QUAD TTL TO ECL CONVERTER	2
3	MOTOROLA	MC10125P	IC	QUAD ECL TO TTL CONVERTER	5
4	MOTOROLA	CD74HC4514EN	IC	4 BIT LATCHED/4 TO 16 LINE DECODER	4
5	BROOKTREE	BT501KC	IC	ECL/TTL OCTAL TRANSCIEVER/TRANSLATOR	2
6	T.I.	NE555D	IC	555 TIMER SURFACE MOUNT	1
7	MOTOROLA	MC10E016FN	IC	8-BIT SYNC BINARY UP COUNTER	1
8	MOTOROLA	MC10E101FN	IC	QUAD 4-INPUT OR/NOR GATE	2
9	MOTOROLA	MC10H115FN	IC	QUAD LINE RECEIVER	1
10	MOTOROLA	MC10H188FN	IC	HEX BUFFER WITH ENABLE	1
11	T.I.	SN74123	IC	RETRIGGERABLE MULTIVIBRATOR	1
12	T.I.	SN10KHT5543FN	IC	OCTAL TTL-ECL TRANSLATOR W/OE	3
13	T.I.	SN10KHT5541FN	IC	OCTAL ECL-TTL TRANSLATOR 3 STATE OUT	1
14	ALTERA	EPM5128JC	IC	128 MACROCELL MAX EPLD	1
15	FERMI	IC 01	IC	"SSD 2/CHAN SUM,DISCRIM AND LATCH"	64
16	FERMI	IC 02	IC	"3/CHAN LOGIC,QUAD ANALOG SUM/CLK"	16
17	US2	IC 03 DAC	IC	IC-3 CMOS DAC/ADC	64
18	FERMI	IC 04	IC	SSD LOGIC AND NHIT	16
19	TI	74ACT11245DW	IC	OCTAL BUS TRANSCEIVER 3/STATE OUTPUT	2
20	TI	SN7404	IC	HEX INVERTER	1
21	NATIONAL	LM311M	IC	LM311M SMT VOLTAGE COMPARATOR	1
22	NATIONAL	LM35CZ	IC	LM35CZ TEMPERATURE SENSOR DEGREE C	1
23	3M	268-6234-71-3877	IC	SOCKET 68 PIN FOR EPM5128JC	1
24	LINEAR TECH	LT317AK	REG	POSITIVE VOLTAGE REGULATOR	1
25	ELMEC TECH.	FDC 2510	DELAY	SIP ULTRA HIGH-SPEED FIXED DELAY LINE	1
26	LITTELFUSE	251007	FUSE	FUSE PICO 7A	5
27	GENERAL SEMI	5KP5.0A	DIODE	TRANSIENT SUPPRESSION TRANSZORB	5
28	MOTOROLA	1N5401	DIODE	1N5401 DIODE	2
29	MOTOROLA	1N4002	DIODE	1N4002 DIODE	1
30	BEAU	85802	CONN	EUROSTYLE TERMINAL STRIPS STYLE B 90'	2
31	DU PONT	68002-236	CONN	BERGSTICK II HEADERS (36 PINS) @ PINS	265
32	E.F.JOHNSON	105-1041-001	CONN	TIP JACKS WHITE	3
33	E.F.JOHNSON	105-1042-001	CONN	TIP JACKS RED	3
34	E.F.JOHNSON	105-1043-001	CONN	TIP JACKS BLACK	1
35	YAMAICHI	NFP-64A-0122	CONN	BOX HEADER 64 PIN 25mil PITCH	4
36	AMP	2-532956-0	CONN	FASTBUS MODULE SEGMENT CONNECTOR	1
37	AMP	534974-9	CONN	FASTBUS MODULE AUX CONNECTOR 195 PIN	1
37A	SAMTEC	SNT-100-BK-G	CONN	2 PIN SAMTEC SHUNT	3
38	BRADFORD	HC0ZXX	SIP	50 OHM/.01uF CAP TERMINATING SIP	64
39	MURATA/ERIE	GRM42-6X7R103K050BB	CAP	.01uF CAPACITOR SURFACE MOUNT1206	1
40	MURATA/ERIE	GRM42-6X7R104K050B	CAP	.1uF CAPACITOR SURFACE MOUNT 1206	447
41	MEPCO/CENT	49MC106C010KOASFT	CAP	10uF SMT POLARIZED CAP - CASE STYLE C	2
42	MURATA/ERIE	GRM42-2Y5V105Z016B	CAP	1uF CHIP CAP 16 WVDC S.M. 1210	34
43	MURATA/ERIE	GRM42-6X7R471K050B	CAP	470 pF 1206 SMT CAP	1
44	MURATA/ERIE	GRM42-6C0G500K050B	CAP	50 pF 1206 SMT CAP	1
45	RCD	MC 1A 100Ω 5% B	RES	RESISTOR 100 OHM 1206 SMT	117
46		5.1K Ω 5% 1/4 WATT	RES	RESISTOR 5.1K OHM	7
47	RCD	MC 1A 1K Ω 5% B	RES	RESISTOR 1K OHM 1206 SMT	5
48	RCD	MC 1A 220 Ω 5% B	RES	RESISTOR 220 OHM 1206 SMT	6
49	RCD	MCR 1A 24.9K Ω 1% B	RES	RES 25K OHM 1206 SMT	32
50	RCD	MC 1A 51 Ω 5% B	RES	RES 50 OHM 1206 SMT	12
51	RCD	MC 1A 150 Ω 5% B	RES	RES 150 OHM 1206 SMT	32
52	OHMITE RES	1 Ω 5% 45J1R0	RES	1 OHM 5 WATT	2
53	RCD	MC 1A 330 Ω 5% B	RES	330 OHM 1206 SMT	34
54	RCD	MC 1A 1.5K Ω 5% B	RES	1.5K OHM 1206 SMT	1
55	RCD	MC 1A 3K Ω 5% B	RES	3K OHM 1206 SMT	1
56	RCD	MC 1A 6.8K Ω 5% B	RES	6.8K OHM 1206 SMT	1
57	RCD	MC 1A 13K Ω 5% B	RES	13K OHM 1206 SMT	1
58	RCD	MC 1A 47K Ω 5% B	RES	47K OHM 1206 SMT	1
59		120 Ω 5% 1/4 WATT	RES	120 OHM 1/4 WATT	1
60	RCD	MC 1A 2.2 MEG Ω 5% B	RES	2.2 MEG OHM 1206 SMT	1
61	RCD	MC 1A 4.3K Ω 5% B	RES	4.3K OHM 1206 SMT	1
62	RCD	MC 1A 200 Ω 5% B	RES	200 OHM 1206 SMT	1
63	MEPCO/CENT	ST4MA 501	RES	500 OHM POT SMT	1
64	RCD	MC 1A 10K Ω 5% B	RES	10K OHM 1206 SMT	2

#	MANUFACTURER	PART NUMBER	PART NAME	PART DESCRIPTION	ONTY PER BD
65	RCD	MC 1A 43 Ω 5% B	RES	43 OHM 1206 SMT	2
66	FERMI		BOARD	POST/AMP COMPARATOR BOARD	1
67	FERMI		BOARD	P/C CIRCUIT BOARD STUFFING	1
68	FERMI	0882-MB-199070	HRDWRE	"RAF 3/8"" HEX SPACER"	2
69	RAF	7039.SS MOD E =1/2	SCREW	SHOULDER SCREW	2
70	DIALITE	559-2301-001	LED	YELLOW LED	3
71	DIALITE	559-2101-001	LED	RED LED	2
72	FERMI	2563.000-MD-215752	PANEL	FRONT PANEL	1
73	FERMI	2563.000-MD-215752	PANEL	FRONT PANEL FILLER	1
74			SCREW	"4-40 x 5/16"" FLAT HEAD SCREW"	4
75			SCREW	" 2-56 x 3/8"" SS FLAT HEAD"	1
76	SECME	27 11008 23	SWITCH	TROPICAL VERSION ROTARY 8 POS. SWITCH	1
77	BELDEN CABLE	#9R28010	WIRE	28 GA. 10 CONDUCTOR RIBBON CABLE (IN)	27
78	OK IND		WIRE	PRE STRIPPED WIRE WRAP WIRE (IN)	240
79	DU PONT	68002-536	CONN	BERGSTIKII HEADERS STRAIGHT SINGLE ROW	96
80			SCREW	"6-32 x 3/8"" BINDER HEAD"	2
81			NUT	6-32 NUT	2
				TOTAL	1348

2.2.6 Connector Pinouts

2.2.6.1 FASTBUS Segment Connector

(Viewed From Front of Crate)

B01-GND	A01-GND
B02- -3.5V BUS	A02-AL00
B03- -3.5V BUS	A03-AL01
B04- +3.5V BUS	A04-AL02
B05- +3.5V BUS	A05-GND
B06- N/C	A06-AL03
B07- -5.2V BUS	A07-AL04
B08- -5.2V BUS	A08-AL05
B09- -5.2V BUS	A09-AR
B10-AK	A10-GND
B11-AI	A11-GK
B12-SS0	A12-DK
B13- -2V BUS	A13-AK
B14- +5V BUS	A14-WT
B15- +5V BUS	A15-GND
B16-SS1	A16-AS
B17-SS2	A17-DS
B18-RD	A18-MS0
B19-MS2	A19-MS1
B20-B20R	A20-GND
B21-EG	A21-AD00
B22- +5V BUS	A22-AD01
B23-SR	A23-AD02
B24-RB	A24-AD03
B25-BH	A25-GND
B26-B26R	A26-AD04
B27-GA00	A27-AD05
B28-GA01	A28-AD06
B29-GA02	A29-AD07
B30-GA03	A30-GND
B31-GA04	A31-AD08
B32- -2V BUS	A32-AD09
B33-DLA	A33-AD10
B34-DRA	A34-AD11
B35-DLB	A35-GND
B36-DRB	A36-AD12
B37-DAR	A37-AD13
B38-DBR	A38-AD14
B39-B39R	A39-AD15
B40- +3.5V BUS	A40-GND
B41-FP1	A41-TP
B42- -5.2V BUS	A42-A42R

(continuation)

B43-FP2	A43-PE
B44-FP3	A44-PA
B45-B45R	A45-GND
B46-TR0	A46-AD16
B47-TR1	A47-AD17
B48-TR2	A48-AD18
B49-TR3	A49-AD19
B50-TR4	A50-GND
B51-TR5	A51-AD20
B52- +5V BUS	A52-AD21
B53-TR6	A53-AD22
B54-TR7	A54-AD23
B55-UR0	A55-GND
B56-UR1	A56-AD24
B57-TX	A57-AD25
B58-RX	A58-AD26
B59- -5.2V BUS	A59-AD27
B60- -5.2V BUS	A60-GND
B61- -5.2V BUS	A61-AD28
B62- -2V BUS	A62-AD29
B63- +5V BUS	A63-AD30
B64- +5V BUS	A64-AD31
B65-GND	A65-GND

Notes:
1.-5.2 Volts uses 7 pins
2.+5.0 Volts uses 6 pins
3.-2.0 Volts uses 3 pins
4.+3.5 Volts uses 3 pins
5.-3.5 Volts uses 2 pins
6.Ground uses 14 pins

2.2.6.2 FASTBUS Auxiliary Connector

(Viewed From Front of Crate)

C	B	A
C01-H53MHZ,Ø1 Clock	B01-Post./Comp. Ch.01	A01-Post./Comp. Ch.00
C02-GND	B02-Post./Comp. Ch.03	A02-Post./Comp. Ch.02
C03-L53MHZ,Ø1 Clock	B03-Post./Comp. Ch.05	A03-Post./Comp. Ch.04
C04-GND	B04-Post./Comp. Ch.07	A04-Post./Comp. Ch.06
C05-GND	B05-Post./Comp. Ch.09	A05-Post./Comp. Ch.08
C06-GND	B06-Post./Comp. Ch.11	A06-Post./Comp. Ch.10
C07-GND	B07-Post./Comp. Ch.13	A07-Post./Comp. Ch.12
C08-Reset	B08-Post./Comp. Ch.15	A08-Post./Comp. Ch.14
C09-GND	B09-Post./Comp. Ch.17	A09-Post./Comp. Ch.16
C10-GND	B10-Post./Comp. Ch.19	A10-Post./Comp. Ch.18
C11-GND	B11-Post./Comp. Ch.21	A11-Post./Comp. Ch.20
C12-Analog Sum 0*	B12-Post./Comp. Ch.23	A12-Post./Comp. Ch.22
C13-Analog Sum 0	B13-Post./Comp. Ch.25	A13-Post./Comp. Ch.24
C14-Digital Sum 0*	B14-Post./Comp. Ch.27	A14-Post./Comp. Ch.26
C15-Digital Sum 0	B15-Post./Comp. Ch.29	A15-Post./Comp. Ch.28
C16-Analog Sum 1*	B16-Post./Comp. Ch.31	A16-Post./Comp. Ch.30
C17-Analog Sum 1	B17-Post./Comp. Ch.33	A17-Post./Comp. Ch.32
C18-Digital Sum 1*	B18-Post./Comp. Ch.35	A18-Post./Comp. Ch.34
C19-Digital Sum 1	B19-Post./Comp. Ch.37	A19-Post./Comp. Ch.36
C20-GND	B20-Post./Comp. Ch.39	A20-Post./Comp. Ch.38
C21-Analog Sum 2*	B21-Post./Comp. Ch.41	A21-Post./Comp. Ch.40
C22-Analog Sum 2	B22-Post./Comp. Ch.43	A22-Post./Comp. Ch.42
C23-Digital Sum 2*	B23-Post./Comp. Ch.45	A23-Post./Comp. Ch.44
C24-Digital Sum 2	B24-Post./Comp. Ch.47	A24-Post./Comp. Ch.46
C25-Analog Sum 3*	B25-Post./Comp. Ch.49	A25-Post./Comp. Ch.48
C26-Analog Sum 3	B26-Post./Comp. Ch.51	A26-Post./Comp. Ch.50
C27-Digital Sum 3*	B27-Post./Comp. Ch.53	A27-Post./Comp. Ch.52
C28-Digital Sum 3	B28-Post./Comp. Ch.55	A28-Post./Comp. Ch.54
C29-GND	B29-Post./Comp. Ch.57	A29-Post./Comp. Ch.56
C30-GND	B30-Post./Comp. Ch.59	A30-Post./Comp. Ch.58
C31-Analog Sum 4*	B31-Post./Comp. Ch.61	A31-Post./Comp. Ch.60
C32-Analog Sum 4	B32-Post./Comp. Ch.63	A32-Post./Comp. Ch.62
C33-Digital Sum 4*	B33-Post./Comp. Ch.65	A33-Post./Comp. Ch.64
C34-Digital Sum 4	B34-Post./Comp. Ch.67	A34-Post./Comp. Ch.66
C35-Analog Sum 5*	B35-Post./Comp. Ch.69	A35-Post./Comp. Ch.68
C36-Analog Sum 5	B36-Post./Comp. Ch.71	A36-Post./Comp. Ch.70
C37-Digital Sum 5*	B37-Post./Comp. Ch.73	A37-Post./Comp. Ch.72
C38-Digital Sum 5	B38-Post./Comp. Ch.75	A38-Post./Comp. Ch.74
C39-GND	B39-Post./Comp. Ch.77	A39-Post./Comp. Ch.76
C40-Analog Sum 6*	B40-Post./Comp. Ch.79	A40-Post./Comp. Ch.78
C41-Analog Sum 6	B41-Post./Comp. Ch.81	A41-Post./Comp. Ch.80
C42-Digital Sum 6*	B42-Post./Comp. Ch.83	A42-Post./Comp. Ch.82
C43-Digital Sum 6	B43-Post./Comp. Ch.85	A43-Post./Comp. Ch.84
C44-Analog Sum 7*	B44-Post./Comp. Ch.87	A44-Post./Comp. Ch.86
C45-Analog Sum 7	B45-Post./Comp. Ch.89	A45-Post./Comp. Ch.88
C46-Digital Sum 7*	B46-Post./Comp. Ch.91	A46-Post./Comp. Ch.90
C47-Digital Sum 7	B47-Post./Comp. Ch.93	A47-Post./Comp. Ch.92
C48-GND	B48-Post./Comp. Ch.95	A48-Post./Comp. Ch.94
C49-GND	B49-Post./Comp. Ch.97	A49-Post./Comp. Ch.96
C50-GND	B50-Post./Comp. Ch.99	A50-Post./Comp. Ch.98
C51-GND	B51-Post./Comp. Ch.101	A51-Post./Comp. Ch.100
C52-GND	B52-Post./Comp. Ch.103	A52-Post./Comp. Ch.102
C53-GND	B53-Post./Comp. Ch.105	A53-Post./Comp. Ch.104
C54-GND	B54-Post./Comp. Ch.107	A54-Post./Comp. Ch.106
C55-GND	B55-Post./Comp. Ch.109	A55-Post./Comp. Ch.108
C56-GND	B56-Post./Comp. Ch.111	A56-Post./Comp. Ch.110
C57-GND	B57-Post./Comp. Ch.113	A57-Post./Comp. Ch.112
C58-GND	B58-Post./Comp. Ch.115	A58-Post./Comp. Ch.114
C59-GND	B59-Post./Comp. Ch.117	A59-Post./Comp. Ch.116
C60-GND	B60-Post./Comp. Ch.119	A60-Post./Comp. Ch.118
C61-GND	B61-Post./Comp. Ch.121	A61-Post./Comp. Ch.120
C62-GND	B62-Post./Comp. Ch.123	A62-Post./Comp. Ch.122
C63-GND	B63-Post./Comp. Ch.125	A63-Post./Comp. Ch.124
C64-N/C	B64-Post./Comp. Ch.127	A64-Post./Comp. Ch.126
C65-N/C	B65-GND	A65-N/C

2.2.6.3 Input Connector Pin Definitions

J1	J2	J3	J4	PIN #
CH0*	CH2*	CH125*	CH127*	1
CH0	CH2	CH125	CH127	2
CH4*	CH6*	CH121*	CH123*	3
CH4	CH6	CH121	CH123	4
CH8	CH10	CH117	CH119	5
CH8*	CH10*	CH117*	CH119*	6
CH12	CH14	CH113	CH115	7
CH12*	CH14*	CH113*	CH115*	8
CH16*	CH18*	CH109*	CH111*	9
CH16	CH18	CH109	CH111	10
CH20*	CH22*	CH105*	CH107*	11
CH20	CH22	CH105	CH107	12
CH24	CH26	CH101	CH103	13
CH24*	CH26*	CH101*	CH103*	14
CH28	CH30	CH97	CH99	15
CH28*	CH30*	CH97*	CH99*	16
CH32*	CH34*	CH93*	CH95*	17
CH32	CH34	CH93	CH95	18
CH36*	CH38*	CH89*	CH91*	19
CH36	CH38	CH89	CH91	20
CH40	CH42	CH85	CH87	21
CH40*	CH42*	CH85*	CH87*	22
CH44	CH46	CH81	CH83	23
CH44*	CH46*	CH81*	CH83*	24
CH48*	CH50*	CH77*	CH79*	25
CH48	CH50	CH77	CH79	26
CH52*	CH54*	CH73*	CH75*	27
CH52	CH54	CH73	CH75	28
CH56	CH58	CH69	CH71	29
CH56*	CH58*	CH69*	CH71*	30
CH60	CH62	CH65	CH67	31
CH64*	CH66*	CH61*	CH63*	33
CH64	CH66	CH61	CH63	34
CH68*	CH70*	CH57*	CH59*	35
CH68	CH70	CH57	CH59	36
CH72	CH74	CH53	CH55	37
CH72*	CH74*	CH53*	CH55*	38
CH76	CH78	CH49	CH51	39
CH76*	CH78*	CH49*	CH51*	40
CH80*	CH82*	CH45*	CH47*	41
CH80	CH82	CH45	CH47	42
CH84*	CH86*	CH41*	CH43*	43
CH84	CH86	CH41	CH43	44
CH88	CH90	CH37	CH39	45
CH88*	CH90*	CH37*	CH39*	46
CH92	CH94	CH33	CH35	47
CH92*	CH94*	CH33*	CH35*	48
CH96*	CH98*	CH29*	CH31*	49
CH96	CH98	CH29	CH31	50
CH100*	CH102*	CH25*	CH27*	51
CH100	CH102	CH25	CH27	52
CH104	CH106	CH21	CH23	53
CH104*	CH106*	CH21*	CH23*	54
CH108	CH110	CH17	CH19	55
CH108*	CH110*	CH17*	CH19*	56
CH112*	CH114*	CH13*	CH15*	57
CH112	CH114	CH13	CH15	58
CH116*	CH118*	CH9*	CH11*	59
CH116	CH118	CH9	CH11	60
CH120	CH122	CH5	CH7	61
CH120*	CH122*	CH5*	CH7*	62
CH124	CH126	CH1	CH3	63
CH124*	CH126*	CH1*	CH3*	64

2.3 TYPICAL CHARACTERISTICS

P/C Module: Typical Characteristics

Package	FASTBUS module P.C. board extended 3-7/8" through front panel
Power Supply Voltages	+5.0 Volts @ 0.6-2.74 A +3.5 Volts @ 4.27 A -2.0 Volts @ 2.42 A -3.5 Volts @ 4.08 A -5.2 Volts @ 4.45 A
Typical Power Dissipation	78 Watts
Required cooling air flow	400 ft/min. minimum
Required cooling air temp.	40 Degrees Celsius maximum entering module
Differential Input Signal Range	+/- 90 mV
Common Mode Signal Range	+/- 1 Volt
Input Impedance	50 Ω Transmission Line terminated with 0.1 μF in series with 50 Ω resistor.
Input Bias Current	0 A
Output Voltage	-1 to -1.8 Volts. Driving 100Ω termination to -2.0 V.
Propagation Delay	15.6 ns between input and output connectors measured with 80 mV differential input signal; DAC set for 250 mV (corresponds to 20 mV differential input signal).
Adjacent Channel Crosstalk	30 db
Hysteresis	10 mV
Triggering Uncertainty	0.5 mV
Ambient Temperature Range	0 to 40°C

3.0 P/C MODULE - THEORY OF OPERATION

3.1 BLOCK DIAGRAM

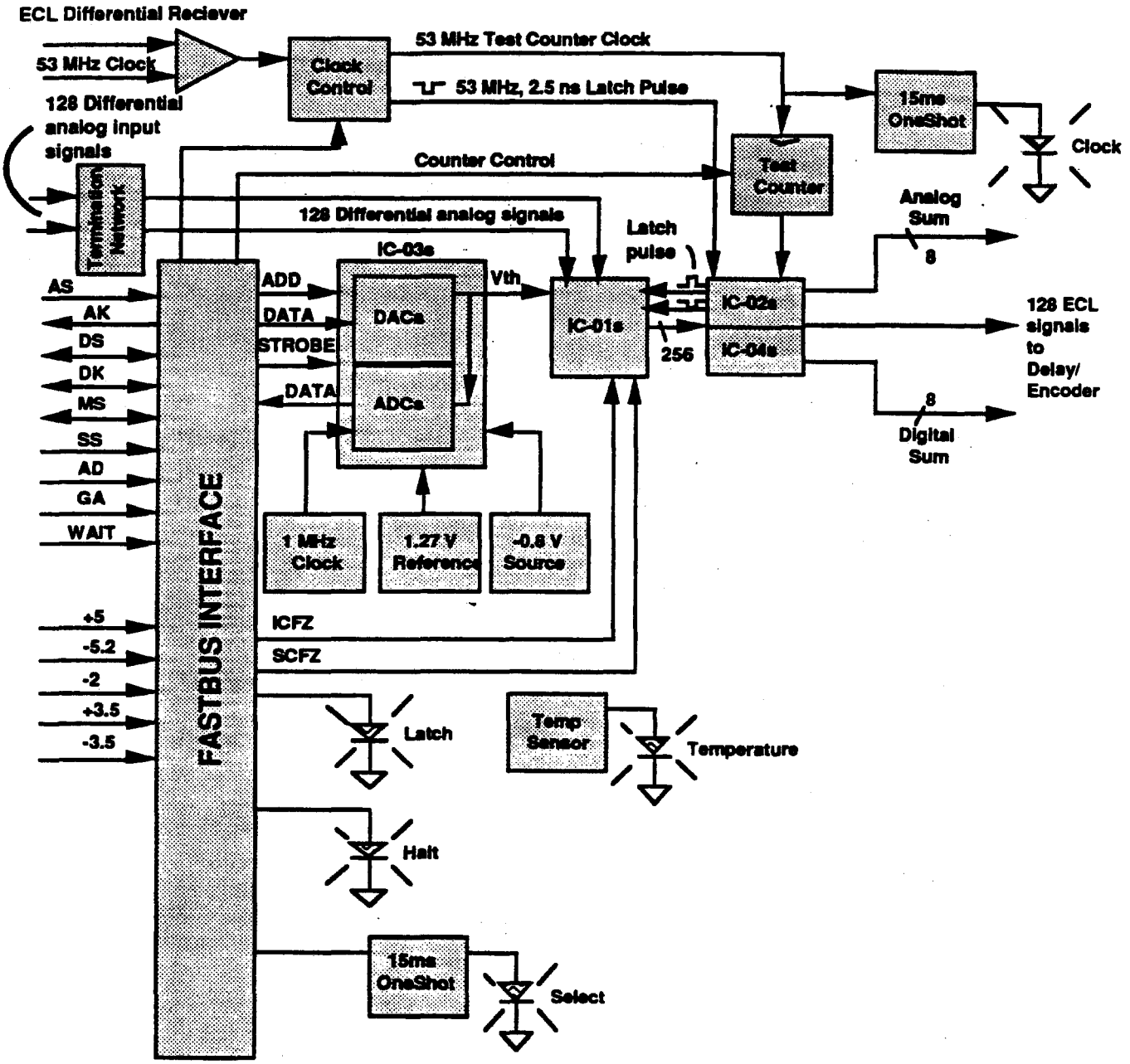


Figure 8: P/C Module Block Diagram

3.2 FASTBUS INTERFACE

The P/C Module is a FASTBUS slave. It responds to geographic addressing in control space only. It allows both single and block transfers and will respond with the following slave status responses. Information on the control space registers is found on pages 14 thru 18.

3.2.1 Slave Status Responses

The P/C Module asserts the following FASTBUS Slave Status (SS) Responses

SS=2 for end of block on DAC, Read or Write block transfers.

SS=6 for Reads or Writes to invalid addresses, or if an attempt is made to write to a DAC while the board is in the **RUN*(LATCH)*** mode.

SS=7 for Reads or Writes to invalid secondary addresses.

3.2.2 CSR0

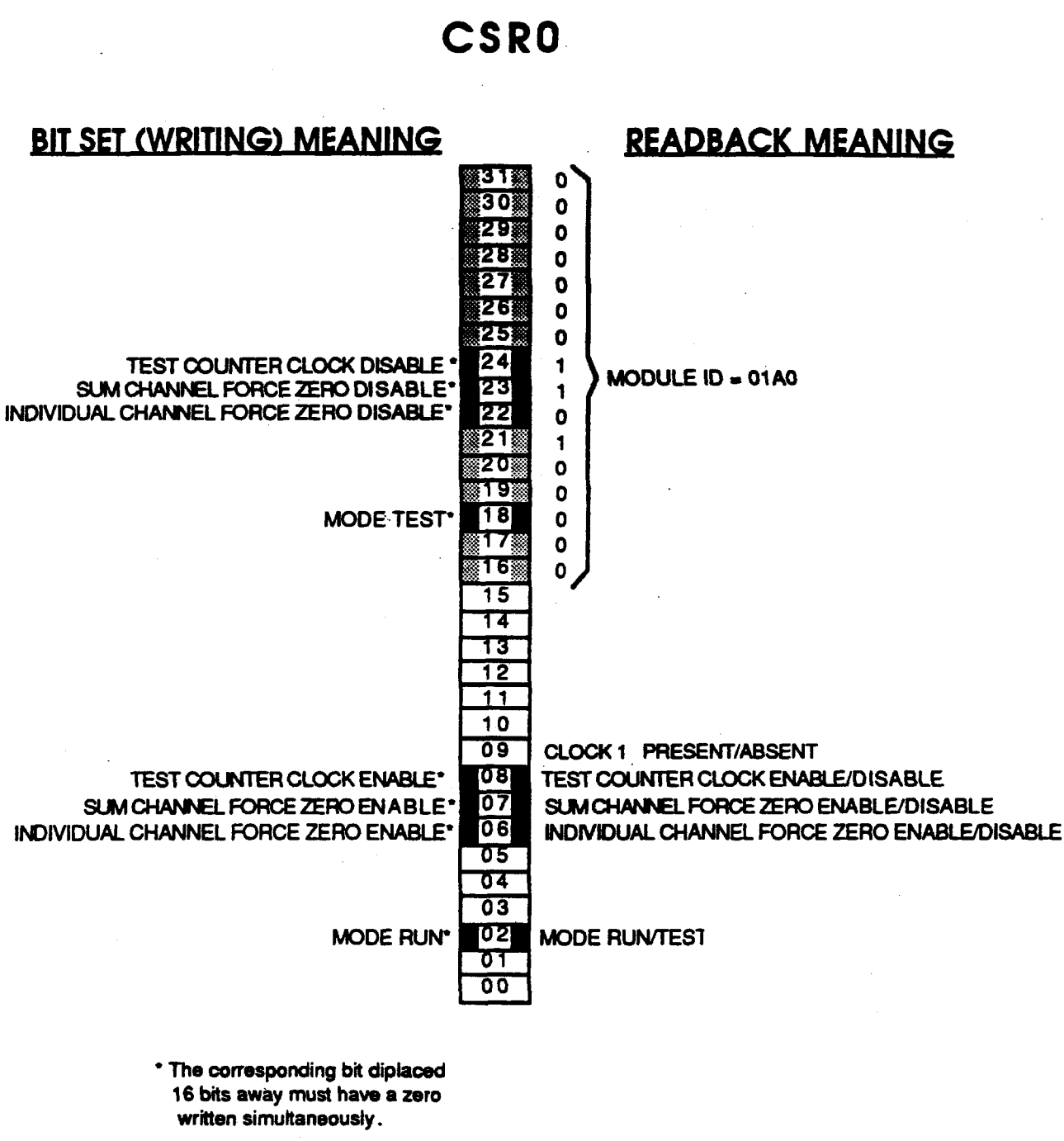


Figure 9: Control Space Register-0

CSR0 This is the main control and status register. It is a selective set and reset register with the bit assignments shown on the next page:

CSR0 Bit assignments:

BIT 02 -- **RUN**

Writing a "1" to this bit puts the module in the **RUN*(LATCH)*** mode. A "0" must be simultaneously written to BIT 18. This bit is cleared by writing a "1" to BIT 18 thus putting the P/C in the **TEST*(HALT)*** mode.

Reads a "1" back when in **RUN*(LATCH)*** mode or a "0" in **TEST*(HALT)*** mode.

BIT 06 -- **ICFZ__EN**

Writing a "1" to this bit forces the outputs of the individual channel latches to a zero. A zero must be simultaneously written to bit 22

BIT 07 -- **SCFZ__EN**

Writing a "1" to this bit forces the outputs of the Summed channel latches to a zero. A zero must be simultaneously written to bit 23

BIT 08 -- **CCLK__EN**

When in the **TEST*(HALT)*** mode, writing a "1" to this bit ENABLES the 8-bit test counter which, if the 53 MHz clock is present, counts at the clock frequency. The test counter is disabled by setting bit 24; the count remains at the count it stopped at or was last preset to.

While in the **RUN*(LATCH)*** mode, the eight output bits of the test counter are forced to zero.

BIT 09 -- **CLK1-ON**

Reads a "1" back if 53 MHz. clock is present.

BIT 18 -- **TEST**

Writing a "1" to this bit places the module in the **TEST*(HALT)*** mode. A "0" must be simultaneously written to BIT 02.

BIT 22 -- **ICFZ__DIS**

Writing a "1" to this bit DISABLES the "**Individual Channel Force Zero**" feature and allows the latches to track the data in the latched or transparent modes. A "0" must be simultaneously written to BIT 06.

BIT 23 -- **SCFZ__DIS**

Writing a "1" to this bit DISABLES the "**Summed Channel Force Zero**" feature and allows the latches to track the data in the latched or transparent modes. A "0" must be simultaneously written to BIT 07.

BIT 24 -- **CCLK__DIS**

Writing a "1" to this bit disables the clock to the 8-bit test counter.

BIT 16-31 -- **DEVICE ID**

The device ID (**01A0**) is read back on these bits.

3.2.3 CSR1

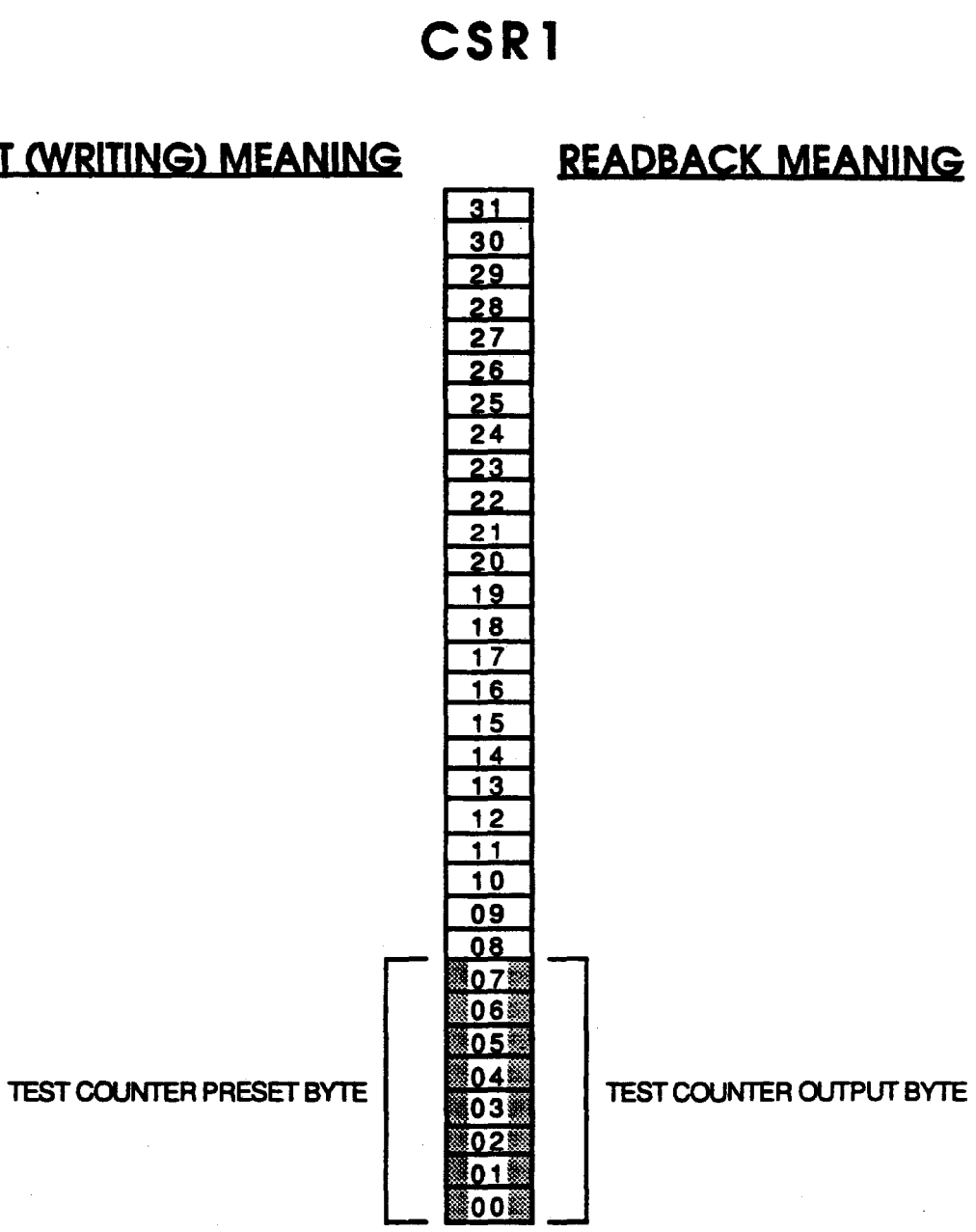


Figure 10: Control Space Register-1

CSR1 This register is used to write and read the 8-bit test counter. The counter is disabled and the outputs are forced to zero, when the **P/C** module is in the **RUN*(LATCH)*** mode.

BIT 00-07 Data to be written to or read back from the 8-bit test counter.

3.2.4 CSR10

CSR10

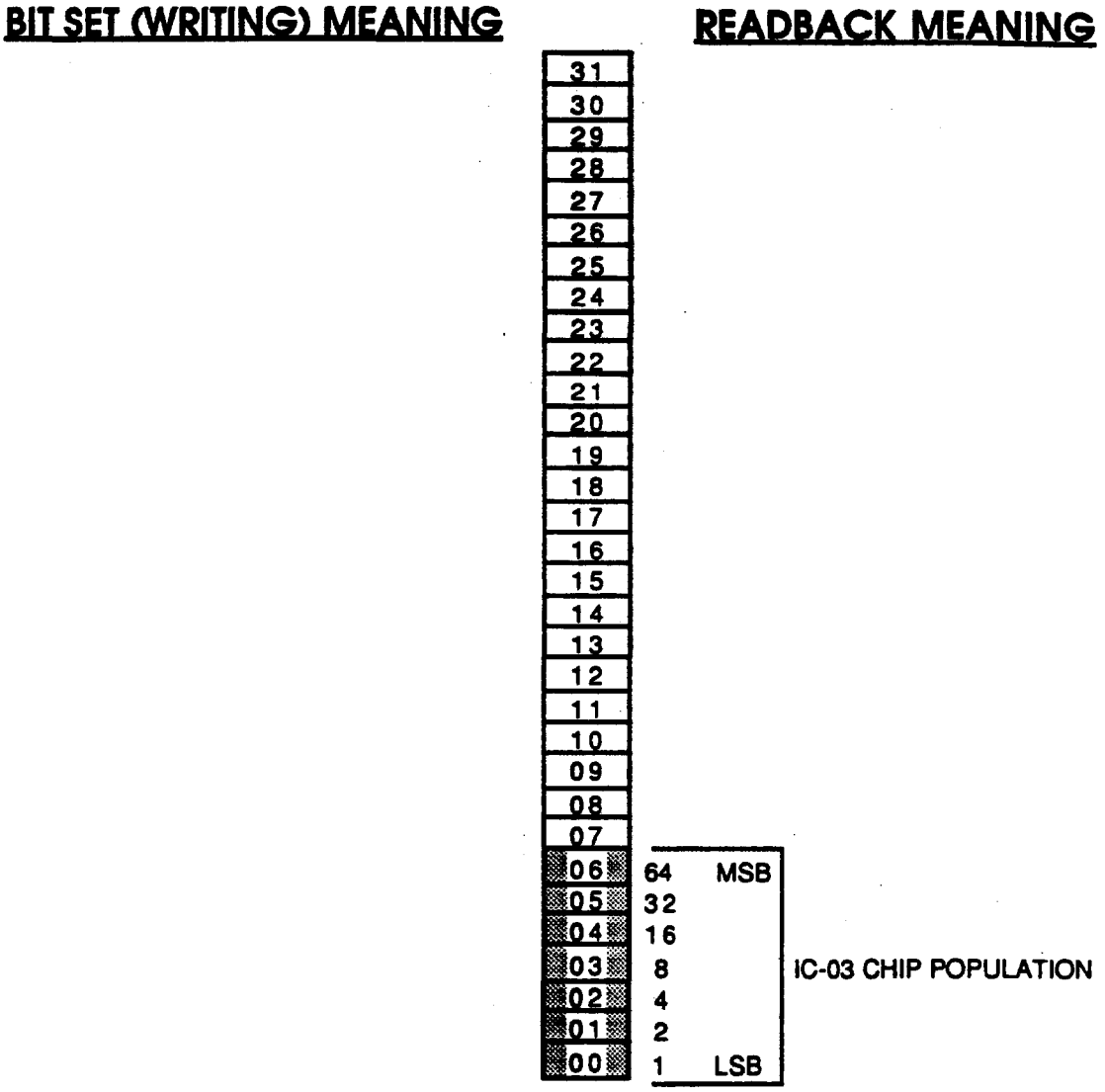


Figure 10: Control Space Register-10

CSR10 This register allows one to read back a value which indicates how this module is populated with IC-03 DAC/ADC IC's. The value read back is dependent upon the location of the three manually placed jumpers JU2, JU3 and JU4. Refer to section 3.5.2 CSR10 JUMPERS.
Although the bits have standard binary weighting, a "1" can exist in only one of the seven locations at a given time, thus one can only have IC-03 DAC/ADC populations of 1,2,4,8, 16,32 & 64, providing 4,8,16,32,64,128 & 256 DACs. A zero read back indicates a non-standard population.

3.2.5 CSR C000_0000 - C000_00FF

CSR C000_0000 through C000_00FF

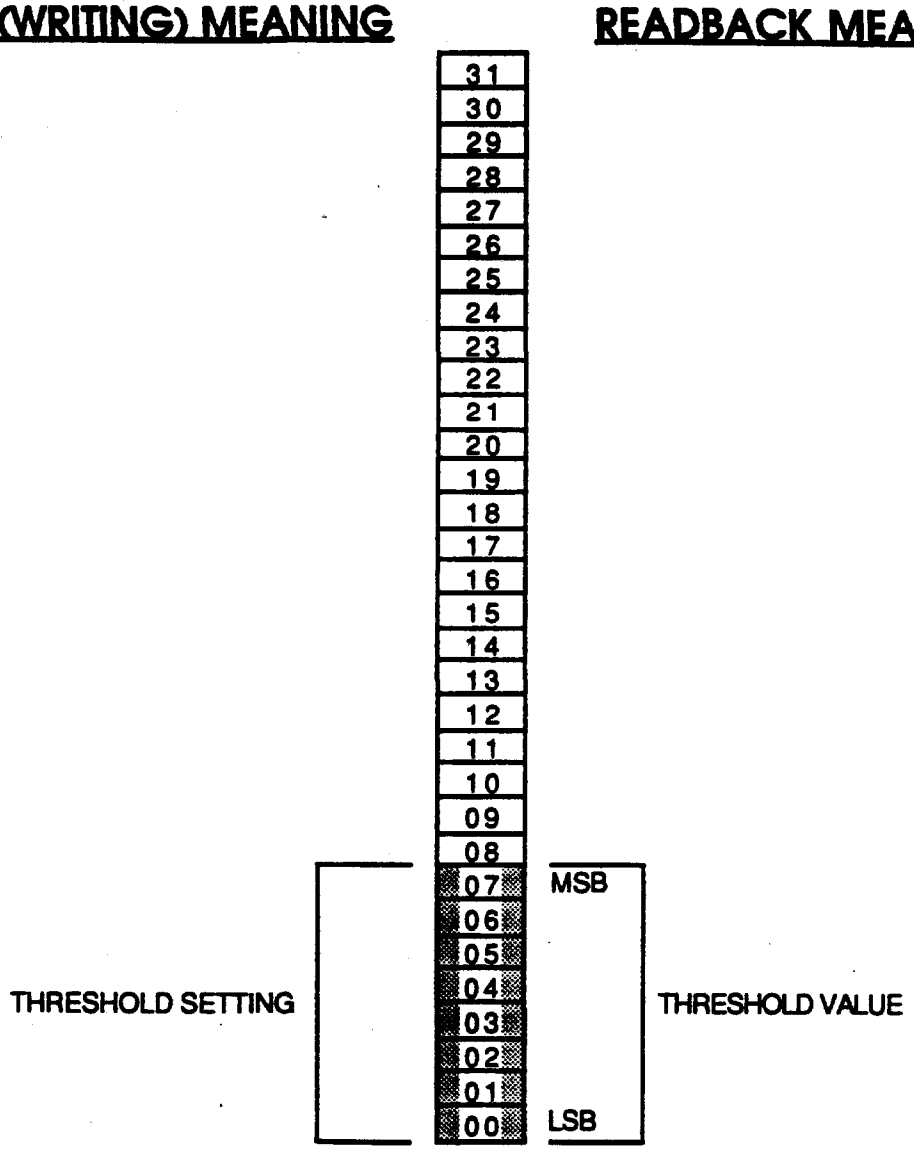


Figure 11: Control Space Register-C000_0000-C000_00FF

CSR C000__0000 - C000__00FF (Threshold Registers)

The 8 LSB of these 256 registers are used to write threshold settings or read threshold values from the IC-03 DAC/ADCs. These registers can be written only when the module is in the **TEST*(HALT)*** mode. If a write is attempted when in the **RUN*(LATCH)*** mode, the module returns an SS=6 error.

BIT 00-07 -- DATA Data to write/read the DAC/ADCs.

3.3 MODES OF OPERATION

There are two general operating modes of the P/C module; "**RUN*(LATCH)***" mode and "**TEST*(HALT)***" mode. The "**RUN*(LATCH)***" mode is used when data is being sent to the module through the input connectors and data is being latched. The "**TEST*(HALT)***" mode is used when the module is being tested or is communicating with FASTBUS. The module mode is set via the FASTBUS control space register; CSR0(refer to section "3.2.2 CSR0").

3.3.1 RUN*(LATCH)* Mode Operation

In "**RUN**" mode, the module is set up to latch the 256 channels of data that is present at the discriminator outputs(refer to section "3.4.2 IC-01..."). The data is latched every 18.9 ns by the *2.5 ns LATCH pulse* synchronized with the 53 MHz. clock. The 256 channels of latched data is passed to the logic of IC-02 and IC-04(refer to section "3.4.3 IC-02....") where it is reduced to 128 single ended channels of ECL level signals(to be passed via the FASTBUS Auxiliary *backplane to the DE)as well as 8 differential analog sum signals and 8 differential NHIT signals*, both also sent to the Auxiliary backplane for further processing.
During "**RUN**" mode the module is accessible by FASTBUS only to manipulate control space register CSRO so as to place the module into "**TEST**" mode, attempts to access other control space registers will result in an SS=6 error.
The ***LATCH*** LED on the front panel should be lit during "**RUN**" mode.

3.3.2 TEST*(HALT)* Mode Operation

"**TEST**" mode is used either to test the module via an 8-bit on board test counter(refer to section "3.4.6 Test Counter Circuit") or to manipulate FASTBUS controllable circuitry such as the control space registers or to change DAC threshold settings.
The ***HALT*** LED on the front panel should be lit while the module is in "**TEST**" mode.

3.4 CIRCUITRY OPERATION

3.4.1 Inputs

128 differential SSD preamplifier signals are received by the P/C module via four, 64-conductor ribbon cables. The input signal order does not correspond to the order of the silicon strips in the detector for reasons having to do with signal density on both the detector itself and the pre-amplifier cards connected to the detector. The input signal traces on the P/C module between the input connectors and discriminators are routed to restore the monotonic strip order of the detector. These differential signals are AC coupled through 0.1μF capacitors near the inputs of the *IC-01 ASIC's, both sides of which are terminated through 50 Ω to ground. These* termination components are housed in a custom 10-lead sip each of which will terminate two channels (4 wires) of the 128 channels; thus 64 sips per module.

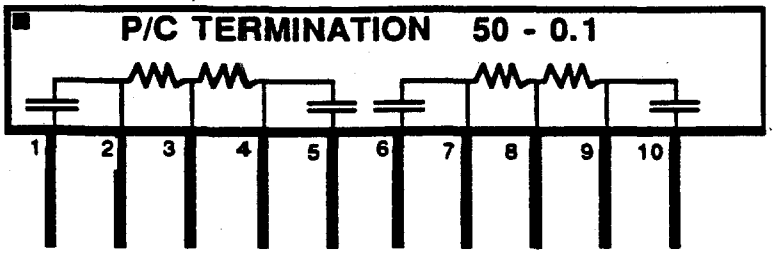


Figure 13: P/C Termination

3.4.2 IC-01 (Two Channel Sum, Discriminator and Latch)

IC-01 is a bipolar ASIC made using a Tektronix Quickchip 2K-130 linear array. It contains two linear summing amplifiers, four time-over-threshold discriminators(each with a 10 mV hysteresis and variable threshold setting capability), and four latches.

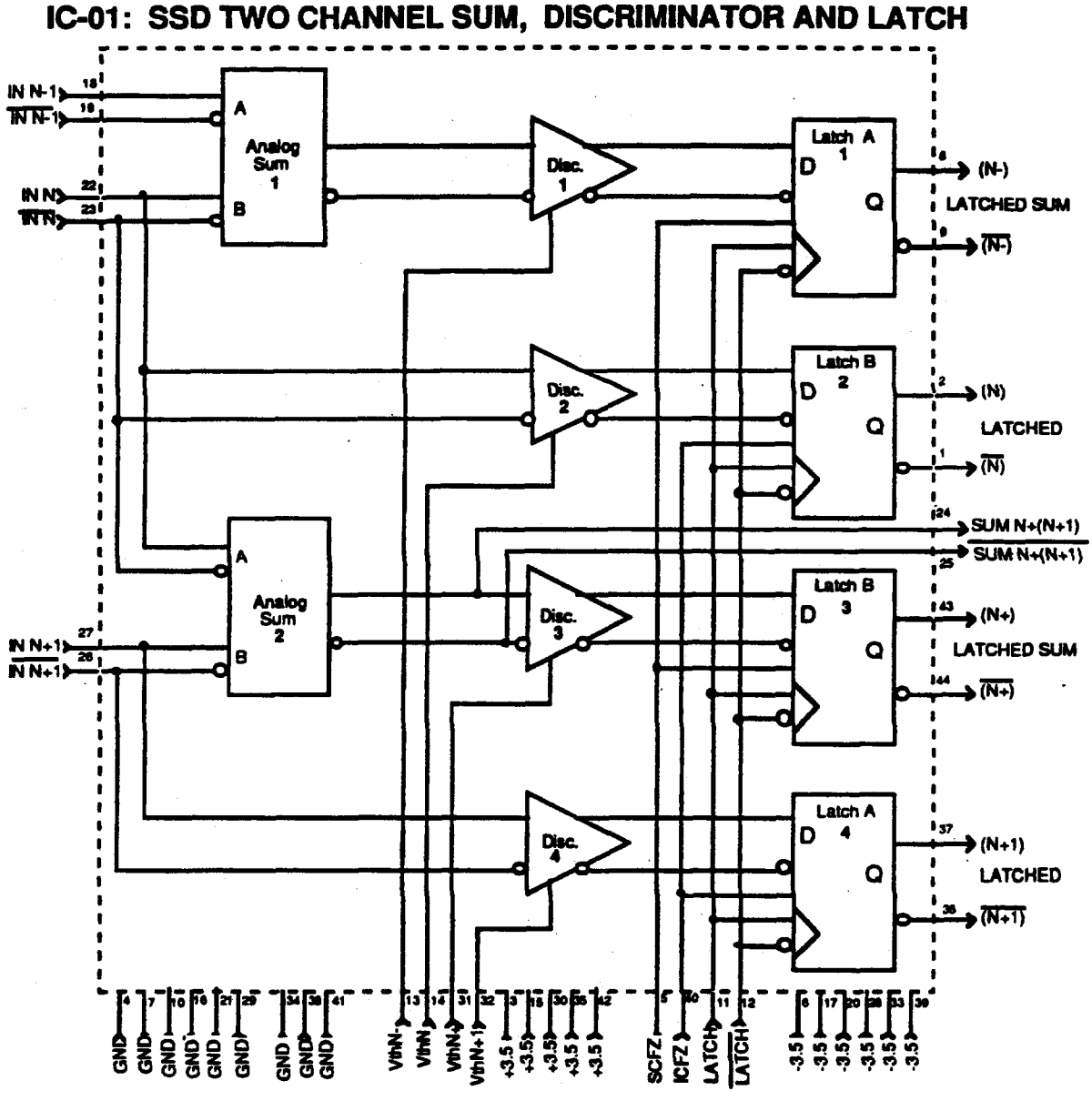


Figure 14: IC-01: Two Channel Sum, Disriminator and Latch

Each IC-01 (except for the units with the lowest and highest numbered channels on the board) receive signals from three consecutive silicon strips. The first and third signals are also input to neighboring IC-01 chips. Two linear sums are made; the first by adding the first and second inputs, and the second by adding the second and third inputs. The output of each of the two linear sums is input to a discriminator. The output of the second sum is also sent to IC-02 for additional summations. The second and third inputs are connected to the remaining two discriminators. The result is two pairs of discriminators, one pair discriminating individual strip signals (inputs two and three), and the other pair discriminating the two linear sum outputs. Each discriminator has a separate threshold settin input.

Typical Discriminator differential input voltage versus threshold voltage setting are shown in Appendix E (IC-01 ASIC data sheets). The 10 mV hysteresis band prevents the discriminators from following extraneous signals(white noise, cross-talk, overshoot, radiation noise, etc.) As the graphs show the discriminators are linear up to approximately 50 mV differential signals. The device has been characterized up to 90 mV differential.

The output of each of the four discriminators is sent directly to a latch (one per discriminator output) which is controlled by a differential, *2.5 ns wide LATCH pulse*. This *LATCH pulse*, common to all four latches, controls the function of the latches, which can be thought of as digital sample-and-holds. When the *LATCH pulse* is present the latches are in the "transparent" state during which time the latch output follows the discriminator output. When the *LATCH pulse* is not present, the latches are in the "latched" state and the latch outputs remain unchanged.

In normal operation, an IC-02 located within four inches of it's associated IC-01's, provides this *2.5ns wide LATCH pulse*. This *LATCH pulse* is synchronized to the 53 MHz CLK signal received from the Seq. This pulse switches the latch input from "latched" mode to "transparent" mode for a time equal to the width of the pulse (approximately 2.5 ns), and then returns it to the "latched" mode. The result is that the state of the discriminator is latched and held for one RF cycle less the width of the *LATCH pulse*(see diagram below). Two more control signals, called INDIVIDUAL CHANNEL FORCE ZERO and SUMMED CHANNEL FORCE ZERO, also effect the operation of the latches. INDIVIDUAL CHANNEL FORCE ZERO is connected to the two individual strip latches and SUMMED CHANNEL FORCE ZERO is connected to the sum latches. When FORCE ZERO is HI, the effected latches are forced to logical zero. This function is used to insure that no signal is received from an IC-01 chip while the P/C module is in the **TEST*(HALT)*** mode. It also makes it possible to disable either bank of discriminators while leaving the other bank operational.

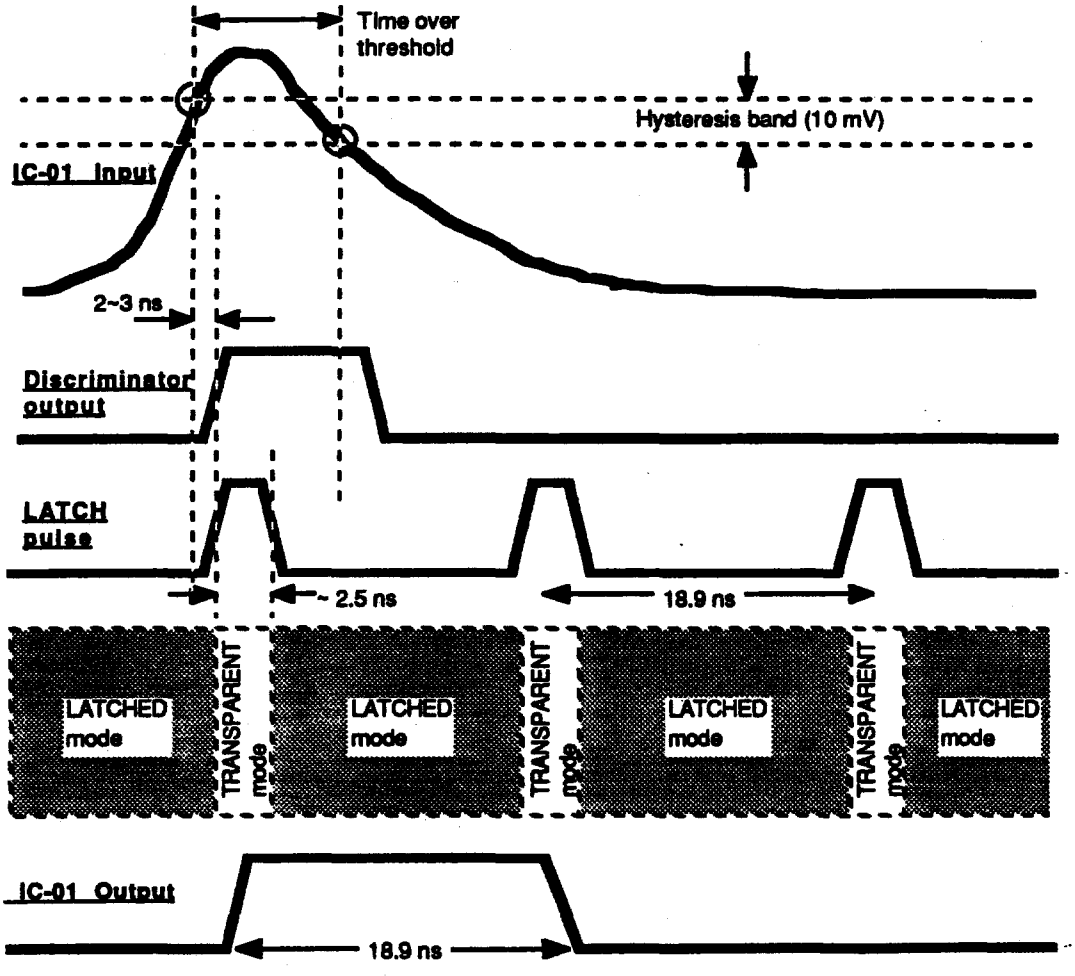


Figure 15: IC-01 Function Diagram

3.4.3 IC-02 (3-Channel Logic, Quad Analog Sum & Latch Driver) and IC-04 (5-Channel Logic & Octal NHIT)

The IC-01 latch outputs are input to two other bipolar ASIC's; IC-02 and IC-04 also made usi Tektronix Quickchip 2K-130 linear arrays. These two chips as a group can be broken down into fo functions: 1.)Logic, 2.)Quad **ANALOG SUM**, 3.)**NHIT** or **DIGITAL SUM**, and 4.)*LATCH pulse* buffer. The four functions will be discussed in order starting with the logic. Internal block diagrams of the two ASICs can be found on the following page as well as in Appendix E.

The logic functions as follows, the combination of these two chips accept inputs from four IC-01's (8 SSD channels), and produces eight ECL output signals which are sent, via the auxiliary backplane, to the **DE** module in the adjacent slot. Since the outputs of the latches of IC-01 are differential, the inputs to the logic on both IC-02 and IC-04 are differential receivers. The function of the receivers is three-fold; convert from differential to single ended, shift the level of the latch outputs to the the level of the logic gates, and to switch input polarity by switching inputs to a receiver where a polarity change is necessary. The logic producing the eight ECL output signals performs in the following way: any given output channel (N) is set HIGH if the corresponding single strip latch (N) is set, or if one of the 2 summed-channel latches including strip N is set, but neither of the individual channel latches for the two channels contributing to the sum are set.

These logic signals are OR'ed with test inputs(t1 and t2) which sit at logical low levels while the module is in the **RUN*(LATCH)*** mode. Test signals are sent to these test inputs in the **TEST*(HALT)*** mode during which time the latch outputs are forced to a logical zero. The test inputs are implemented so that the adjacent **DE** module can be tested with a fixed or incremental pattern.

Recall that IC-01 supplied an output which was produced from the analog sum of two individual channel inputs, this output goes directly to one of the inputs of the quad **ANALOG SUM** of the IC-02. The four IC-01s(8 SSD channels) each supply the quad **ANALOG SUM** with an output which in turn the quad **ANALOG SUM** converts to an analog sum of the same eight strips used in the logic function above. The output current of the quad **ANALOG SUM** is a linear sum of the eight individual strip signals.

IC-04 is used to produce a signal known as **NHIT** or **DIGITAL SUM** which consists of a current, the magnitude of which is proportional (100 uA/strip hit) to the number of hit strips in the group of 8 SSD channels. The **NHIT** inputs are constructed by both IC-02 and IC-04 logic, forming a logical OR of a single strip signal and one of the summed channel signals which has that strip as one of it's inputs. The result is that the output of **NHIT** will register a ONE if a hit is shared by two adjacent strips even though two bits (for two adjacent strips) are set in the **DE**.

The *LATCH pulse* buffer of IC-02 is designed to accept a narrow pulse (2.5 ns wide) and fan out a differential version of this pulse to the latch inputs of four **IC-01** chips.

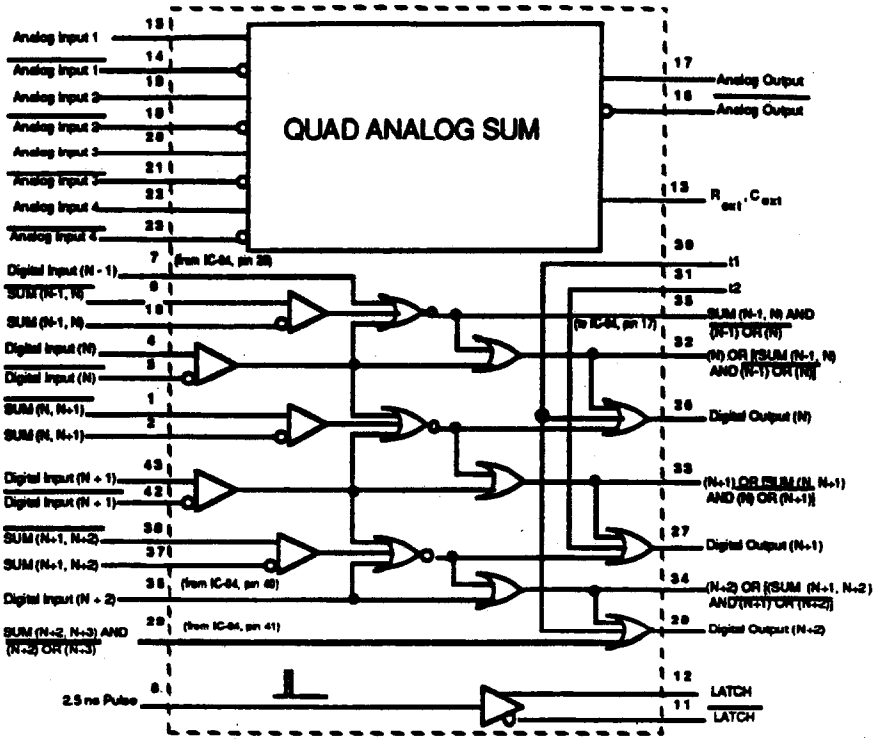


Figure 16: IC-02: Three Channel Logic, Quad Analog Sum & Latch Driver

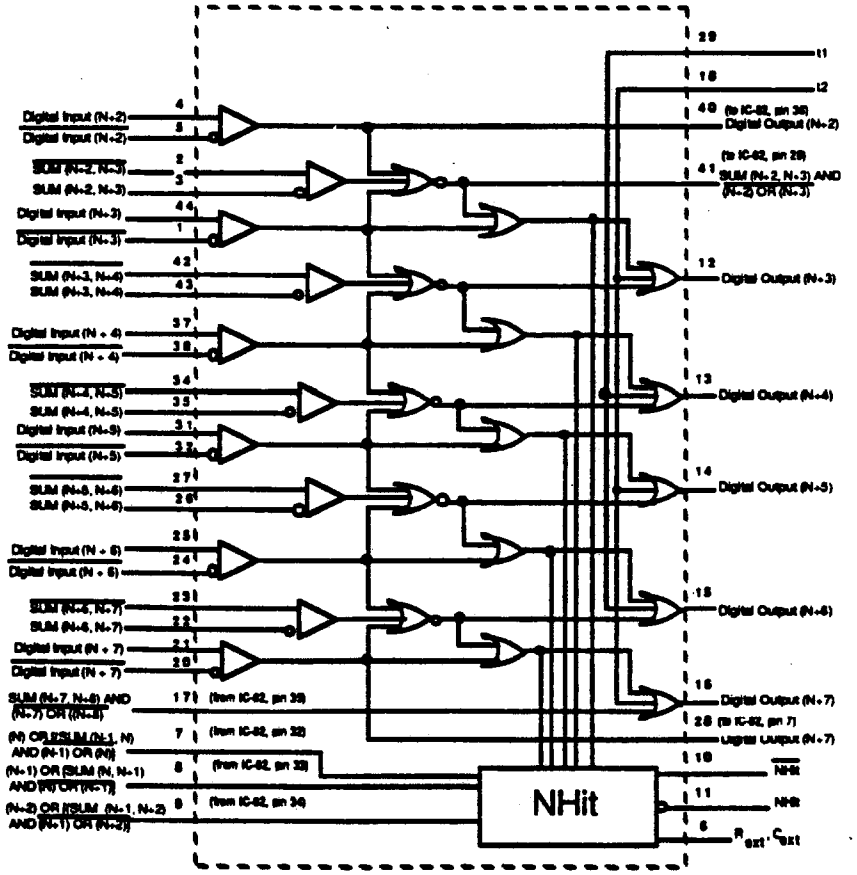


Figure 16: IC-04: Five Channel Logic and Octal NHIT

3.4.4 IC-03 DAC/ADC

Voltage levels are provided for the threshold voltage (Vth) inputs of the **IC-01 (Sum, Discriminator and Latch)** chips by a CMOS ASIC produced by United Silicon Structures. This chip contains four 8 bit D/A converters and one 8 bit A/D which requires an on board 1 MHz clock. An internal block diagram of IC-03 is shown below.

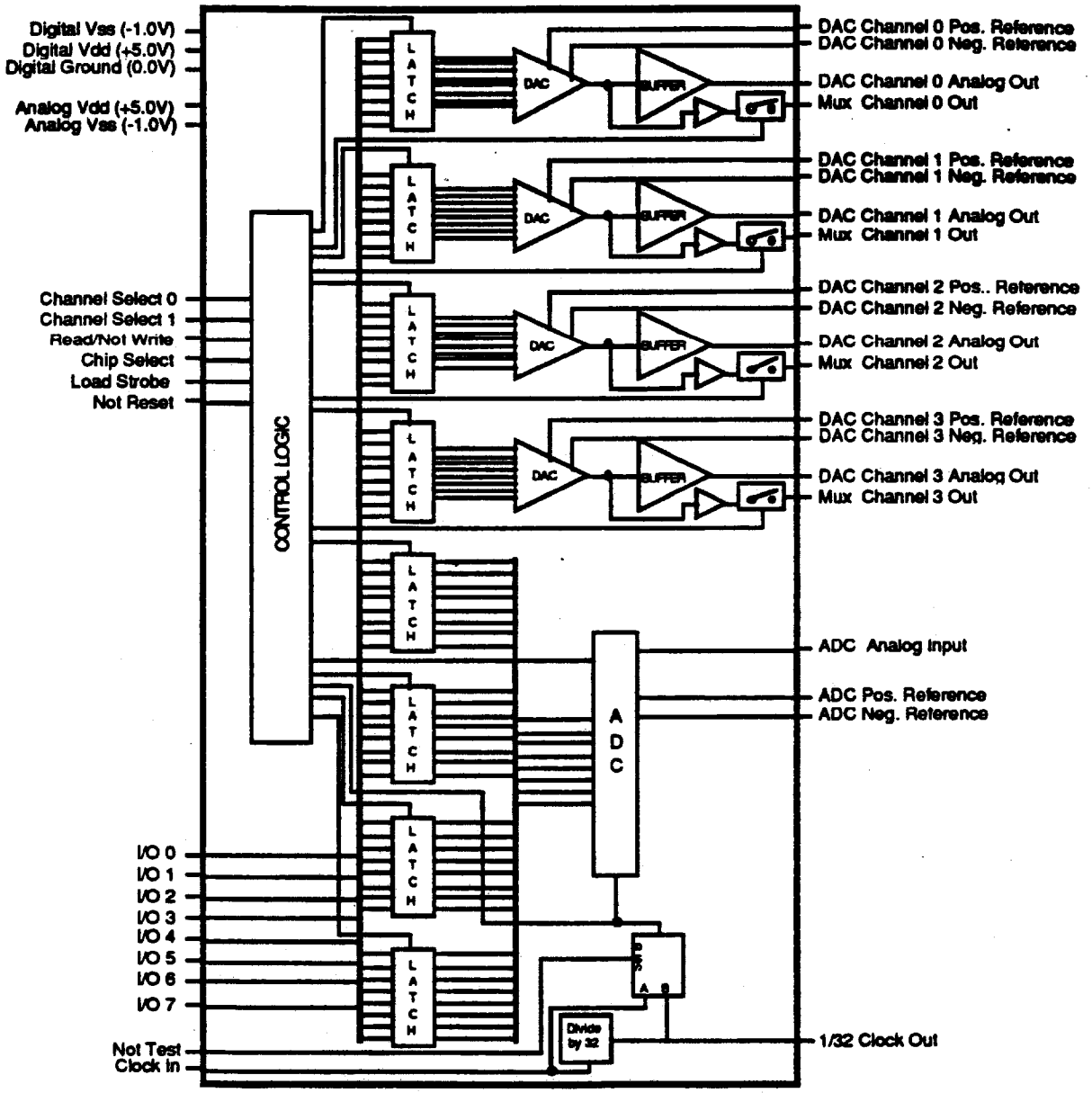


Figure 18: IC-03 Quad DAC/ADC

64 IC-03 chips may be mounted, providing individual control of all 256 discriminators, or DAC outputs may be jumpered together to allow the use of as few as one IC-03 chip to control all discriminators. The IC-03 chips share 8 bussed I/O lines and 4 control lines. Each chip has a separate select line. The DAC output range is from 0V to the reference +1.27V. The output impedance of these DACs varies between 10 and 30 Ω. Each chip requires power supply voltages of -0.8V, +5.0V, and a reference voltage of +1.27V.

Once a DAC has been written a value, one must wait for 32 counts of the 1 MHz clock before the data can be read properly.

3.4.5 2.5ns LATCH Pulse Circuitry

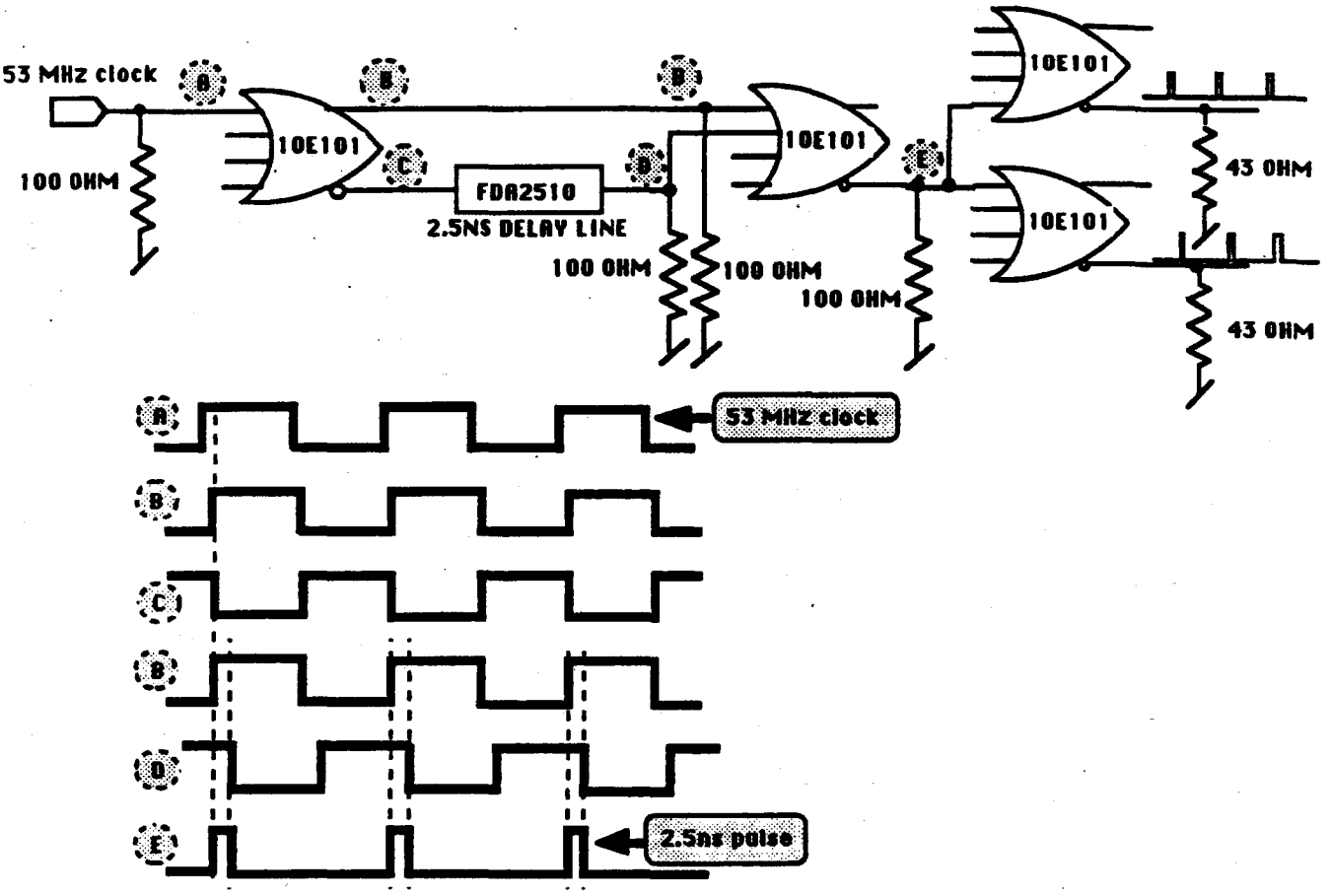
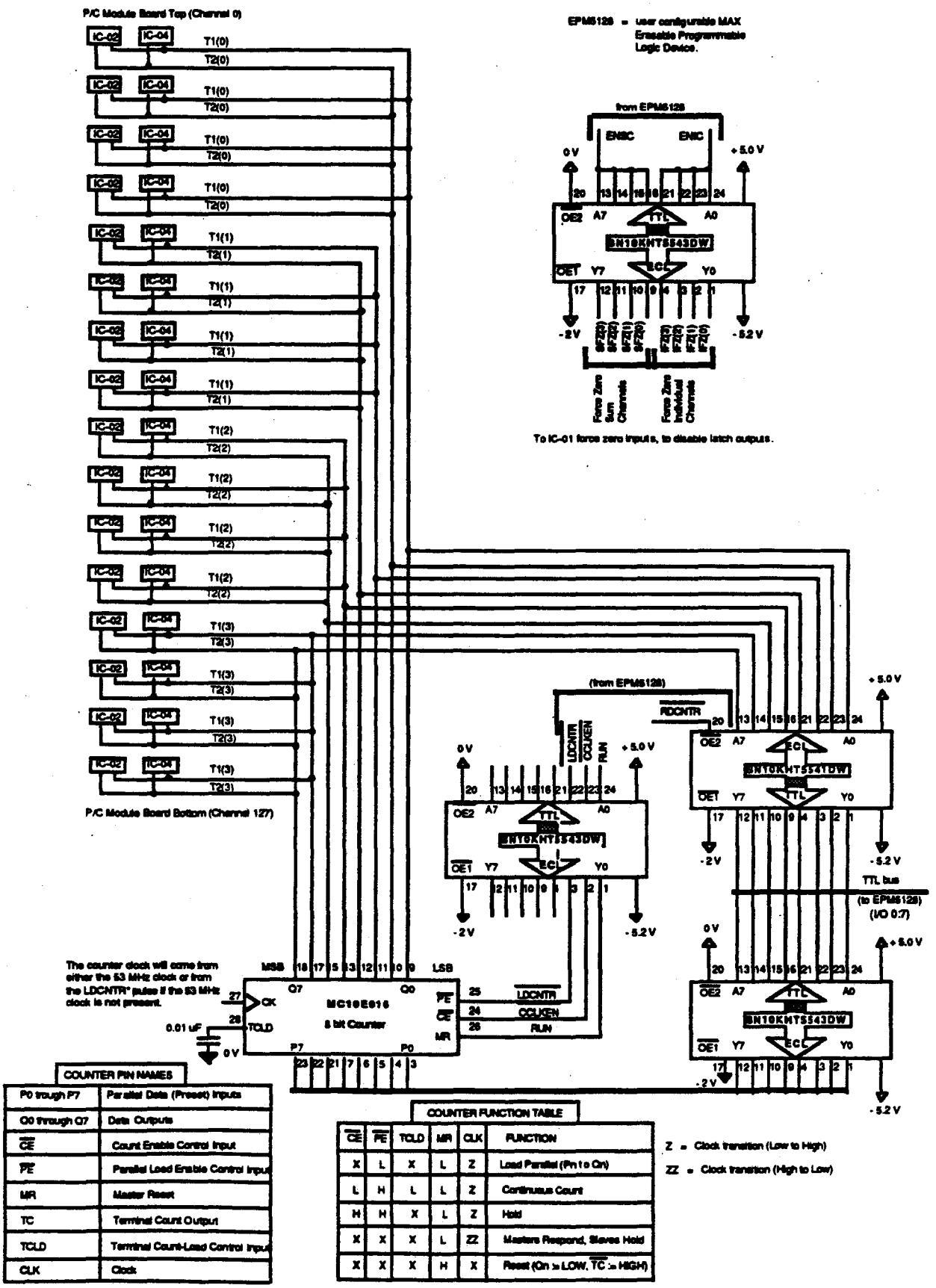


Figure 19: 2.5ns Latch Pulse Generation Circuit

3.4.6 Test Counter Circuit

A presettable 8 bit counter is included on the **P/C** module to provide a means of sending test data to the associated **DE** module. This counter is controlled via CSR0. The counter's outputs are reset (forced to zero) when the **P/C** module is in the **RUN*(LATCH)*** mode. When the board is in the **TEST*(HALT)*** mode, the counter may be set in the count or hold mode. In the hold mode, the counter output pattern can be altered by presetting the counter via CSR1. In the count mode, the counter is incremented by the CLK (53 MHz) signal. The control to and connection of outputs from the test counter is illustrated on the following page. This 8-bit counter provides 256 unique patterns which are sent to the **DE** module through the IC-02 and IC-04 test inputs(t1 and t2).



P0 trough P7	Parallel Data (Preset) Inputs
Q0 through Q7	Data Outputs
$\overline{CE}$	Count Enable Control Input
$\overline{PE}$	Parallel Load Enable Control Input
MR	Master Reset
TC	Terminal Count Output
TCLD	Terminal Count-Load Control Input
CLK	Clock

$\overline{CE}$	$\overline{PE}$	TCLD	MR	CLK	FUNCTION
X	L	X	L	Z	Load Parallel (Pn to Qn)
L	H	L	L	Z	Continuous Count
H	H	X	L	Z	Hold
X	X	X	L	ZZ	Masters Respond, Slaves Hold
X	X	X	H	X	Reset (Qn = LOW, $\overline{TC}$ = HIGH)

Figure 20: Test Counter Circuitry

3.5 HARDWARE JUMPERS

3.5.1 DAC/ADC Jumpers

The illustration is an abbreviated pattern of the IC-03 DAC/ADC jumpering scheme which exists on the P/C Module.
There are 256 DACs which are addressed via CSR addresses C000_0000 through C000_00FF inclusive. DAC address 00 (Hex), will always set the threshold voltage on the uppermost trace which is the threshold for the discriminator connected to the output of the summed channel which takes the sum of channel -1 and channel 0 (Σ-1,0). Likewise DAC address 01 (Hex), will always set the threshold voltage on the trace immediately below the uppermost trace which is the threshold for the discriminator connected directly to the individual channel #0 input. DAC address 02 (Hex), will always set the threshold voltage for the discriminator connected to the output of the summed channel which takes the sum of channel 0 and channel 1 (Σ0,1). DAC address 03 (Hex), will always set the threshold voltage for the discriminator connected directly to the individual channel #1 input. This pattern continues for the 256 DACs and the 256 discriminator threshold setting inputs.
It should be obvious that if all 256 DAC's are employed, no jumpers are required and each discriminator has an individual DAC to control it's threshold. However, in the event that only 4 of the 256 DAC's are mounted on the P/C Module, the jumpers would probably be applied such that DAC 00 (hex) would drive every other sum channel. i.e., Σ(0), Σ(1,2), Σ(3,4), Σ(5,6) etc. DAC 01 (hex) would drive the even individual channels; DAC 02 (hex) would drive sum channels Σ(0,1), Σ(2,3), Σ(4,5), Σ(6,7) etc., and DAC 03 (hex) would drive the odd, individual channels.
The common choices of DAC populations are 4,8,16,32,64,128 or 256 which corresponds to 1,2,4,8,16,32 and 64 IC-03 packages. It seems reasonable to assume that for four DAC's every fourth threshold lead would be connected together and so on. However, any and all possibilities exist for connecting DAC outputs to threshold inputs and the method used may be dictated by the detector geometry.

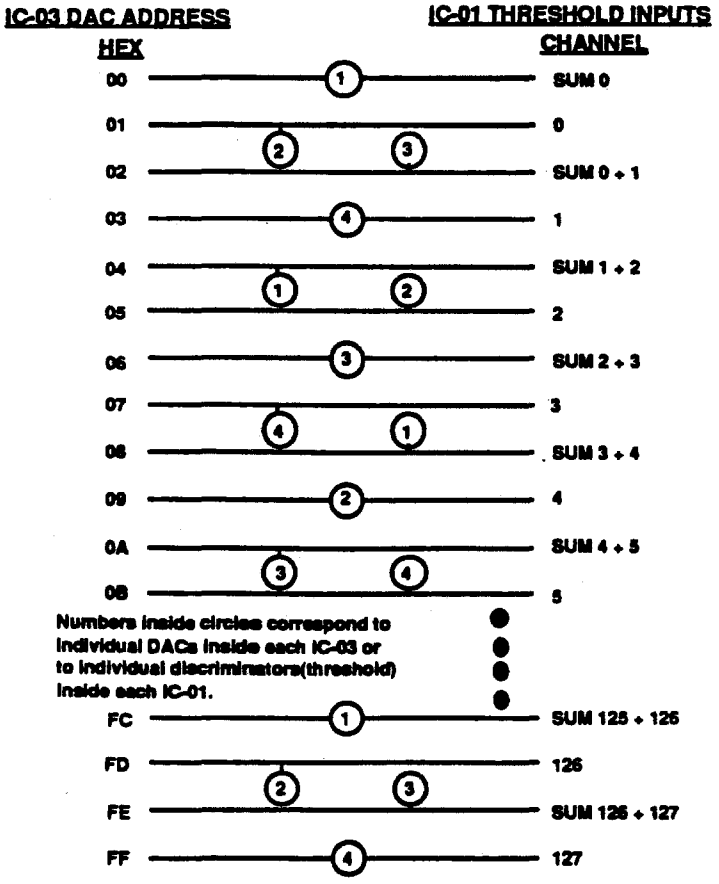


Figure 21: DAC/ADC Jumpering

3.5.2 CSR10 Jumpers

The value found in CSR10 corresponds to the configuration of three, two position jumpers(JU2, JU3, and JU4) that connect a signal trace to ground(GND) or +5V. The physical module location of the three posts for each jumper, and their connections are shown below. Table 1 on the following page provides the relationship between DAC population, jumper location, CSR10 readback and lowest ordered address for each DAC chip.
When less than 256 DAC's are utilized, the P/C module will be populated from the lowest address onward without skipping addresses, this is done by a mapping routine within the EPM5128 programmable PAL. It has been done this way to make the DACs accessible by block transfer. The PAL receives a 3 bit code from the jumpers to indicate the number of IC-03 DAC/ADC chips mounted and coverts the sequential DAC address it receives from an external source to the appropriate DAC module address. I.E. if only two IC-03s(8 DACs) are used and located at DAC board address 0 through 3(U97) and DAC board address 64 through 67(U129), the PAL knowing from the jumpering scheme will only accept DAC addresses 0 through 7 from an external source and in turn converts external address 0 to DAC board address 0 and so on up to external address 7 to DAC board address 67. When less than 256 DACs are employed the IC-03s physical location is important to insure that the external address received is converted to the appropriate DAC board address. The physical locations for the common number of IC-03s used are shown in Table 2 on the following page.

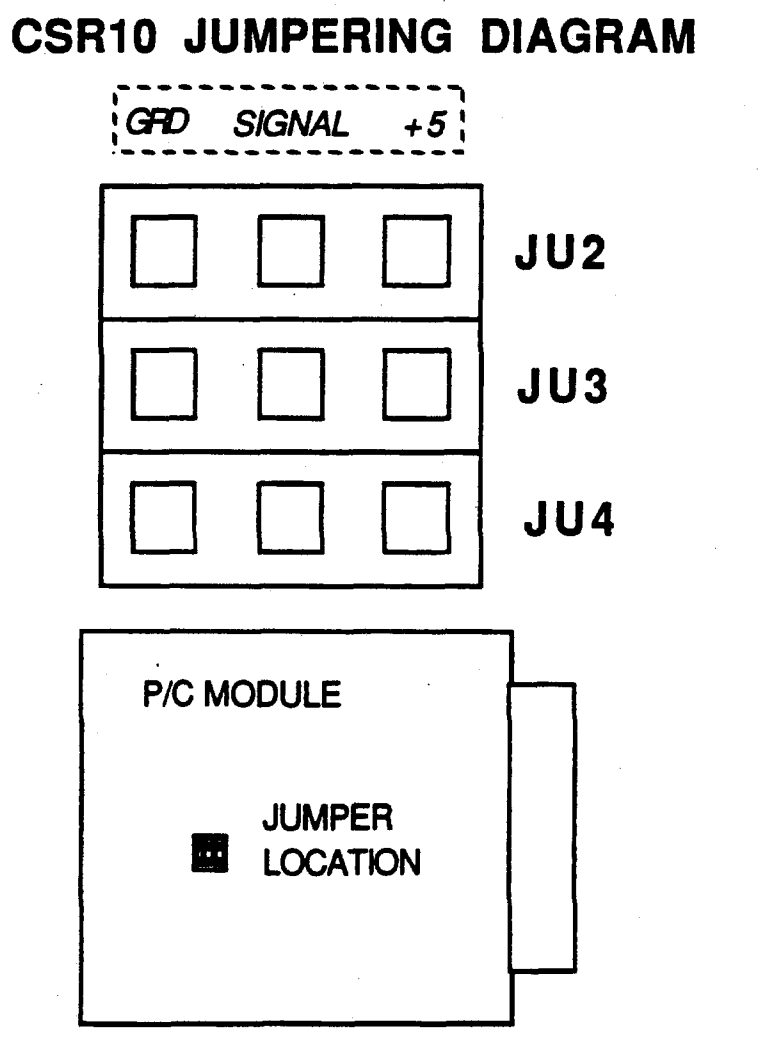


Figure 22: CSR10 Jumpering

Table 1

Jumper Numbers			# of DACs	CSR10 Value	First DAC board address of each IC-03
JU2	JU3	JU4			
Gnd	Gnd	Gnd	????	0	This is a special case, unique to the user.
Gnd	Gnd	+5V	4	1	0
Gnd	+5V	Gnd	8	2	0,80
Gnd	+5V	+5V	16	4	0,40,80,C0
+5V	Gnd	Gnd	32	8	0,20,40,60,80,A0,C0,E0
+5V	Gnd	+5 V	64	10	0,10,20,30,40,50,60,70,80,90,A0,B0,C0,D0,E0,F0
+5V	+5V	Gnd	128	20	0,8,10,18,20,28,30,38,40,48,50,58,60,68,70,7880,88,90,98,A0,A8,B0,B8,C0,C8,D0,D8,E0,E8, F0,F8
+5V	+5V	+5V	256	40	All addresses from 00 through FF inclusive are valid

Table 2

Number of IC-03s mounted(#DACs)	IC-03 Physical location on module
1(4)	U97
2(8)	U97, U129
4(16)	U97, U113, U129, U145
8(32)	U97, U105, U113, U121, U129, U137, U145, U153
16(64)	U97, U101, U105, U109, U113, U117, U121, U125, U129, U133, U137, U141, U145, U149, U153, U157
32(128)	U97, U99, U101, U103, U105, U107, U109, U111, U113, U115, U117, U119, U121, U123, U125, U127, U129, U131, U133, U135, U137, U139, U141, U143, U145, U147, U149, U151, U153, U155, U157, U159
64(256)	ALL LOCATIONS

4.0 P/C MODULE - TESTING AND CALIBRATION

4.1 UNPOWERED TEST

A test should be performed to check for shorts between the power planes (layers) of the P/C module. The test should verify no direct connection between any of the following power planes: -5.2V, -3.5V, -2V, -.8V top, -.8V bottom, +5V, +3.5V, +1.25V, and ground. The test can be done with an OHM METER. Connection to the power plane can be found at the following locations, with the measured impedances between the power and ground plane with the PC board fully assembled(number of IC-03s equals four).

-5.2V = right side of D5	= 8 ohm
-3.5V = right side of fuse f4	= 3 ohm
-2V = right side of fuse f1	= 22 ohm
-.8V top = right side of CR2	= 23 ohm
-.8V bottom = right side of CR4	= 23 ohm
+5V = right side of fuse f6	= 400 ohm(90 ohm with EPM5128 installed)
153 = right side of fuse f2	= 580 ohm
+1.25V = case of Q1	= 73 ohm
GROUND = left side of D5	

NOTE: Right side is towards the front of the board where the LABEL is located. The drawing "POWER TEST POINT DETAILS"#2563.000-MD-215808 shows the power pin locations of various devices on the top side of the board, this should help in finding shorts.

4.2 AUTOMATED TESTING

Their is an "Acceptance Test Description" document that uses two software packages to verify wether a P/C module is acceptable from an outside vendor. This document can be found in APPENDIX A.

4.3 CALIBRATION

Adjusting the value at which the ***TEMP*** LED on the front panel of the P/C module will turn on is the only calibration needed for the module. Adjusting trim pot R295 sets the temperature at which the LED will turn on and off, the setting of R295 controls the voltage on pin 2 of U185. If pin 3 of U185 which is connected to the LM35 temperature sensor is at a more positive value than pin 2 of U185 the LED will turn on. The voltage on pin 2 should be set to 0.5V this corresponds to 50 degrees Celsius. The LM35 will output to pin 3 of the U185 a 10 mV per degree Celsius voltage level. If the temperature rises above 50 degrees Celsius(0.5V on pin 3) the LED will turn on. The LED will turn off if the temperature falls below 50 degrees Celsius. The ***TEMP*** LED and associated circuitry can be tested by adjusting R295 for a voltage on pin 2 of U185 that is less than the value on pin 3 of U185, the lower voltage should cause the ***TEMP*** LED to turn on.

5.0 APPENDICES

APPENDIX A - TEST SOFTWARE
APPENDIX B - FASTBUS INTERFACE PAL PROGRAM
APPENDIX C - POSTAMP/COMPARATOR MODULE - SCHEMATIC
APPENDIX D - MODULE DRAWINGS
APPENDIX E - DATA SHEETS

APPENDIX A - TEST SOFTWARE

This appendix contains an acceptance criteria specification that pertains to the testing of the POSTAMP/COMPARATOR printed circuit board.

POSTAMP/COMPARATOR MODULE - ACCEPTANCE TEST DESCRIPTION

The party performing the acceptance test on the POSTAMP/COMPARATOR module should be familiar with the "IEEE Standard FASTBUS Modular High-Speed Data Acquisition and Control system"(ANSI/IEEE Std. 960-1986), the POSTAMP/COMPARATOR HARDWARE DESCRIPTION-HN100, and the two software packages listed below which the acceptance criteria is based upon. Questions pertaining to the documents should be addressed to the appropriate originator.

SILICON STRIP DETECTOR (SSD)
System Test Software Guide
Version: 0 15 march 1991

Originators:
Wolfgang Kowald Duke University
EXP.771 #708-840-4250
Panagiotis Spentzouris UCA
EXP.771 #708-840-4250
Dave Slimmer Fermilab
Computing Dept. #708-840-4334

SILICON STRIP DETECTOR SYSTEM "SINGLE BOARD DIAGNOSTICS TESTS"-PN434
Software Description
Version: 1 4/25/91

Originator:
Garry R. Moore Fermilab
Computing Dept. #708-840-4059

POSTAMP/COMPARATOR HARDWARE DESCRIPTION-HN100
Complete Module Documentation Manual
Version: 3 5/7/91

Originators:
Merle Haldeman Fermilab
RD/DEG group #708-840-3958
Scott Holm Fermilab
RD/DEG group #708-840-4340
Bruce Merkel Fermilab
RD/DEG group #708-840-3263

In order for the PC module to be accepted, the PC module must pass the following four tests.

1. PC_TEST

Refer to the "SINGLE BOARD DIAGNOSTICS TESTS" documentation listed above.

PC_TEST is a menu driven software tool capable of effectively testing and/or exercising all FASTBUS accessible circuitry of the POSTAMP/COMPARATOR module. The test should be run in the "Exercise Postamp Comparator" mode; this is #3 under the PC_TEST main menu. This option will fully exercise all FASTBUS accessible circuitry of the PC module including all geographical address, secondary address, and DAC/ADC circuitry.

ACCEPTANCE- The software will respond with errors if the module responds incorrectly. The module is unacceptable when any errors are reported.

EXCEPTIONS: A part of the test writes and then reads a series of IC-03 DAC values, the software will respond with an error if the DAC value read back is more than plus or minus one LSB different than the written value. The module however, will be accepted when the DAC value read back is within two LSB's of the written value.

Refer to the "SILICON STRIP DETECTOR (SSD)" documentation listed on page 1 for the remaining three tests.

2. PC test counter test

This test uses an 8-bit counter on the PC module to generate 256 patterns used for testing the output portion(IC-02 and IC-04 ASIC's) of the PC module. The 8-bit counter cycles through 256 output pattern possibilities which are sent to the IC-02 and IC-04 test inputs; t1 and t2. The outputs of the IC-02s and IC-04s provide 256 different 128-bit patterns which are sent to the DE module through the FASTBUS auxiliary backplane. This test effectively tests the outputs of all IC-02 and IC-04 ASIC's as well as their connections to the backplane. The DE module has 256 memory locations which are continually being loaded with the 256 pattern possibilities. In this test an individual DE memory location is always being loaded with the same pattern generated by the PC module test counter.(Refer to section 5.4.2 PC test counter test and to APPENDIX A for test counter patterns).

ACCEPTANCE- The PC module is acceptable if no errors occur. The software will know in which memory location in the DE module each pattern generated by PC module's IC-02 and IC-04 ASICs are stored. When a memory location in the DE module is read by the software, the 128-bit pattern in that location is compared with the 128-bit pattern that was expected for that location. An error is reported if the values do not match.

3. PC channel characterization tests

This test characterizes individual and sum channel threshold voltage sensitivity for the PC module discriminator electronics. Characterizing a channel involves placing a test signal at a channel input and then scanning through the threshold setting DAC, value range with the software recording the highest threshold voltage(DAC value) at which the channel discriminator was able to detect the input test signal. The test should be performed in the default mode(DAC vs CHANNEL mode). In this mode, five separate routines need to be run on the module. These routines are listed in a menu that is displayed after "PC channel characterization test"(h) is selected from the System test menu. Invoking any one of the routines, automatically sets up the PC module for that

routine, i.e. turning on individual channel or sum channel mode, and placing the PC module in the RUN mode.

The first routine, "INDIVIDUAL CHANNEL", tests all individual channels of the PC module. The software performs in the following way: the TSM module's 256 word by 128 bit memory(refer to TSM document#) is loaded with a pattern of alternating words consisting of all A's and 5's, which means adjacent bits in a given word have opposite values as well as a given bit in adjacent memory addresses. The TSM module feeds this pattern to the LS module(refer to LS module documentation#)which continually places this pattern at the PC module input. The DE continuously updates its 256 memory locations with the 128-bit pattern that the PC module outputs through the FASTBUS auxiliary backplane. At the start, all DACs are set to the highest value in their range(1.25v) and at some instance the software performs 8 nonconsecutive reads of a DE memory location, recording which if any of the 128 channels were on(hit) during each of the 8 reads. The DAC value is then lowered 1 LSB(5mv) and the DE module memory is read 8 more times, this continues until the DACs are at their lowest value in their range(0mv). A graph is made that shows the DAC values at which a channel responded to 8 out of 8 reads with the channel on(hit) and also when the channel responds at least once but less than 8 out of 8 times.

The remaining four routines test the SUM channels. Each routine uses a different pattern such as 1's, 2's, 4's or 8's, i.e. for the 2's pattern every 2nd channel of a 4 channel combination will be toggling on and for the 8's pattern every 4th channel of a 4 channel combination will be toggling on. A graph is made for each routine. In the SUM channel mode when placing an input signal on one channel the adjacent channels also receive this input signal, therefor the adjacent channels should also turn on and the graphs should reflect this.(Refer to section 5.4.6 PC Channel Characterization Tests).

ACCEPTANCE- The input signal from the LS module should be set for a 40mv peak to peak differential amplitude. The PC module passes the test if the resulting graphs that are produced by the software are as follows. ***{SHOW GRAPHS}***The DAC values of all channels may vary no more than plus or minus 15 counts from the norm, i.e. if the norm is at the DAC value of 90, acceptable DAC values range from 75 to 105. A module is unacceptable if a channel is characterized outside of the plus or minus 15 count range.

4. PC Crosstalk Test

PC Crosstalk test will test the ability of the PC module to reject crosstalk to other channels from an input signal of substantial amplitude on a certain other channel. The Crosstalk test uses the TSM and LS modules as in the PC characterization test. In this test the TSM memory is loaded with a 1 in one of the 128 bits of the first word and the remaining 255 words have a 0 in all 128 bits. This means that 1 channel is being stimulated with a 18.9ns pulse every 4.8us. The TSM module feeds this pattern to the LS module which continuously places the pattern at the PC module input, meanwhile the DE is continuously updating its 256 memory locations with the 128-bit pattern that the PC module outputs through the FASTBUS auxiliary backplane. Initially, all DACs are set to the highest value in their range(1.25v). The software reads the DE memory location containing the channel that is on(hit), the 128-bit value read, should contain only the one bit that is on. More than one bit on indicates crosstalk. The DAC value is then lowered 1 LSB(5mv) and the DE memory is read again, this repeats until the DAC output is reduced to 150mv. The software then shifts the pattern in the TSM memory one bit so that the next channel is being stimulated with a large amplitude signal, the process of reading the DE memory and lowering the DAC values is repeated. The shifting continues until all 128 channels have been stimulated. Crosstalk is reported each time a nonstimulated channel is on. This procedure shows that channels not responding to the

signals on other channels have less than 3% crosstalk.(Refer to section 5.4.7 PC Crosstalk test).

ACCEPTANCE-The signal amplitude of the input should be set to 500mv peak to peak differential amplitude. The PC module is unacceptable if crosstalk is found on any channel.

SCOTT HOLM 5/8/91

APPENDIX B - FASTBUS INTERFACE PAL PROGRAM

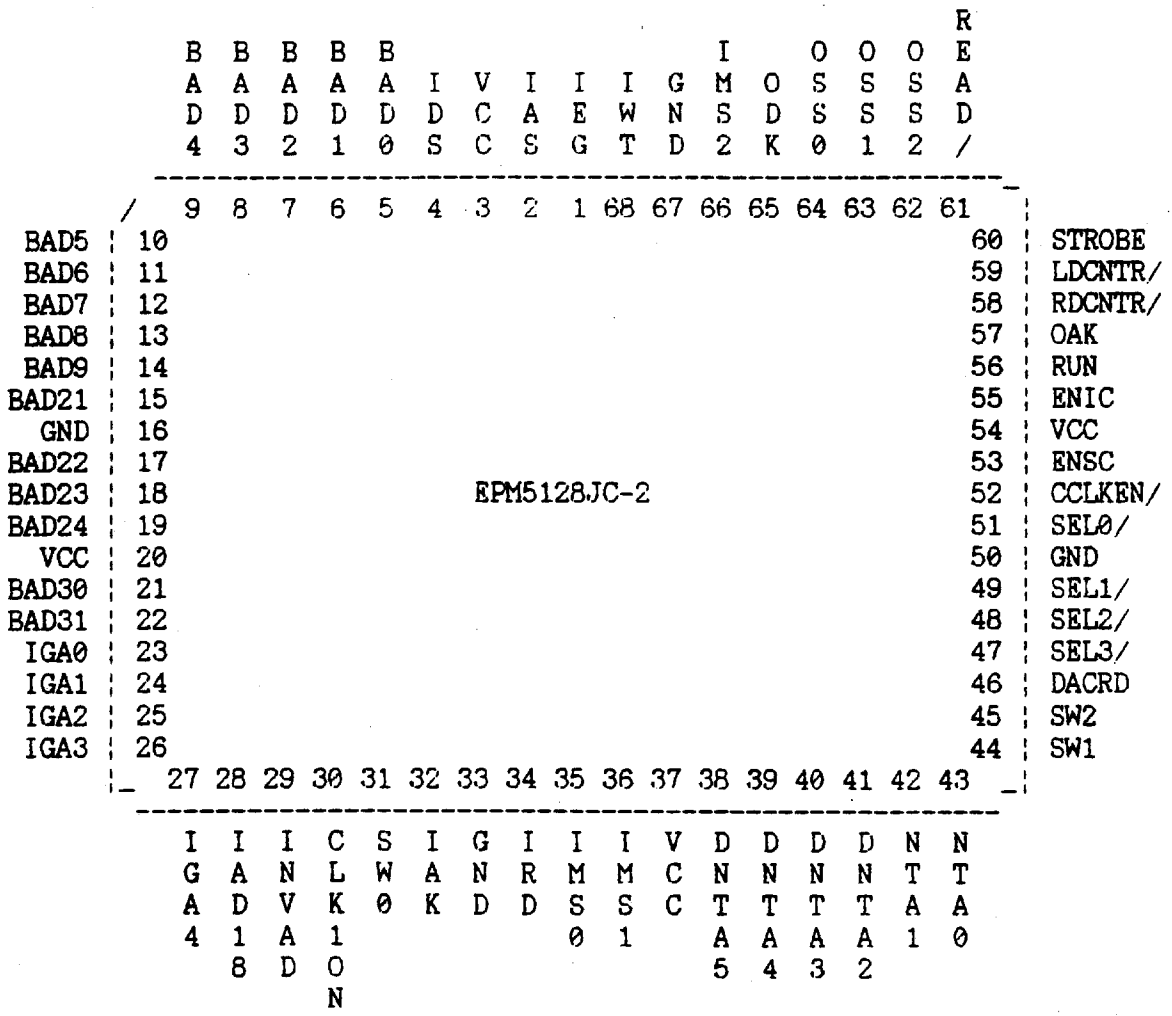
This appendix contains the information on the programming of the EPM5128 PAL that is used for the FASTBUS interface on the POSTAMP/COMPARATOR printed circuit board.

E:\KEN\SSD\PACMP\REVE\PACMPI.RPT

MAX+PLUS Compiler Report File
Version 2.50 05/31/90

** Design compiled without errors

Title: PA/CMP FASTBUS INTERFACE
Company: Fermilab
Designer: KEN TREPTOW
Rev: E
Date: 3:06p 11-15-1990
Turbo: ON
Security: OFF



** RESOURCE USAGE **

ic Array Block	Macrocells	I/O Pins	Expanders	External Interconnect
A: MC1 - MC16	15/16(93%)	8/ 8(100%)	27/32(84%)	19/24(79%)
B: MC17 - MC32	7/16(43%)	5/ 5(100%)	11/32(34%)	19/24(79%)
C: MC33 - MC48	4/16(25%)	5/ 5(100%)	1/32(3%)	6/24(25%)
D: MC49 - MC64	15/16(93%)	8/ 8(100%)	0/32(0%)	11/24(45%)
E: MC65 - MC80	16/16(100%)	8/ 8(100%)	15/32(46%)	23/24(95%)
F: MC81 - MC96	6/16(37%)	5/ 5(100%)	15/32(46%)	17/24(70%)
G: MC97 - MC112	7/16(43%)	5/ 5(100%)	3/32(9%)	24/24(100%)
H: MC113 - MC128	16/16(100%)	8/ 8(100%)	12/32(37%)	23/24(95%)

Total dedicated input pins used:	8/ 8 (100%)
Total I/O pins used:	52/ 52 (100%)
Total macrocells used:	86/128 (67%)
Total expanders used:	84/256 (32%)
Total input pins required:	20
Total output pins required:	24
Total bidirectional pins required:	16
Total macrocells required:	86
Total expanders in database:	69
Synthesized macrocells:	0/128 (0%)

** FILE HIERARCHY **

¦DACMAP:78¦
¦DACMAP:78¦74139:28¦
¦DACMAP:78¦74151:30¦
¦DACMAP:78¦74151:18¦
¦DACMAP:78¦74151:17¦
¦DACMAP:78¦74151:16¦
¦DACMAP:78¦74151:15¦
¦DACMAP:78¦74151:5¦
¦NTAREG:48¦
¦NTAREG:48¦DACSEL:108¦
¦NTAREG:48¦DACSEL:108¦74138H:11¦
¦NTAREG:48¦CT8:93¦
¦CONTROL:45¦
¦CONTROL:45¦DELAY:165¦
¦GASEL:46¦
¦OUTMUX:50¦
¦OUTMUX:50¦74153:81¦
¦OUTMUX:50¦74138H:82¦
¦OUTMUX:50¦74153:74¦
¦OUTMUX:50¦74153:75¦
¦OUTMUX:50¦74153:76¦
¦OUTMUX:50¦74153:77¦
¦OUTMUX:50¦74153:78¦
TMUX:50¦74153:79¦
TMUX:50¦74153:80¦
¦CSR0:49¦

** INPUTS **

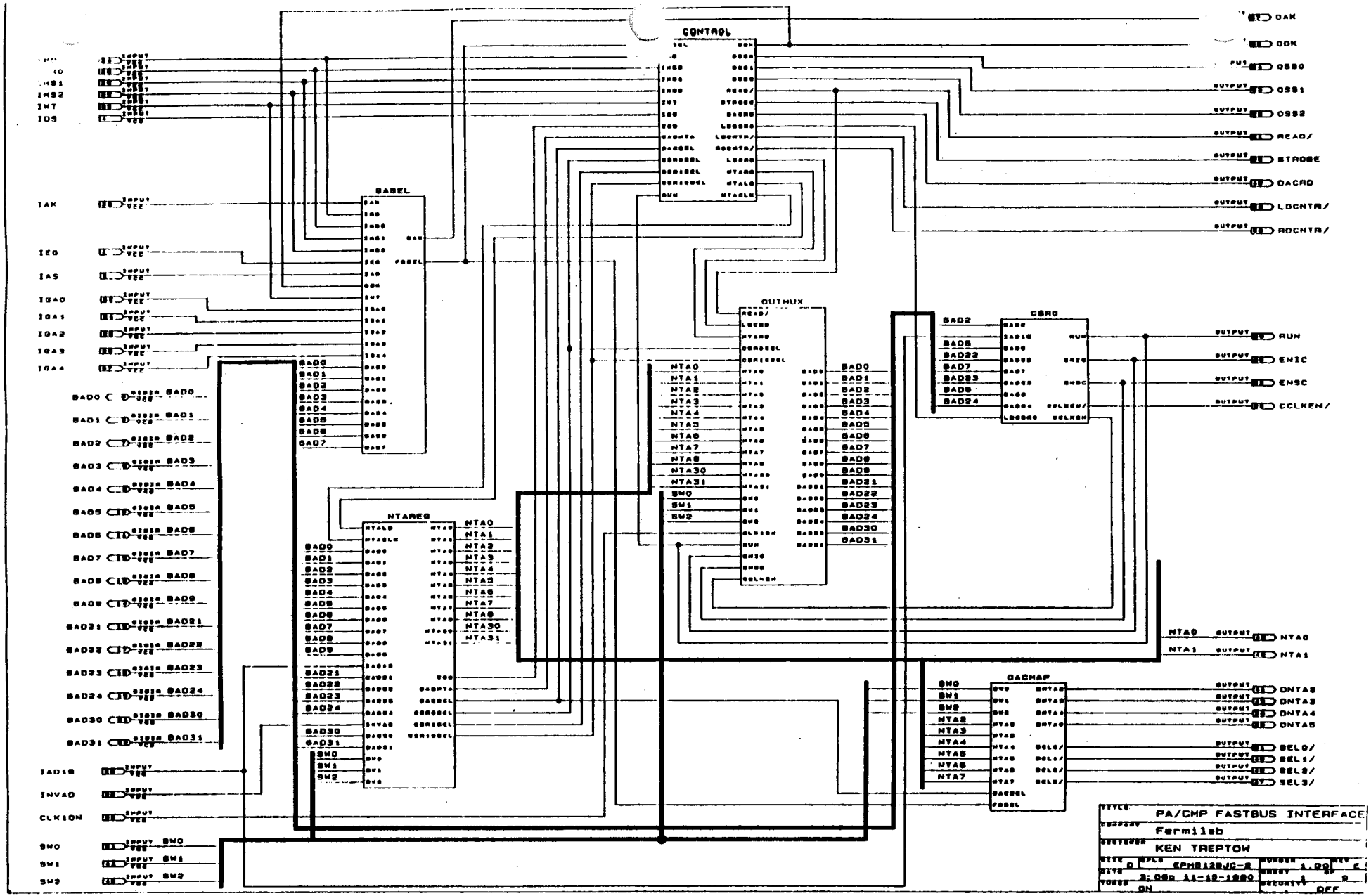
				Expanders		Fan-In		
1	MCell	LAB	Primitive	Total	Shared	INP	FBK	Name
5	(2)	(A)	INPUT	0	0	0	0	BAD0
6	(3)	(A)	INPUT	0	0	0	0	BAD1
7	(4)	(A)	INPUT	0	0	0	0	BAD2
8	(5)	(A)	INPUT	0	0	0	0	BAD3
9	(6)	(A)	INPUT	0	0	0	0	BAD4
10	(7)	(A)	INPUT	0	0	0	0	BAD5
11	(8)	(A)	INPUT	0	0	0	0	BAD6
12	(17)	(B)	INPUT	0	0	0	0	BAD7
13	(18)	(B)	INPUT	0	0	0	0	BAD8
14	(19)	(B)	INPUT	0	0	0	0	BAD9
15	(20)	(B)	INPUT	0	0	0	0	BAD21
17	(21)	(B)	INPUT	0	0	0	0	BAD22
18	(33)	(C)	INPUT	0	0	0	0	BAD23
19	(34)	(C)	INPUT	0	0	0	0	BAD24
21	(35)	(C)	INPUT	0	0	0	0	BAD30
22	(36)	(C)	INPUT	0	0	0	0	BAD31
30	(55)	(D)	INPUT	0	0	0	0	CLK1ON
28	(53)	(D)	INPUT	0	0	0	0	IAD18
32	-	-	INPUT	0	0	0	0	IAK
2	-	-	INPUT	0	0	0	0	IAS
4	(1)	(A)	INPUT	0	0	0	0	IDS
1	-	-	INPUT	0	0	0	0	IEG
23	(37)	(C)	INPUT	0	0	0	0	IGA0
24	(49)	(D)	INPUT	0	0	0	0	IGA1
25	(50)	(D)	INPUT	0	0	0	0	IGA2
6	(51)	(D)	INPUT	0	0	0	0	IGA3
27	(52)	(D)	INPUT	0	0	0	0	IGA4
35	-	-	INPUT	0	0	0	0	IMS0
36	-	-	INPUT	0	0	0	0	IMS1
66	-	-	INPUT	0	0	0	0	IMS2
29	(54)	(D)	INPUT	0	0	0	0	INVAD
34	-	-	INPUT	0	0	0	0	IRD
68	-	-	INPUT	0	0	0	0	IWT
31	(56)	(D)	INPUT	0	0	0	0	SW0
44	(71)	(E)	INPUT	0	0	0	0	SW1
45	(72)	(E)	INPUT	0	0	0	0	SW2

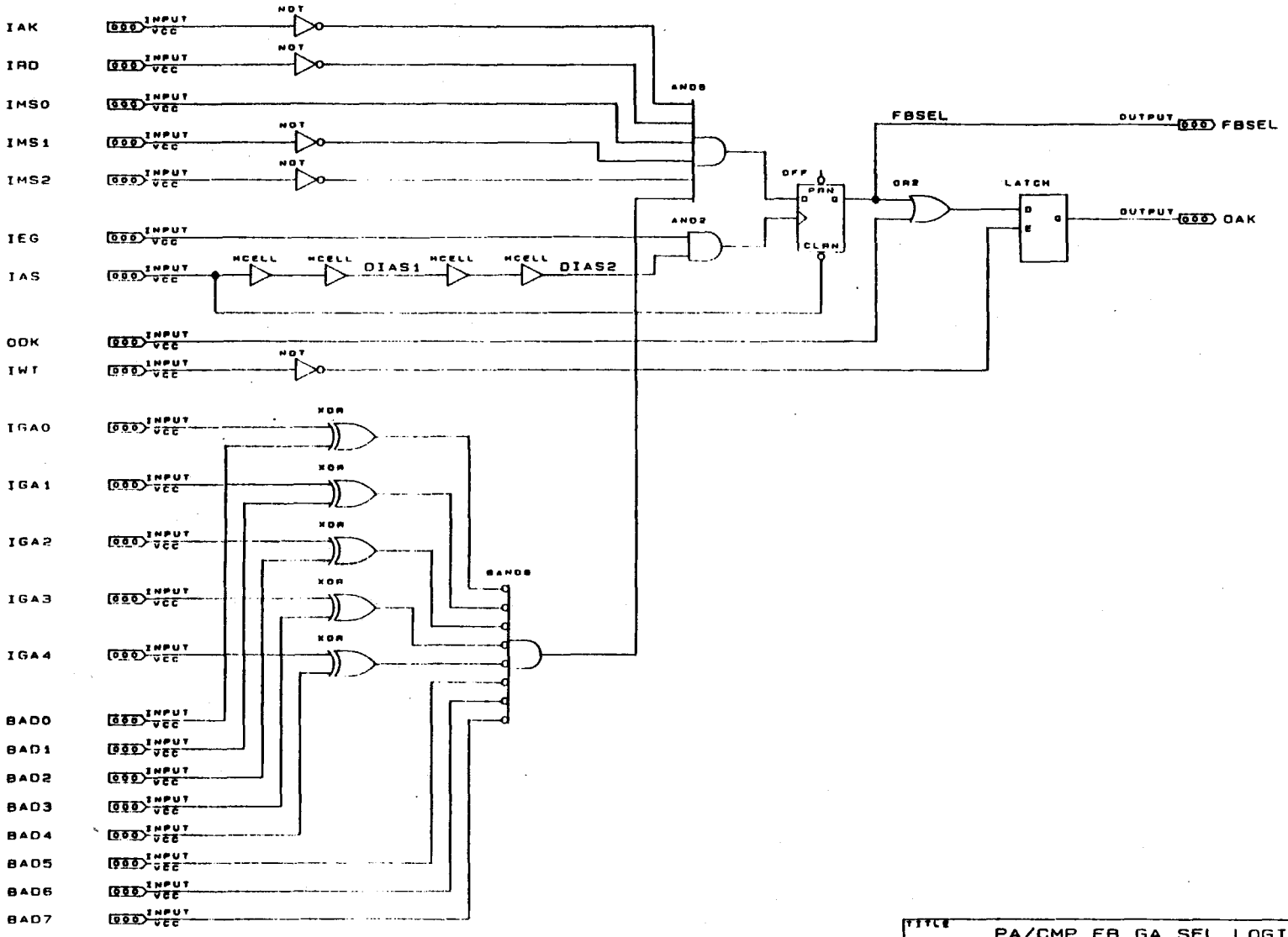
** OUTPUTS **

n	MCell	LAB	Primitive	Expanders Total	Expanders Shared	Fan-In INP	Fan-In FBK	Name
5	2	A	OR4	11	11	7	14	BAD0
6	3	A	OR4	11	11	7	14	BAD1
7	4	A	OR4	11	11	7	15	BAD2
8	5	A	OR4	11	11	7	14	BAD3
9	6	A	OR4	11	11	7	15	BAD4
10	7	A	OR4	11	11	7	14	BAD5
11	8	A	OR4	11	11	7	15	BAD6
12	17	B	OR4	11	11	4	15	BAD7
13	18	B	OR4	1	1	4	6	BAD8
14	19	B	OR4	1	1	5	4	BAD9
15	20	B	OR4	1	1	4	4	BAD21
17	21	B	OR4	1	1	4	2	BAD22
18	33	C	OR4	1	1	4	4	BAD23
19	34	C	OR4	1	1	4	4	BAD24
21	35	C	OR4	1	1	4	5	BAD30
22	36	C	OR4	1	1	4	5	BAD31
52	97	G	MCELL	0	0	0	1	CCLKEN/
46	81	F	OUTPUT	0	0	4	7	DACRD
41	68	E	OUTPUT	0	0	3	1	DNTA2
40	67	E	OUTPUT	0	0	3	2	DNTA3
39	66	E	OUTPUT	0	0	3	3	DNTA4
38	65	E	OUTPUT	5	0	3	4	DNTA5
55	99	G	DFF	3	3	6	15	ENIC
53	98	G	DFF	3	3	6	15	ENSC
59	114	H	OUTPUT	0	0	4	15	LDCNTR/
3	70	E	DFF	0	0	1	3	NTA0
2	69	E	DFF	0	0	1	4	NTA1
57	101	G	LATCH	0	0	1	2	OAK
65	120	H	LATCH	0	0	1	2	ODK
64	119	H	OUTPUT	0	0	3	3	OSS0
63	118	H	OUTPUT	8	7	4	11	OSS1
62	117	H	OUTPUT	7	7	4	10	OSS2
58	113	H	OUTPUT	0	0	3	13	RDCNTR/
61	116	H	OUTPUT	0	0	4	2	READ/
56	100	G	DFF	3	3	6	15	RUN
51	85	F	OUTPUT	5	0	3	11	SEL0/
49	84	F	OUTPUT	3	0	3	11	SEL1/
48	83	F	OUTPUT	4	0	3	11	SEL2/
47	82	F	OUTPUT	3	0	3	11	SEL3/
60	115	H	OUTPUT	0	0	3	10	STROBE

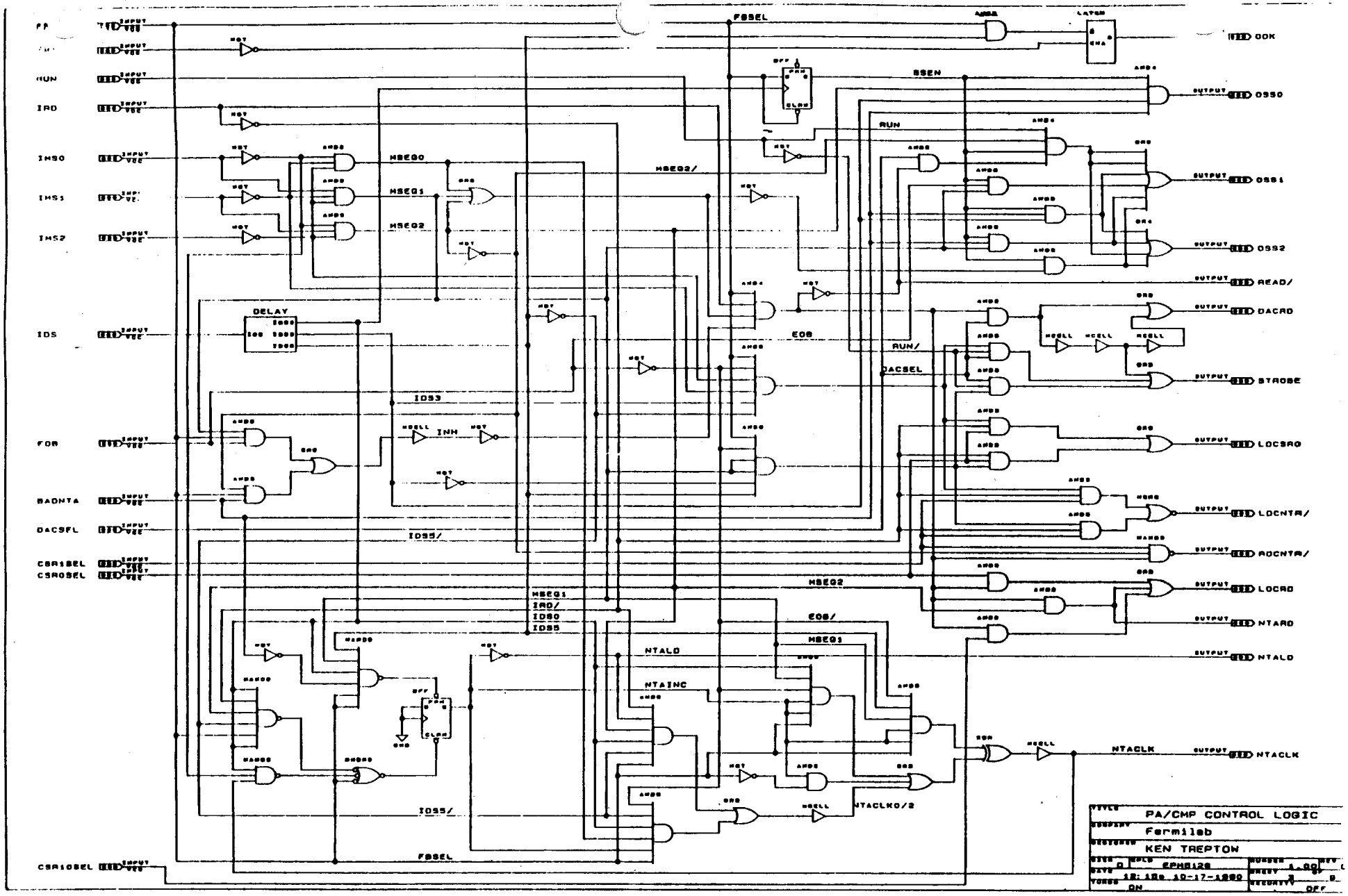
** BURIED LOGIC **

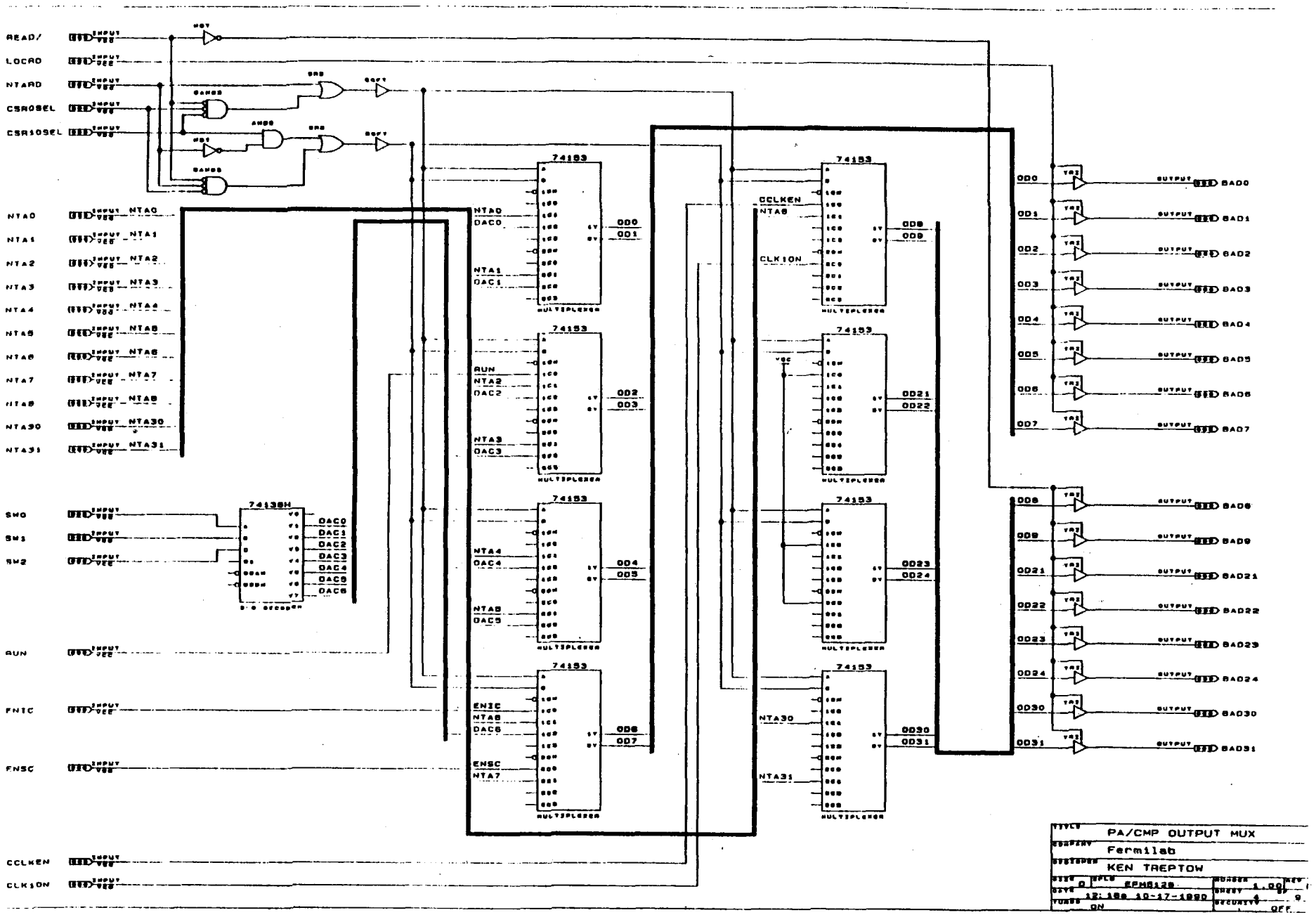
				Expanders		Fan-In		
↓	MCell	LAB	Primitive	Total	Shared	INP	FBK	Name
-	64	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS0
-	63	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS1
-	62	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS2
-	61	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS3
-	60	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS4
-	80	E	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS5
-	57	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:34
(27)	52	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:37
-	59	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:38
-	58	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:39
(31)	56	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:41
(30)	55	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:42
-	79	E	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:43
(29)	54	D	MCELL	0	0	1	0	|CONTROL:45|DELAY:165|:46
(26)	51	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:49
(25)	50	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:50
-	128	H	MCELL	0	0	3	3	|CONTROL:45|INH
-	78	E	MCELL	0	0	3	6	|CONTROL:45|NTACLK
-	77	E	MCELL	0	0	4	5	|CONTROL:45|NTACLK0/2
-	76	E	DFF	3	0	4	5	|CONTROL:45|NTAINC
-	96	F	DFF	0	0	0	2	|CONTROL:45|SSEN
-	16	A	MCELL	0	0	4	6	|CONTROL:45|:187
-	15	A	MCELL	0	0	0	1	|CONTROL:45|:188
-	14	A	MCELL	0	0	0	1	|CONTROL:45|:189
-	127	H	DFF	3	3	6	15	|CSR0:49|CCLKEN
-	13	A	MCELL	0	0	0	1	|GASEL:46|DIAS1
-	112	G	MCELL	0	0	0	1	|GASEL:46|DIAS2
-	75	E	DFF	7	0	20	1	|GASEL:46|FBSEL
-	12	A	MCELL	0	0	1	0	|GASEL:46|:21
-	111	G	MCELL	0	0	0	1	|GASEL:46|:51
-	126	H	MCELL	0	0	0	12	|NTAREG:48|BADNTA
-	31	B	DFF	0	0	1	5	|NTAREG:48|CT8:93|Q2
-	74	E	DFF	0	0	1	6	|NTAREG:48|CT8:93|Q3
-	73	E	DFF	0	0	1	7	|NTAREG:48|CT8:93|Q4
(45)	72	E	DFF	0	0	1	8	|NTAREG:48|CT8:93|Q5
(44)	71	E	DFF	0	0	1	9	|NTAREG:48|CT8:93|Q6
-	125	H	DFF	0	0	1	10	|NTAREG:48|CT8:93|Q7
-	11	A	SOFT	7	0	3	6	|NTAREG:48|DACSEL:108|:52
-	10	A	SOFT	7	0	3	8	|NTAREG:48|DACSEL:108|:53
-	124	H	MCELL	0	0	0	13	|NTAREG:48|EN/
-	123	H	DFF	1	0	0	15	|NTAREG:48|EOB
(28)	53	D	DFF	0	0	8	2	|NTAREG:48|NTA8
-	122	H	DFF	0	0	1	2	|NTAREG:48|NTA30
-	121	H	DFF	0	0	1	2	|NTAREG:48|NTA31
-	9	A	SOFT	2	0	4	13	|OUTMUX:50|:90
-	32	B	SOFT	0	0	4	12	|OUTMUX:50|:91

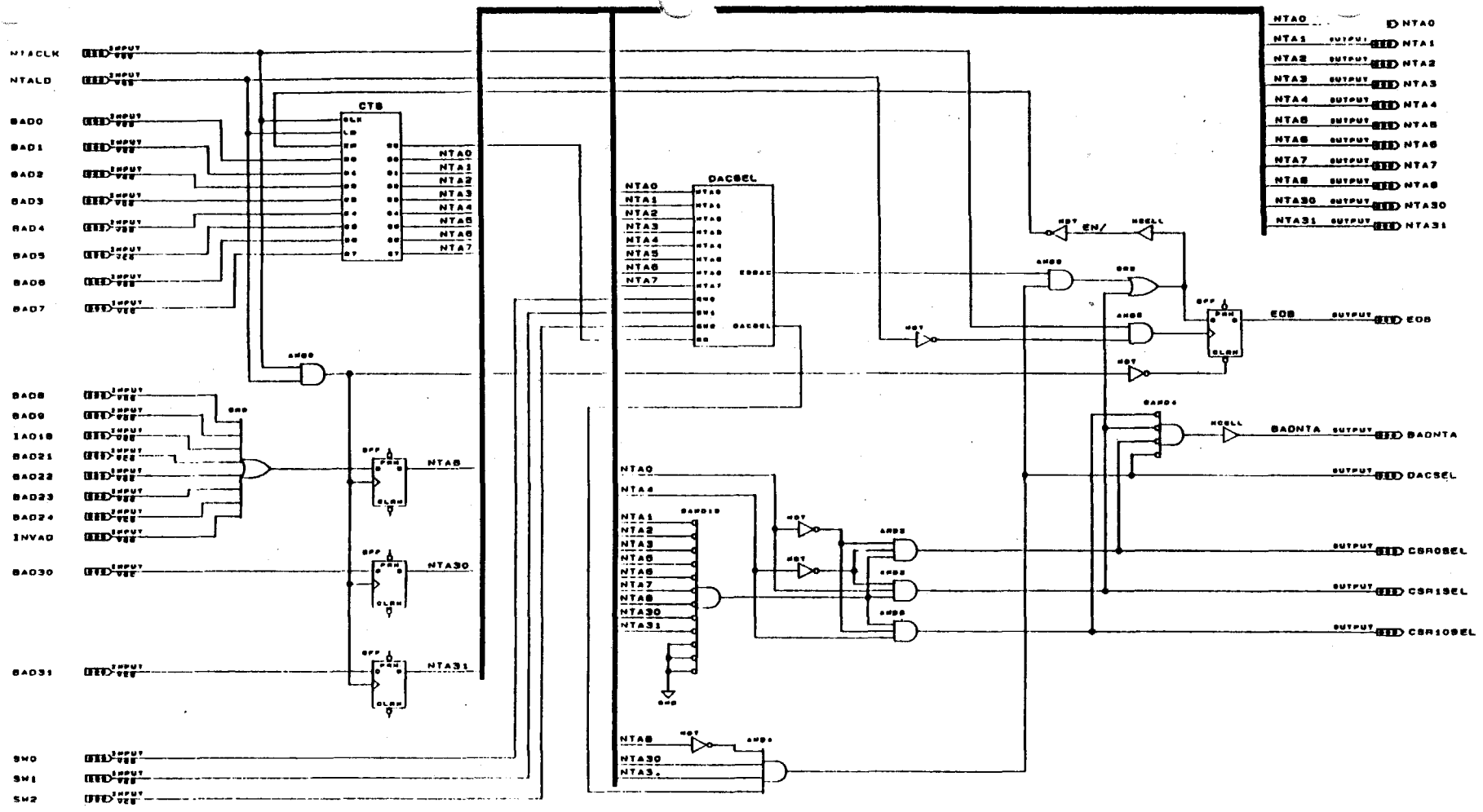

CONTROL
OUTMUX
CSRO
DACMAP
PA/CMP FASTBUS INTERFACE
Fermilab
KEN TREPTOW



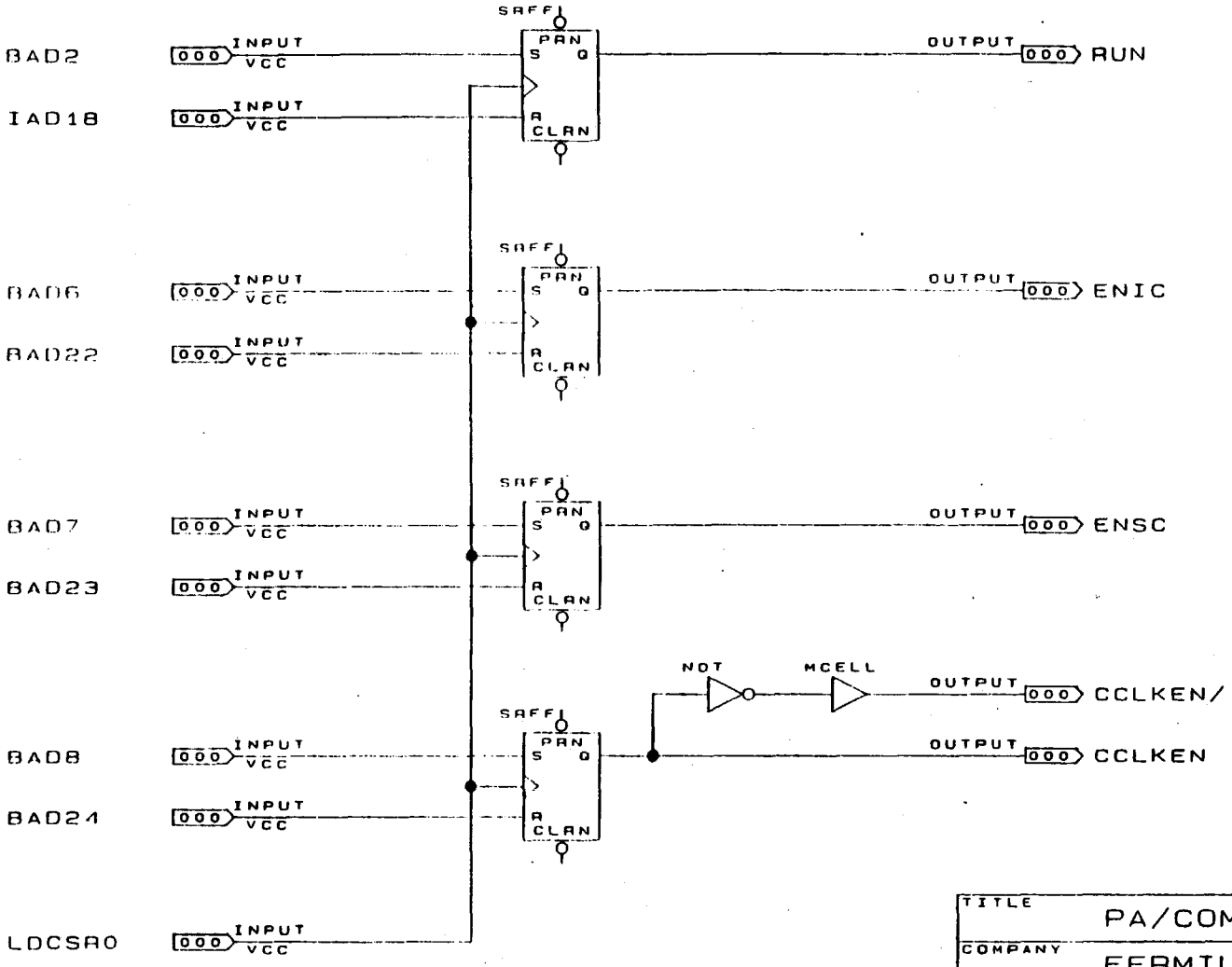
TITLE	PA/CMP FB GA SEL LOGIC
COMPANY	FERMILAB
DESIGNER	KEN TREPTOW
SIZE C / EPLD EPM5128 / NUMBER 1.00 / REV B	
DATE 6/11/90 / SHEET 4 OF 9	
TYPIST QN / SECURITY OFF	


RUN
IRD
IHS0
IHS1
IHS2
IDS
FDR
BADNTA
DACSEL
CSR1SEL
CSR0SEL
CSR10SEL
DELAY
INH
MSEQ0
MSEQ1
MSEQ2
MSEQ2/
IDS3
IDS5/
FBSEL
EOS
EOS/
RUN/
BSEN
NTALD
NTAINC
NTACLK0/2
NTACLK
OOK
OSS0
OSS1
OSS2
READ/
DACRD
STROBE
LOCSRD
LOCNTR/
ROCNTR/
LOCRD
NTARD
NTALD
NTACLK
PA/CMP CONTROL LOGIC
Fermilab
KEN TREPTON
EPM5128
12:12a 10-17-1990

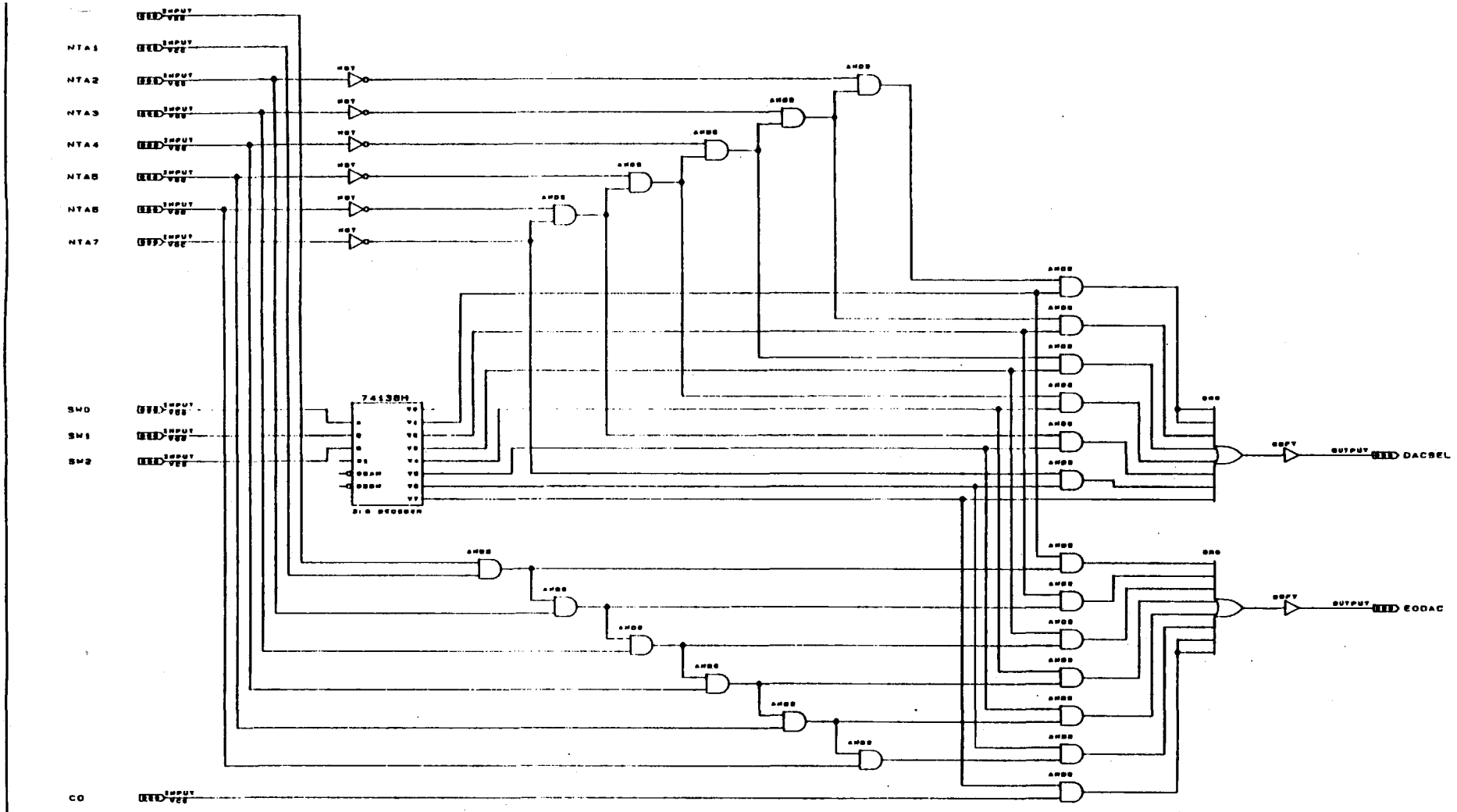

READ/
LOCRD
NTARD
CSR0SEL
CSR1SEL
NTA0
NTA1
NTA2
NTA3
NTA4
NTA5
NTA6
NTA7
NTA8
NTA30
NTA31
SW0
SW1
SW2
RUN
FNIC
FNSC
CCLKEN
CLK1ON
74138H
74153
MULTIPLEXER
DAC0
DAC1
DAC2
DAC3
DAC4
DAC5
DAC6
ENIC
ENSC
OD0
OD1
OD2
OD3
OD4
OD5
OD6
OD7
OD21
OD22
OD23
OD24
OD30
OD31
BAD0
BAD1
BAD2
BAD3
BAD4
BAD5
BAD6
BAD7
BAD8
BAD9
BAD21
BAD22
BAD23
BAD24
BAD30
BAD31
PA/CMP OUTPUT MUX
Fermilab
KEN TREPTOW
EPM5128
12:18a 10-17-1990
ON
OFF



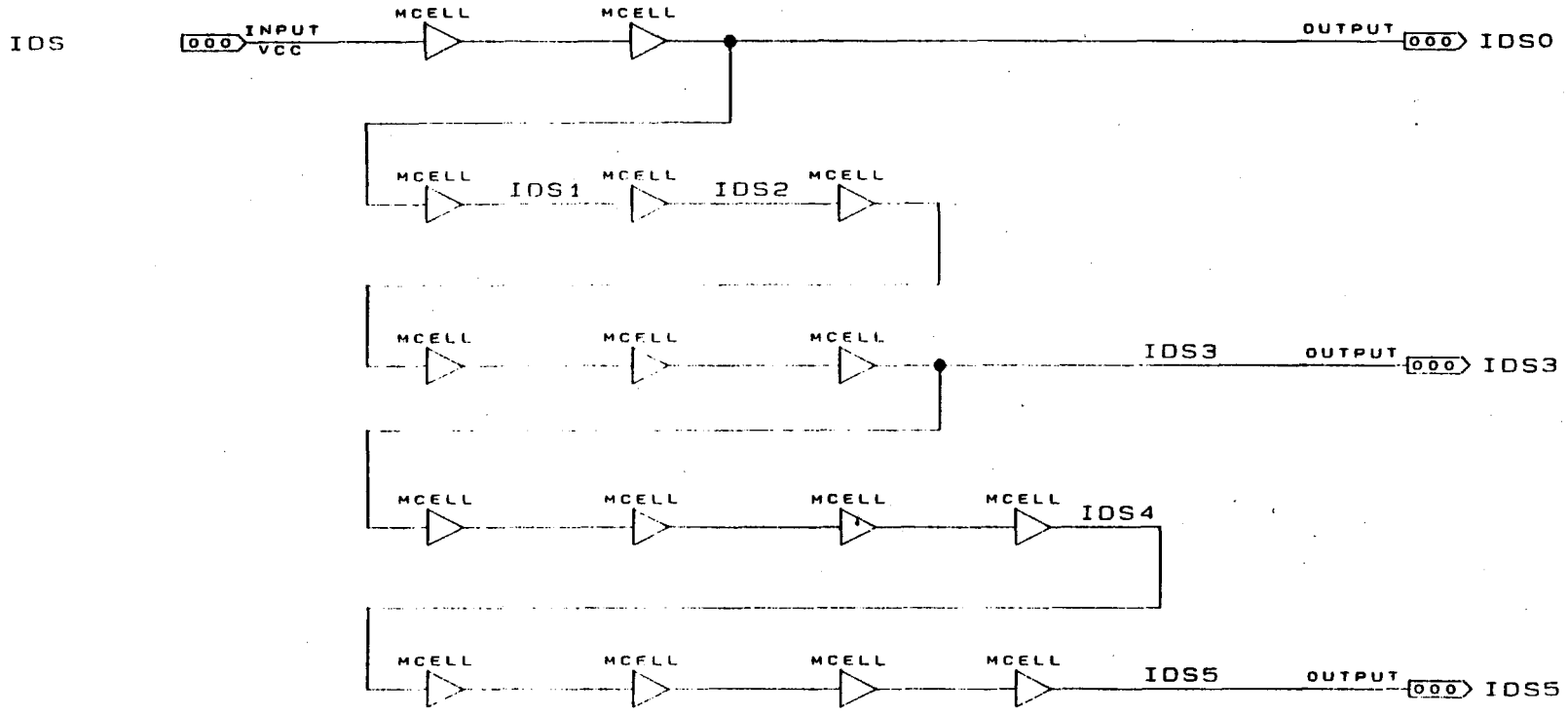
TITLE	PA/CMP NTA REGISTER
COMPANY	Fermilab
DESIGNER	KEN TAEPTOW
STEP D	DPLS FPM8128
DATE	12:14A 10-17-1990
TURBO	ON
NUMBER	A.00 REV
SHEET	OF
SECURITY	OFF



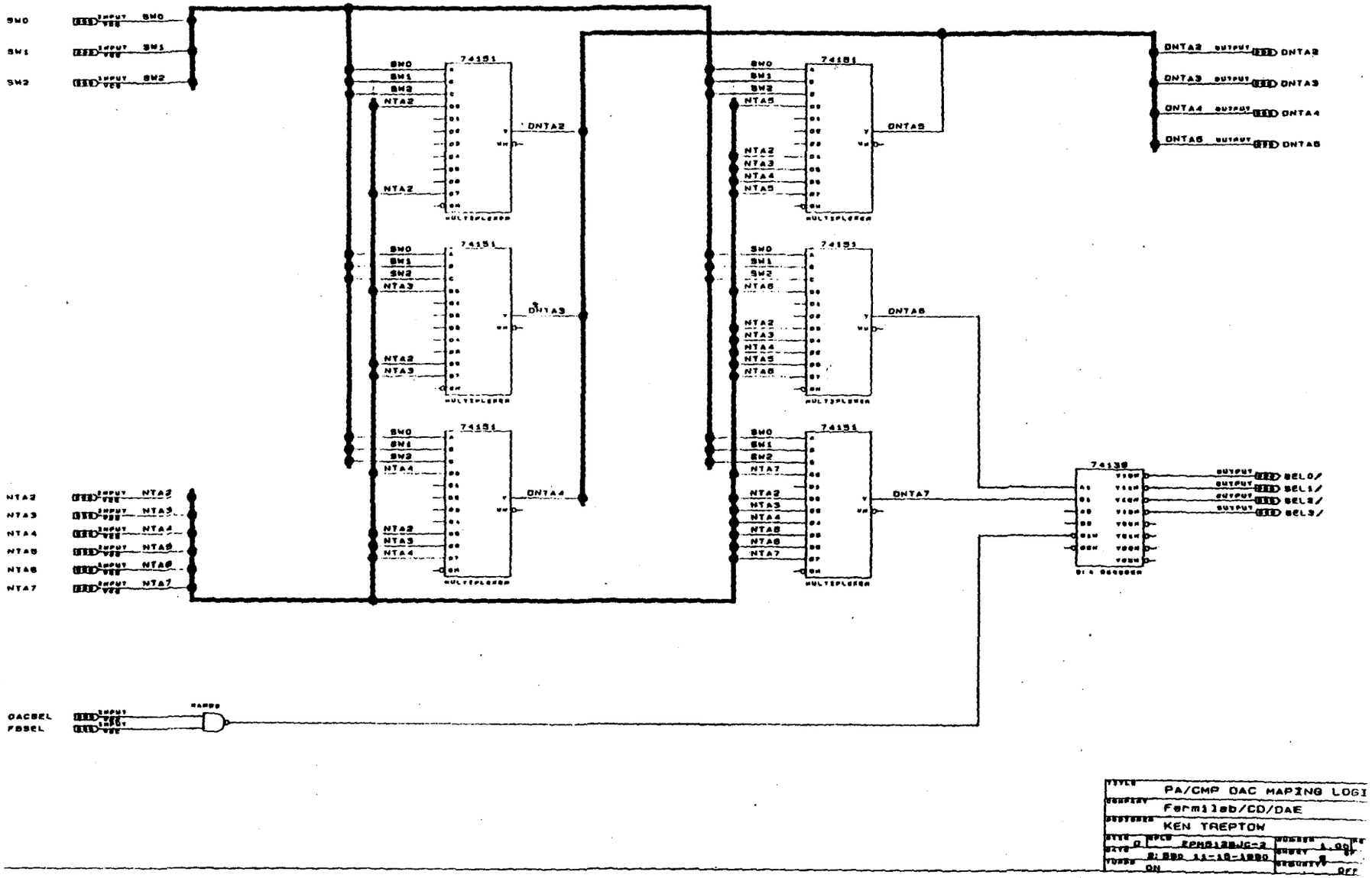
TITLE	PA/COMP CSR0		
COMPANY	FERMILAB/CD/DAE		
DESIGNER	KEN TREPTOW		
SIZE B	EPLD EPM5128	NUMBER 1.00	REV B
DATE 6/26/90		SHEET 6 OF 9	
TURBO ON		SECURITY OFF	



TITLE	PA/CMP DAC ADDR DECODE
COMPANY	Fermilab
DESIGNER	KEN TREPTOW
SIZE D	APLD EPM5128
DATE	12:17a 10-17-1990
TURBO	ON
NUMBER	1.00
SHEET	7 OF 9
SECURITY	OFF



TITLE	PA/CMP DS DELAYS		
COMPANY	Fermilab		
DESIGNER	KEN TREPTOW		
SIZE B	EPLD EPM5128	NUMBER 1.00	REV A
DATE 10:22a 11-13-1990		SHEET 8 OF 9	
TURBO ON		SECURITY OFF	



TITLE	PA/CMP DAC MAPING LOGI
COMPANY	Fermilab/CD/DAE
AUTHOR	KEN TREPTOW
SIZE D	MPLD EPM5128JC-2 NUMBER REV 1.00
DATE	D: SUN 11-15-1992 SHEET OF
TIME	ON DRAWING NO. REF

3.5 HARDWARE JUMPERS

3.5.1 DAC/ADC Jumpers

The illustration is an abbreviated pattern of the IC-03 DAC/ADC jumpering scheme which exists on the P/C Module.
There are 256 DACs which are addressed via CSR addresses C000_0000 through C000_00FF inclusive. DAC address 00 (Hex), will always set the threshold voltage on the uppermost trace which is the threshold for the discriminator connected to the output of the summed channel which takes the sum of channel -1 and channel 0 (Σ-1,0). Likewise DAC address 01 (Hex), will always set the threshold voltage on the trace immediately below the uppermost trace which is the threshold for the discriminator connected directly to the individual channel #0 input. DAC address 02 (Hex), will always set the threshold voltage for the discriminator connected to the output of the summed channel which takes the sum of channel 0 and channel 1 (Σ0,1). DAC address 03 (Hex), will always set the threshold voltage for the discriminator connected directly to the individual channel #1 input. This pattern continues for the 256 DACs and the 256 discriminator threshold setting inputs.
It should be obvious that if all 256 DAC's are employed, no jumpers are required and each discriminator has an individual DAC to control it's threshold. However, in the event that only 4 of the 256 DAC's are mounted on the P/C Module, the jumpers would probably be applied such that DAC 00 (hex) would drive every other sum channel. i.e., Σ(0), Σ(1,2), Σ(3,4), Σ(5,6) etc. DAC 01 (hex) would drive the even individual channels; DAC 02 (hex) would drive sum channels Σ(0,1), Σ(2,3), Σ(4,5), Σ(6,7) etc., and DAC 03 (hex) would drive the odd, individual channels.
The common choices of DAC populations are 4,8,16,32,64,128 or 256 which corresponds to 1,2,4,8,16,32 and 64 IC-03 packages. It seems reasonable to assume that for four DAC's every fourth threshold lead would be connected together and so on. However, any and all possibilities exist for connecting DAC outputs to threshold inputs and the method used may be dictated by the detector geometry.

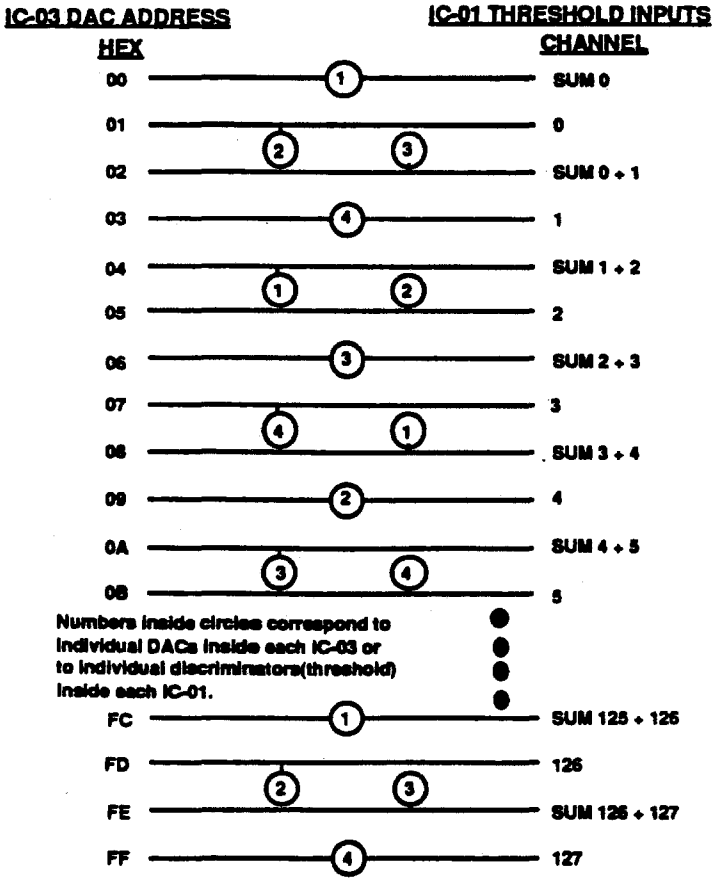


Figure 21: DAC/ADC Jumpering

3.5.2 CSR10 Jumpers

The value found in CSR10 corresponds to the configuration of three, two position jumpers(JU2, JU3, and JU4) that connect a signal trace to ground(GND) or +5V. The physical module location of the three posts for each jumper, and their connections are shown below. Table 1 on the following page provides the relationship between DAC population, jumper location, CSR10 readback and lowest ordered address for each DAC chip.
When less than 256 DAC's are utilized, the P/C module will be populated from the lowest address onward without skipping addresses, this is done by a mapping routine within the EPM5128 programmable PAL. It has been done this way to make the DACs accessible by block transfer. The PAL receives a 3 bit code from the jumpers to indicate the number of IC-03 DAC/ADC chips mounted and coverts the sequential DAC address it receives from an external source to the appropriate DAC module address. I.E. if only two IC-03s(8 DACs) are used and located at DAC board address 0 through 3(U97) and DAC board address 64 through 67(U129), the PAL knowing from the jumpering scheme will only accept DAC addresses 0 through 7 from an external source and in turn converts external address 0 to DAC board address 0 and so on up to external address 7 to DAC board address 67. When less than 256 DACs are employed the IC-03s physical location is important to insure that the external address received is converted to the appropriate DAC board address. The physical locations for the common number of IC-03s used are shown in Table 2 on the following page.

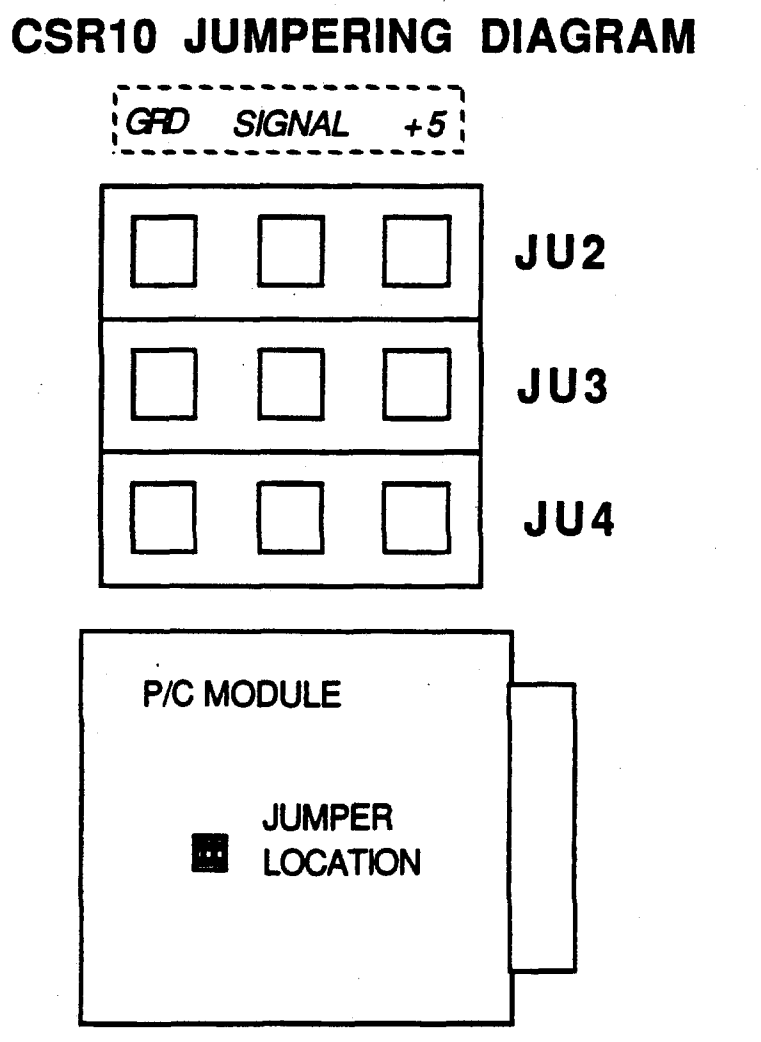


Figure 22: CSR10 Jumpering

Table 1

Jumper Numbers			# of DACs	CSR10 Value	First DAC board address of each IC-03
JU2	JU3	JU4			
Gnd	Gnd	Gnd	????	0	This is a special case, unique to the user.
Gnd	Gnd	+5V	4	1	0
Gnd	+5V	Gnd	8	2	0,80
Gnd	+5V	+5V	16	4	0,40,80,C0
+5V	Gnd	Gnd	32	8	0,20,40,60,80,A0,C0,E0
+5V	Gnd	+5 V	64	10	0,10,20,30,40,50,60,70,80,90,A0,B0,C0,D0,E0,F0
+5V	+5V	Gnd	128	20	0,8,10,18,20,28,30,38,40,48,50,58,60,68,70,7880,88,90,98,A0,A8,B0,B8,C0,C8,D0,D8,E0,E8, F0,F8
+5V	+5V	+5V	256	40	All addresses from 00 through FF inclusive are valid

Table 2

Number of IC-03s mounted(#DACs)	IC-03 Physical location on module
1(4)	U97
2(8)	U97, U129
4(16)	U97, U113, U129, U145
8(32)	U97, U105, U113, U121, U129, U137, U145, U153
16(64)	U97, U101, U105, U109, U113, U117, U121, U125, U129, U133, U137, U141, U145, U149, U153, U157
32(128)	U97, U99, U101, U103, U105, U107, U109, U111, U113, U115, U117, U119, U121, U123, U125, U127, U129, U131, U133, U135, U137, U139, U141, U143, U145, U147, U149, U151, U153, U155, U157, U159
64(256)	ALL LOCATIONS

4.0 P/C MODULE - TESTING AND CALIBRATION

4.1 UNPOWERED TEST

A test should be performed to check for shorts between the power planes (layers) of the P/C module. The test should verify no direct connection between any of the following power planes: -5.2V, -3.5V, -2V, -.8V top, -.8V bottom, +5V, +3.5V, +1.25V, and ground. The test can be done with an OHM METER. Connection to the power plane can be found at the following locations, with the measured impedances between the power and ground plane with the PC board fully assembled(number of IC-03s equals four).

-5.2V = right side of D5	= 8 ohm
-3.5V = right side of fuse f4	= 3 ohm
-2V = right side of fuse f1	= 22 ohm
-.8V top = right side of CR2	= 23 ohm
-.8V bottom = right side of CR4	= 23 ohm
+5V = right side of fuse f6	= 400 ohm(90 ohm with EPM5128 installed)
153 = right side of fuse f2	= 580 ohm
+1.25V = case of Q1	= 73 ohm
GROUND = left side of D5	

NOTE: Right side is towards the front of the board where the LABEL is located. The drawing "POWER TEST POINT DETAILS"#2563.000-MD-215808 shows the power pin locations of various devices on the top side of the board, this should help in finding shorts.

4.2 AUTOMATED TESTING

Their is an "Acceptance Test Description" document that uses two software packages to verify wether a P/C module is acceptable from an outside vendor. This document can be found in APPENDIX A.

4.3 CALIBRATION

Adjusting the value at which the ***TEMP*** LED on the front panel of the P/C module will turn on is the only calibration needed for the module. Adjusting trim pot R295 sets the temperature at which the LED will turn on and off, the setting of R295 controls the voltage on pin 2 of U185. If pin 3 of U185 which is connected to the LM35 temperature sensor is at a more positive value than pin 2 of U185 the LED will turn on. The voltage on pin 2 should be set to 0.5V this corresponds to 50 degrees Celsius. The LM35 will output to pin 3 of the U185 a 10 mV per degree Celsius voltage level. If the temperature rises above 50 degrees Celsius(0.5V on pin 3) the LED will turn on. The LED will turn off if the temperature falls below 50 degrees Celsius. The ***TEMP*** LED and associated circuitry can be tested by adjusting R295 for a voltage on pin 2 of U185 that is less than the value on pin 3 of U185, the lower voltage should cause the ***TEMP*** LED to turn on.

5.0 APPENDICES

APPENDIX A - TEST SOFTWARE
APPENDIX B - FASTBUS INTERFACE PAL PROGRAM
APPENDIX C - POSTAMP/COMPARATOR MODULE - SCHEMATIC
APPENDIX D - MODULE DRAWINGS
APPENDIX E - DATA SHEETS

APPENDIX A - TEST SOFTWARE

This appendix contains an acceptance criteria specification that pertains to the testing of the POSTAMP/COMPARATOR printed circuit board.

POSTAMP/COMPARATOR MODULE - ACCEPTANCE TEST DESCRIPTION

The party performing the acceptance test on the POSTAMP/COMPARATOR module should be familiar with the "IEEE Standard FASTBUS Modular High-Speed Data Acquisition and Control system"(ANSI/IEEE Std. 960-1986), the POSTAMP/COMPARATOR HARDWARE DESCRIPTION-HN100, and the two software packages listed below which the acceptance criteria is based upon. Questions pertaining to the documents should be addressed to the appropriate originator.

SILICON STRIP DETECTOR (SSD)
System Test Software Guide
Version: 0 15 march 1991

Originators:
Wolfgang Kowald Duke University
EXP.771 #708-840-4250
Panagiotis Spentzouris UCA
EXP.771 #708-840-4250
Dave Slimmer Fermilab
Computing Dept. #708-840-4334

SILICON STRIP DETECTOR SYSTEM "SINGLE BOARD DIAGNOSTICS TESTS"-PN434
Software Description
Version: 1 4/25/91

Originator:
Garry R. Moore Fermilab
Computing Dept. #708-840-4059

POSTAMP/COMPARATOR HARDWARE DESCRIPTION-HN100
Complete Module Documentation Manual
Version: 3 5/7/91

Originators:
Merle Haldeman Fermilab
RD/DEG group #708-840-3958
Scott Holm Fermilab
RD/DEG group #708-840-4340
Bruce Merkel Fermilab
RD/DEG group #708-840-3263

In order for the PC module to be accepted, the PC module must pass the following four tests.

1. PC_TEST

Refer to the "SINGLE BOARD DIAGNOSTICS TESTS" documentation listed above.

PC_TEST is a menu driven software tool capable of effectively testing and/or exercising all FASTBUS accessible circuitry of the POSTAMP/COMPARATOR module. The test should be run in the "Exercise Postamp Comparator" mode; this is #3 under the PC_TEST main menu. This option will fully exercise all FASTBUS accessible circuitry of the PC module including all geographical address, secondary address, and DAC/ADC circuitry.

ACCEPTANCE- The software will respond with errors if the module responds incorrectly. The module is unacceptable when any errors are reported.

EXCEPTIONS: A part of the test writes and then reads a series of IC-03 DAC values, the software will respond with an error if the DAC value read back is more than plus or minus one LSB different than the written value. The module however, will be accepted when the DAC value read back is within two LSB's of the written value.

Refer to the "SILICON STRIP DETECTOR (SSD)" documentation listed on page 1 for the remaining three tests.

2. PC test counter test

This test uses an 8-bit counter on the PC module to generate 256 patterns used for testing the output portion(IC-02 and IC-04 ASIC's) of the PC module. The 8-bit counter cycles through 256 output pattern possibilities which are sent to the IC-02 and IC-04 test inputs; t1 and t2. The outputs of the IC-02s and IC-04s provide 256 different 128-bit patterns which are sent to the DE module through the FASTBUS auxiliary backplane. This test effectively tests the outputs of all IC-02 and IC-04 ASIC's as well as their connections to the backplane. The DE module has 256 memory locations which are continually being loaded with the 256 pattern possibilities. In this test an individual DE memory location is always being loaded with the same pattern generated by the PC module test counter.(Refer to section 5.4.2 PC test counter test and to APPENDIX A for test counter patterns).

ACCEPTANCE- The PC module is acceptable if no errors occur. The software will know in which memory location in the DE module each pattern generated by PC module's IC-02 and IC-04 ASICs are stored. When a memory location in the DE module is read by the software, the 128-bit pattern in that location is compared with the 128-bit pattern that was expected for that location. An error is reported if the values do not match.

3. PC channel characterization tests

This test characterizes individual and sum channel threshold voltage sensitivity for the PC module discriminator electronics. Characterizing a channel involves placing a test signal at a channel input and then scanning through the threshold setting DAC, value range with the software recording the highest threshold voltage(DAC value) at which the channel discriminator was able to detect the input test signal. The test should be performed in the default mode(DAC vs CHANNEL mode). In this mode, five separate routines need to be run on the module. These routines are listed in a menu that is displayed after "PC channel characterization test"(h) is selected from the System test menu. Invoking any one of the routines, automatically sets up the PC module for that

routine, i.e. turning on individual channel or sum channel mode, and placing the PC module in the RUN mode.

The first routine, "INDIVIDUAL CHANNEL", tests all individual channels of the PC module. The software performs in the following way: the TSM module's 256 word by 128 bit memory(refer to TSM document#) is loaded with a pattern of alternating words consisting of all A's and 5's, which means adjacent bits in a given word have opposite values as well as a given bit in adjacent memory addresses. The TSM module feeds this pattern to the LS module(refer to LS module documentation#)which continually places this pattern at the PC module input. The DE continuously updates its 256 memory locations with the 128-bit pattern that the PC module outputs through the FASTBUS auxiliary backplane. At the start, all DACs are set to the highest value in their range(1.25v) and at some instance the software performs 8 nonconsecutive reads of a DE memory location, recording which if any of the 128 channels were on(hit) during each of the 8 reads. The DAC value is then lowered 1 LSB(5mv) and the DE module memory is read 8 more times, this continues until the DACs are at their lowest value in their range(0mv). A graph is made that shows the DAC values at which a channel responded to 8 out of 8 reads with the channel on(hit) and also when the channel responds at least once but less than 8 out of 8 times.

The remaining four routines test the SUM channels. Each routine uses a different pattern such as 1's, 2's, 4's or 8's, i.e. for the 2's pattern every 2nd channel of a 4 channel combination will be toggling on and for the 8's pattern every 4th channel of a 4 channel combination will be toggling on. A graph is made for each routine. In the SUM channel mode when placing an input signal on one channel the adjacent channels also receive this input signal, therefor the adjacent channels should also turn on and the graphs should reflect this.(Refer to section 5.4.6 PC Channel Characterization Tests).

ACCEPTANCE- The input signal from the LS module should be set for a 40mv peak to peak differential amplitude. The PC module passes the test if the resulting graphs that are produced by the software are as follows. ***{SHOW GRAPHS}***The DAC values of all channels may vary no more than plus or minus 15 counts from the norm, i.e. if the norm is at the DAC value of 90, acceptable DAC values range from 75 to 105. A module is unacceptable if a channel is characterized outside of the plus or minus 15 count range.

4. PC Crosstalk Test

PC Crosstalk test will test the ability of the PC module to reject crosstalk to other channels from an input signal of substantial amplitude on a certain other channel. The Crosstalk test uses the TSM and LS modules as in the PC characterization test. In this test the TSM memory is loaded with a 1 in one of the 128 bits of the first word and the remaining 255 words have a 0 in all 128 bits. This means that 1 channel is being stimulated with a 18.9ns pulse every 4.8us. The TSM module feeds this pattern to the LS module which continuously places the pattern at the PC module input, meanwhile the DE is continuously updating its 256 memory locations with the 128-bit pattern that the PC module outputs through the FASTBUS auxiliary backplane. Initially, all DACs are set to the highest value in their range(1.25v). The software reads the DE memory location containing the channel that is on(hit), the 128-bit value read, should contain only the one bit that is on. More than one bit on indicates crosstalk. The DAC value is then lowered 1 LSB(5mv) and the DE memory is read again, this repeats until the DAC output is reduced to 150mv. The software then shifts the pattern in the TSM memory one bit so that the next channel is being stimulated with a large amplitude signal, the process of reading the DE memory and lowering the DAC values is repeated. The shifting continues until all 128 channels have been stimulated. Crosstalk is reported each time a nonstimulated channel is on. This procedure shows that channels not responding to the

signals on other channels have less than 3% crosstalk.(Refer to section 5.4.7 PC Crosstalk test).

ACCEPTANCE-The signal amplitude of the input should be set to 500mv peak to peak differential amplitude. The PC module is unacceptable if crosstalk is found on any channel.

SCOTT HOLM 5/8/91

APPENDIX B - FASTBUS INTERFACE PAL PROGRAM

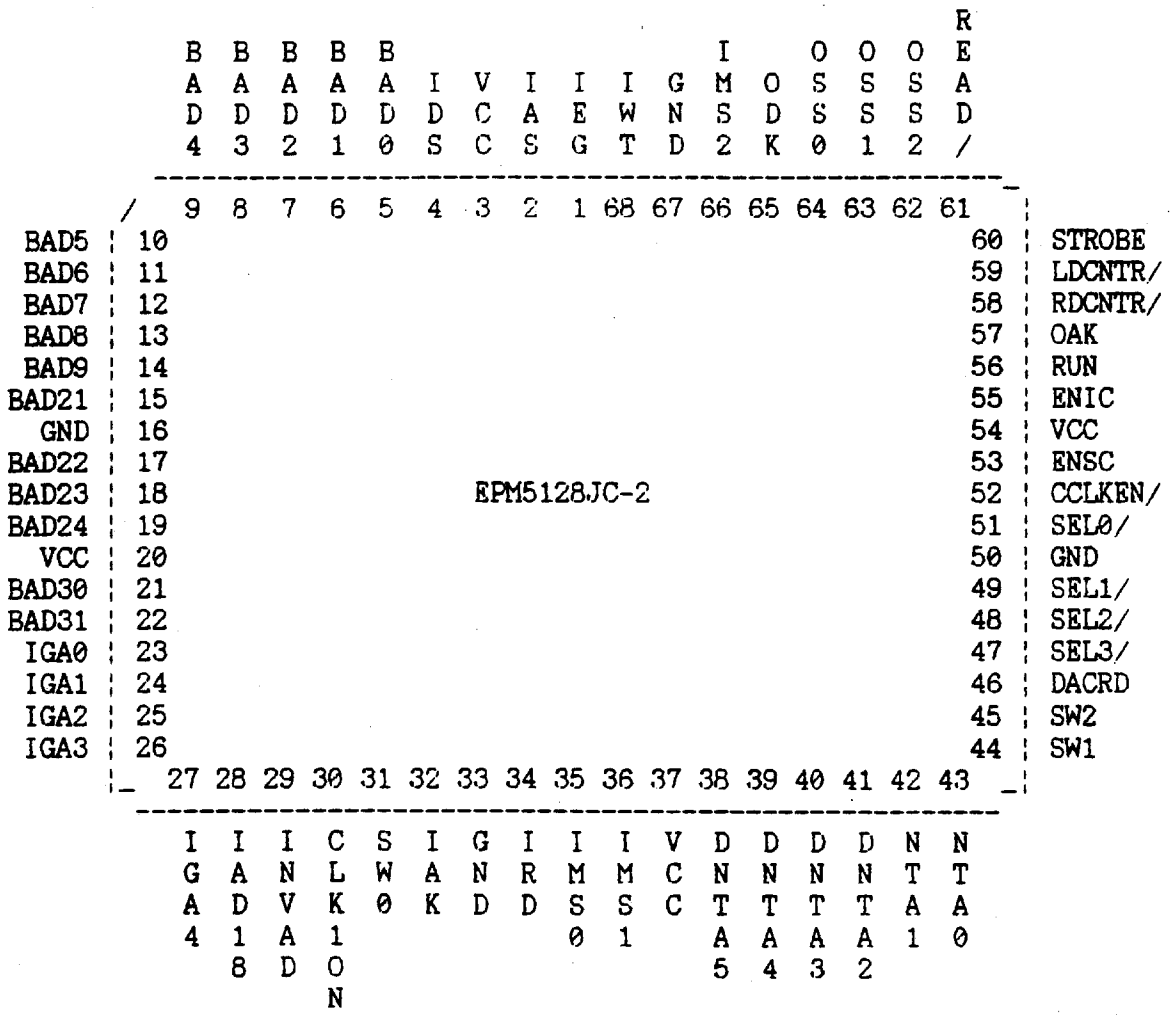
This appendix contains the information on the programming of the EPM5128 PAL that is used for the FASTBUS interface on the POSTAMP/COMPARATOR printed circuit board.

E:\KEN\SSD\PACMP\REVE\PACMPI.RPT

MAX+PLUS Compiler Report File
Version 2.50 05/31/90

** Design compiled without errors

Title: PA/CMP FASTBUS INTERFACE
Company: Fermilab
Designer: KEN TREPTOW
Rev: E
Date: 3:06p 11-15-1990
Turbo: ON
Security: OFF



** RESOURCE USAGE **

ic Array Block	Macrocells	I/O Pins	Expanders	External Interconnect
A: MC1 - MC16	15/16(93%)	8/ 8(100%)	27/32(84%)	19/24(79%)
B: MC17 - MC32	7/16(43%)	5/ 5(100%)	11/32(34%)	19/24(79%)
C: MC33 - MC48	4/16(25%)	5/ 5(100%)	1/32(3%)	6/24(25%)
D: MC49 - MC64	15/16(93%)	8/ 8(100%)	0/32(0%)	11/24(45%)
E: MC65 - MC80	16/16(100%)	8/ 8(100%)	15/32(46%)	23/24(95%)
F: MC81 - MC96	6/16(37%)	5/ 5(100%)	15/32(46%)	17/24(70%)
G: MC97 - MC112	7/16(43%)	5/ 5(100%)	3/32(9%)	24/24(100%)
H: MC113 - MC128	16/16(100%)	8/ 8(100%)	12/32(37%)	23/24(95%)

Total dedicated input pins used:	8/ 8 (100%)
Total I/O pins used:	52/ 52 (100%)
Total macrocells used:	86/128 (67%)
Total expanders used:	84/256 (32%)
Total input pins required:	20
Total output pins required:	24
Total bidirectional pins required:	16
Total macrocells required:	86
Total expanders in database:	69
Synthesized macrocells:	0/128 (0%)

** FILE HIERARCHY **

¦DACMAP:78¦
¦DACMAP:78¦74139:28¦
¦DACMAP:78¦74151:30¦
¦DACMAP:78¦74151:18¦
¦DACMAP:78¦74151:17¦
¦DACMAP:78¦74151:16¦
¦DACMAP:78¦74151:15¦
¦DACMAP:78¦74151:5¦
¦NTAREG:48¦
¦NTAREG:48¦DACSEL:108¦
¦NTAREG:48¦DACSEL:108¦74138H:11¦
¦NTAREG:48¦CT8:93¦
¦CONTROL:45¦
¦CONTROL:45¦DELAY:165¦
¦GASEL:46¦
¦OUTMUX:50¦
¦OUTMUX:50¦74153:81¦
¦OUTMUX:50¦74138H:82¦
¦OUTMUX:50¦74153:74¦
¦OUTMUX:50¦74153:75¦
¦OUTMUX:50¦74153:76¦
¦OUTMUX:50¦74153:77¦
¦OUTMUX:50¦74153:78¦
TMUX:50¦74153:79¦
TMUX:50¦74153:80¦
¦CSR0:49¦

** INPUTS **

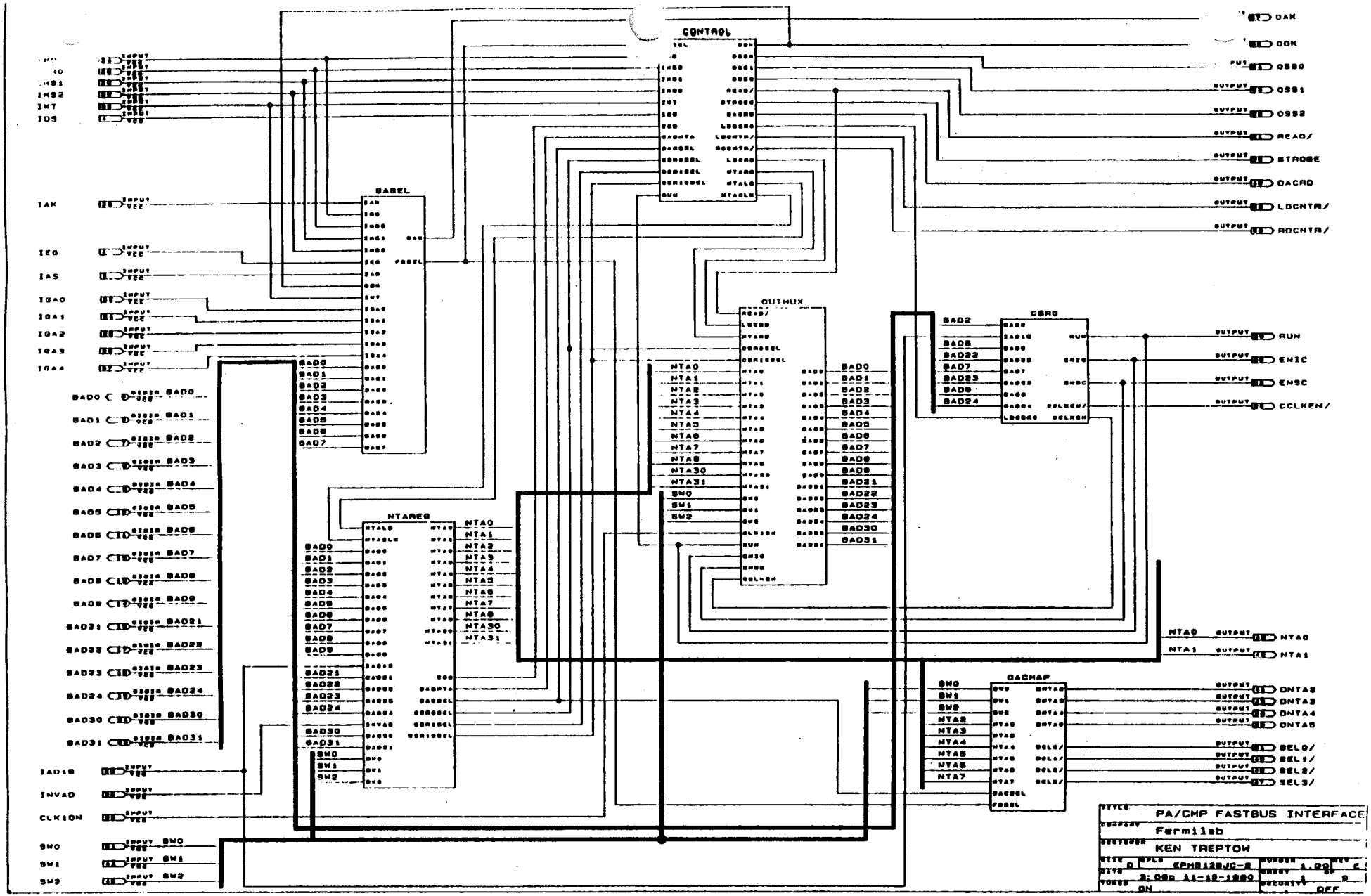
				Expanders		Fan-In		
1	MCell	LAB	Primitive	Total	Shared	INP	FBK	Name
5	(2)	(A)	INPUT	0	0	0	0	BAD0
6	(3)	(A)	INPUT	0	0	0	0	BAD1
7	(4)	(A)	INPUT	0	0	0	0	BAD2
8	(5)	(A)	INPUT	0	0	0	0	BAD3
9	(6)	(A)	INPUT	0	0	0	0	BAD4
10	(7)	(A)	INPUT	0	0	0	0	BAD5
11	(8)	(A)	INPUT	0	0	0	0	BAD6
12	(17)	(B)	INPUT	0	0	0	0	BAD7
13	(18)	(B)	INPUT	0	0	0	0	BAD8
14	(19)	(B)	INPUT	0	0	0	0	BAD9
15	(20)	(B)	INPUT	0	0	0	0	BAD21
17	(21)	(B)	INPUT	0	0	0	0	BAD22
18	(33)	(C)	INPUT	0	0	0	0	BAD23
19	(34)	(C)	INPUT	0	0	0	0	BAD24
21	(35)	(C)	INPUT	0	0	0	0	BAD30
22	(36)	(C)	INPUT	0	0	0	0	BAD31
30	(55)	(D)	INPUT	0	0	0	0	CLK1ON
28	(53)	(D)	INPUT	0	0	0	0	IAD18
32	-	-	INPUT	0	0	0	0	IAK
2	-	-	INPUT	0	0	0	0	IAS
4	(1)	(A)	INPUT	0	0	0	0	IDS
1	-	-	INPUT	0	0	0	0	IEG
23	(37)	(C)	INPUT	0	0	0	0	IGA0
24	(49)	(D)	INPUT	0	0	0	0	IGA1
25	(50)	(D)	INPUT	0	0	0	0	IGA2
6	(51)	(D)	INPUT	0	0	0	0	IGA3
27	(52)	(D)	INPUT	0	0	0	0	IGA4
35	-	-	INPUT	0	0	0	0	IMS0
36	-	-	INPUT	0	0	0	0	IMS1
66	-	-	INPUT	0	0	0	0	IMS2
29	(54)	(D)	INPUT	0	0	0	0	INVAD
34	-	-	INPUT	0	0	0	0	IRD
68	-	-	INPUT	0	0	0	0	IWT
31	(56)	(D)	INPUT	0	0	0	0	SW0
44	(71)	(E)	INPUT	0	0	0	0	SW1
45	(72)	(E)	INPUT	0	0	0	0	SW2

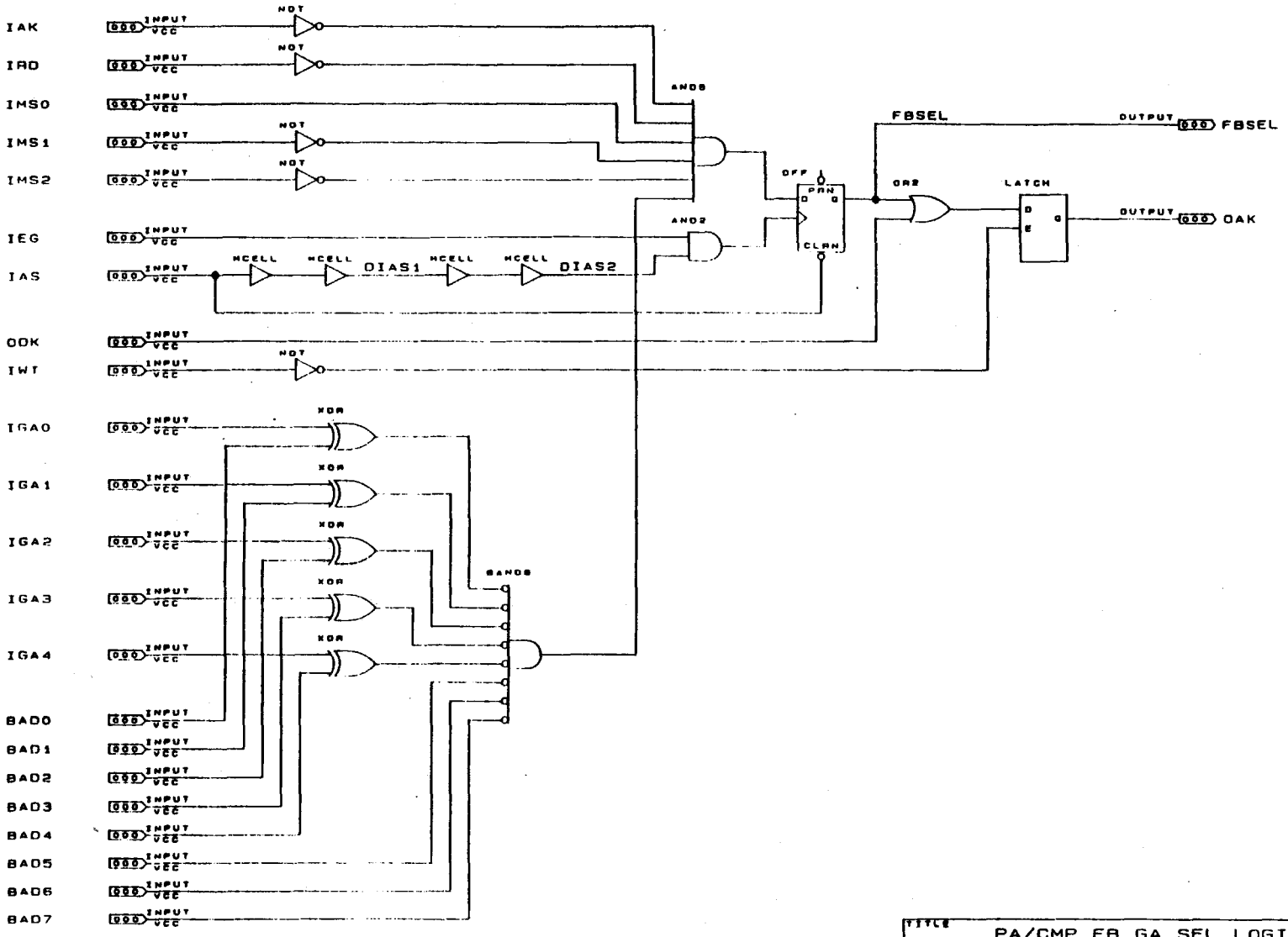
** OUTPUTS **

n	MCell	LAB	Primitive	Expanders Total	Expanders Shared	Fan-In INP	Fan-In FBK	Name
5	2	A	OR4	11	11	7	14	BAD0
6	3	A	OR4	11	11	7	14	BAD1
7	4	A	OR4	11	11	7	15	BAD2
8	5	A	OR4	11	11	7	14	BAD3
9	6	A	OR4	11	11	7	15	BAD4
10	7	A	OR4	11	11	7	14	BAD5
11	8	A	OR4	11	11	7	15	BAD6
12	17	B	OR4	11	11	4	15	BAD7
13	18	B	OR4	1	1	4	6	BAD8
14	19	B	OR4	1	1	5	4	BAD9
15	20	B	OR4	1	1	4	4	BAD21
17	21	B	OR4	1	1	4	2	BAD22
18	33	C	OR4	1	1	4	4	BAD23
19	34	C	OR4	1	1	4	4	BAD24
21	35	C	OR4	1	1	4	5	BAD30
22	36	C	OR4	1	1	4	5	BAD31
52	97	G	MCELL	0	0	0	1	CCLKEN/
46	81	F	OUTPUT	0	0	4	7	DACRD
41	68	E	OUTPUT	0	0	3	1	DNTA2
40	67	E	OUTPUT	0	0	3	2	DNTA3
39	66	E	OUTPUT	0	0	3	3	DNTA4
38	65	E	OUTPUT	5	0	3	4	DNTA5
55	99	G	DFF	3	3	6	15	ENIC
53	98	G	DFF	3	3	6	15	ENSC
59	114	H	OUTPUT	0	0	4	15	LDCNTR/
3	70	E	DFF	0	0	1	3	NTA0
2	69	E	DFF	0	0	1	4	NTA1
57	101	G	LATCH	0	0	1	2	OAK
65	120	H	LATCH	0	0	1	2	ODK
64	119	H	OUTPUT	0	0	3	3	OSS0
63	118	H	OUTPUT	8	7	4	11	OSS1
62	117	H	OUTPUT	7	7	4	10	OSS2
58	113	H	OUTPUT	0	0	3	13	RDCNTR/
61	116	H	OUTPUT	0	0	4	2	READ/
56	100	G	DFF	3	3	6	15	RUN
51	85	F	OUTPUT	5	0	3	11	SEL0/
49	84	F	OUTPUT	3	0	3	11	SEL1/
48	83	F	OUTPUT	4	0	3	11	SEL2/
47	82	F	OUTPUT	3	0	3	11	SEL3/
60	115	H	OUTPUT	0	0	3	10	STROBE

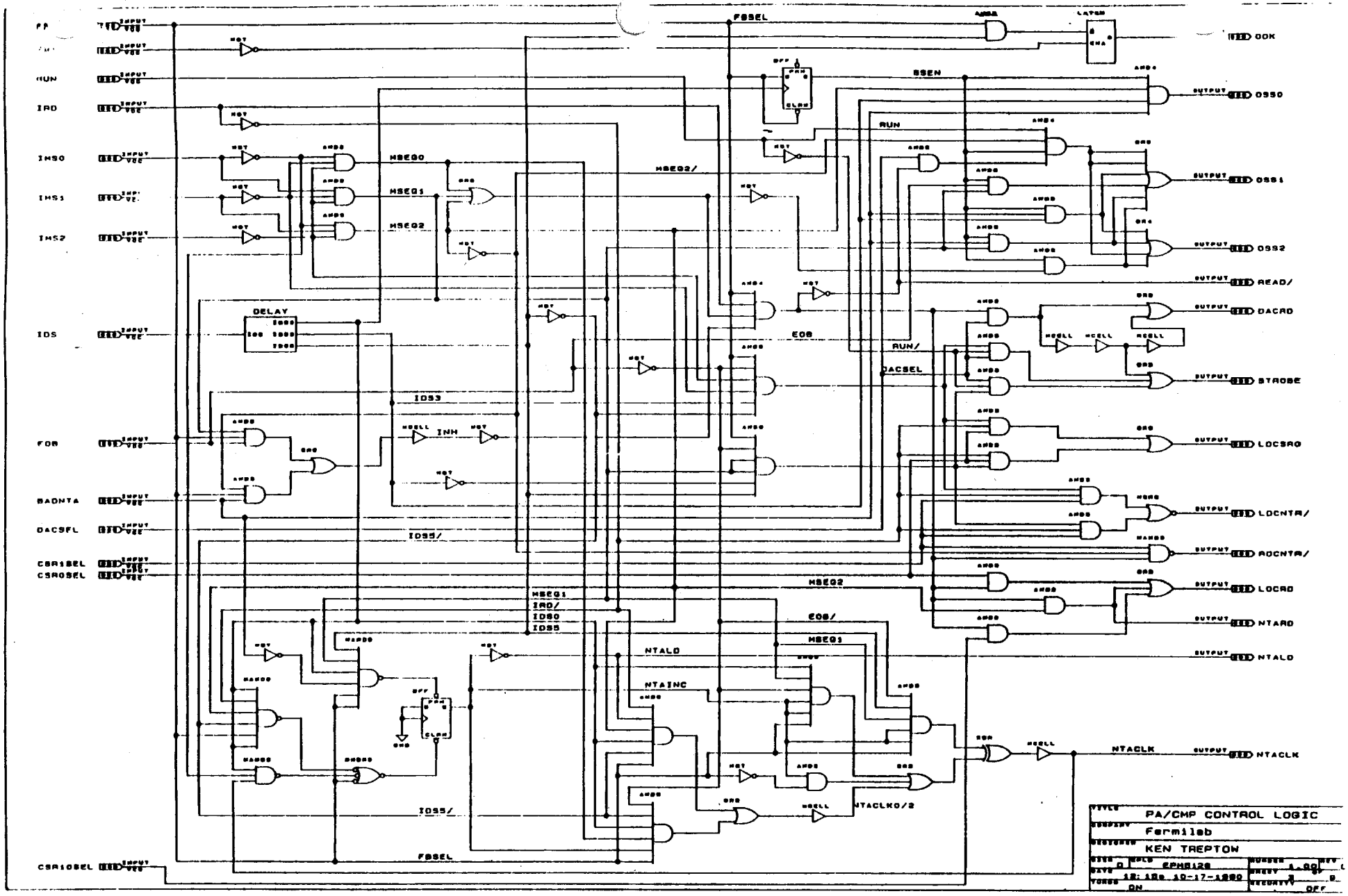
** BURIED LOGIC **

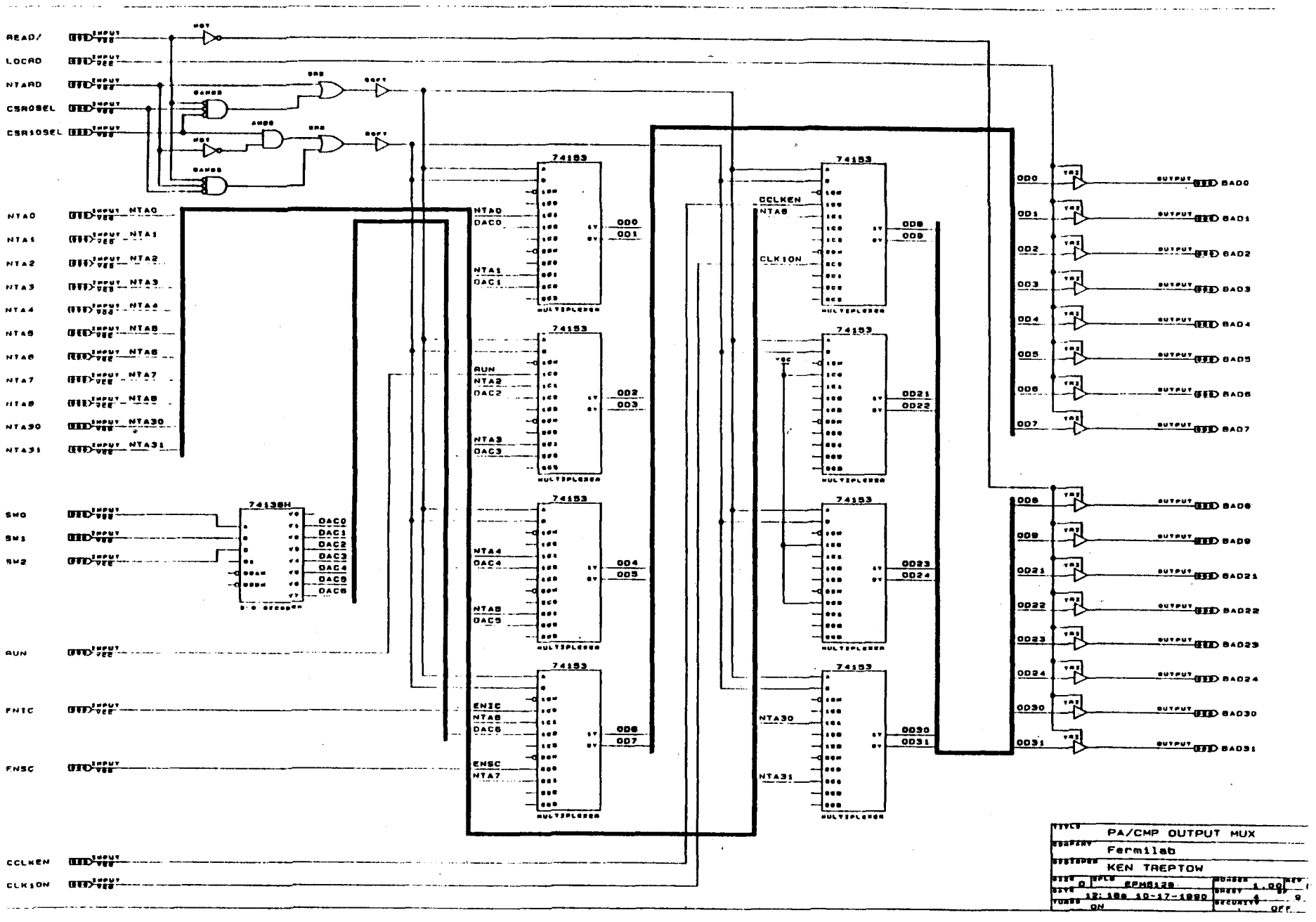
				Expanders		Fan-In		
↓	MCell	LAB	Primitive	Total	Shared	INP	FBK	Name
-	64	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS0
-	63	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS1
-	62	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS2
-	61	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS3
-	60	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS4
-	80	E	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|IDS5
-	57	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:34
(27)	52	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:37
-	59	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:38
-	58	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:39
(31)	56	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:41
(30)	55	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:42
-	79	E	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:43
(29)	54	D	MCELL	0	0	1	0	|CONTROL:45|DELAY:165|:46
(26)	51	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:49
(25)	50	D	MCELL	0	0	0	1	|CONTROL:45|DELAY:165|:50
-	128	H	MCELL	0	0	3	3	|CONTROL:45|INH
-	78	E	MCELL	0	0	3	6	|CONTROL:45|NTACLK
-	77	E	MCELL	0	0	4	5	|CONTROL:45|NTACLK0/2
-	76	E	DFF	3	0	4	5	|CONTROL:45|NTAINC
-	96	F	DFF	0	0	0	2	|CONTROL:45|SSEN
-	16	A	MCELL	0	0	4	6	|CONTROL:45|:187
-	15	A	MCELL	0	0	0	1	|CONTROL:45|:188
-	14	A	MCELL	0	0	0	1	|CONTROL:45|:189
-	127	H	DFF	3	3	6	15	|CSR0:49|CCLKEN
-	13	A	MCELL	0	0	0	1	|GASEL:46|DIAS1
-	112	G	MCELL	0	0	0	1	|GASEL:46|DIAS2
-	75	E	DFF	7	0	20	1	|GASEL:46|FBSEL
-	12	A	MCELL	0	0	1	0	|GASEL:46|:21
-	111	G	MCELL	0	0	0	1	|GASEL:46|:51
-	126	H	MCELL	0	0	0	12	|NTAREG:48|BADNTA
-	31	B	DFF	0	0	1	5	|NTAREG:48|CT8:93|Q2
-	74	E	DFF	0	0	1	6	|NTAREG:48|CT8:93|Q3
-	73	E	DFF	0	0	1	7	|NTAREG:48|CT8:93|Q4
(45)	72	E	DFF	0	0	1	8	|NTAREG:48|CT8:93|Q5
(44)	71	E	DFF	0	0	1	9	|NTAREG:48|CT8:93|Q6
-	125	H	DFF	0	0	1	10	|NTAREG:48|CT8:93|Q7
-	11	A	SOFT	7	0	3	6	|NTAREG:48|DACSEL:108|:52
-	10	A	SOFT	7	0	3	8	|NTAREG:48|DACSEL:108|:53
-	124	H	MCELL	0	0	0	13	|NTAREG:48|EN/
-	123	H	DFF	1	0	0	15	|NTAREG:48|EOB
(28)	53	D	DFF	0	0	8	2	|NTAREG:48|NTA8
-	122	H	DFF	0	0	1	2	|NTAREG:48|NTA30
-	121	H	DFF	0	0	1	2	|NTAREG:48|NTA31
-	9	A	SOFT	2	0	4	13	|OUTMUX:50|:90
-	32	B	SOFT	0	0	4	12	|OUTMUX:50|:91

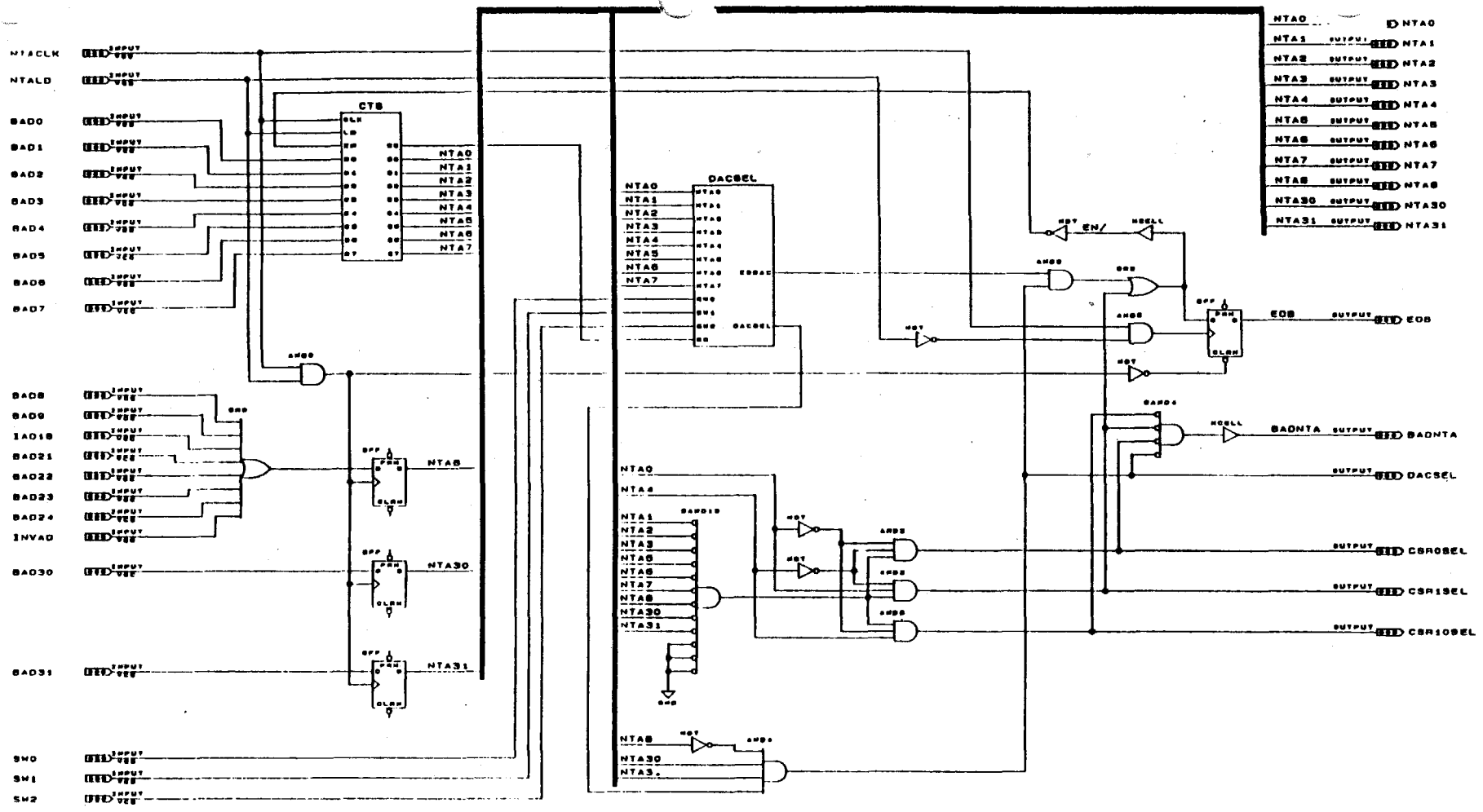

CONTROL
OUTMUX
CSRO
DACMAP
PA/CMP FASTBUS INTERFACE
Fermilab
KEN TREPTOW



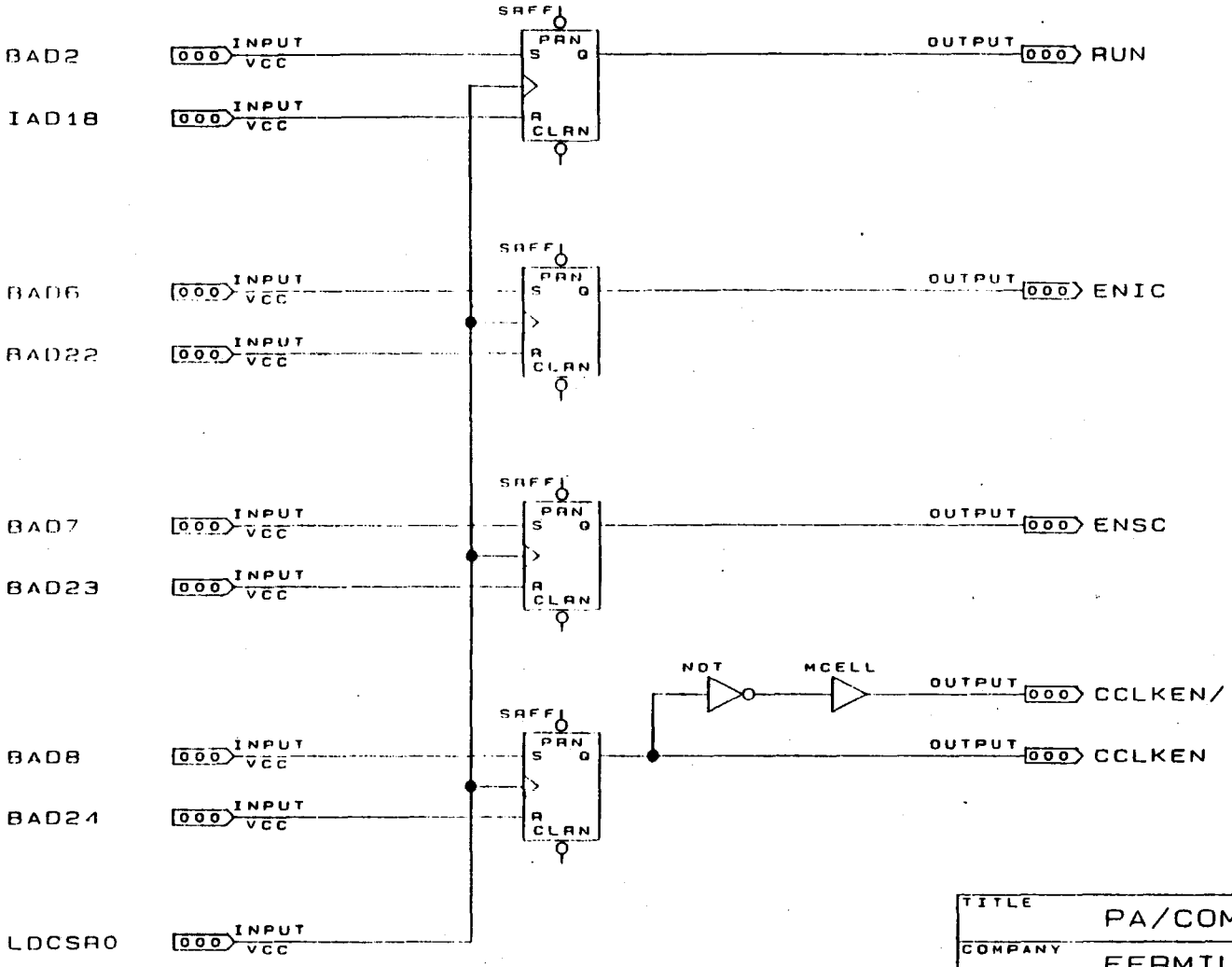
TITLE	PA/CMP FB GA SEL LOGIC
COMPANY	FERMILAB
DESIGNER	KEN TREPTOW
SIZE C / EPLD EPM5128 / NUMBER 1.00 / REV B	
DATE 6/11/90 / SHEET 4 OF 9	
TYPIST QN / SECURITY OFF	


RUN
IRD
IHS0
IHS1
IHS2
IDS
FDR
BADNTA
DACSEL
CSR1SEL
CSR0SEL
CSR10SEL
DELAY
INH
MSEQ0
MSEQ1
MSEQ2
MSEQ2/
IDS3
IDS5/
FBSEL
EOS
EOS/
RUN/
BSEN
NTALD
NTAINC
NTACLK0/2
NTACLK
OOK
OSS0
OSS1
OSS2
READ/
DACRD
STROBE
LOCSRD
LOCNTR/
ROCNTR/
LOCRD
NTARD
NTALD
NTACLK
PA/CMP CONTROL LOGIC
Fermilab
KEN TREPTON
EPM5128
12:12a 10-17-1990

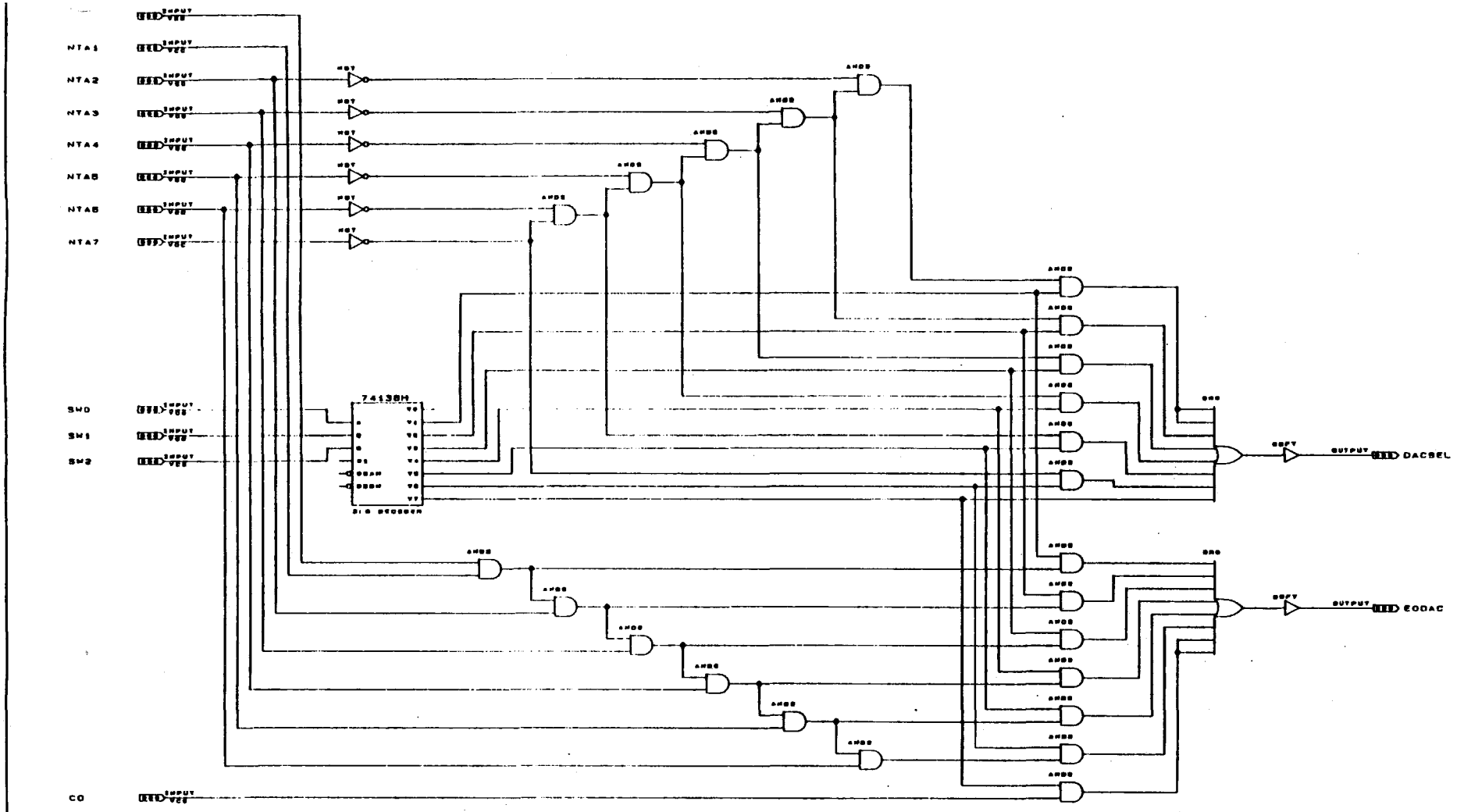

READ/
LOCRD
NTARD
CSR0SEL
CSR1SEL
NTA0
NTA1
NTA2
NTA3
NTA4
NTA5
NTA6
NTA7
NTA8
NTA30
NTA31
SW0
SW1
SW2
RUN
FNIC
FNSC
CCLKEN
CLK1ON
74138H
74153
MULTIPLEXER
DAC0
DAC1
DAC2
DAC3
DAC4
DAC5
DAC6
ENIC
ENSC
OD0
OD1
OD2
OD3
OD4
OD5
OD6
OD7
OD21
OD22
OD23
OD24
OD30
OD31
BAD0
BAD1
BAD2
BAD3
BAD4
BAD5
BAD6
BAD7
BAD8
BAD9
BAD21
BAD22
BAD23
BAD24
BAD30
BAD31
PA/CMP OUTPUT MUX
Fermilab
KEN TREPTOW
EPM5128
12:18a 10-17-1990
ON
OFF



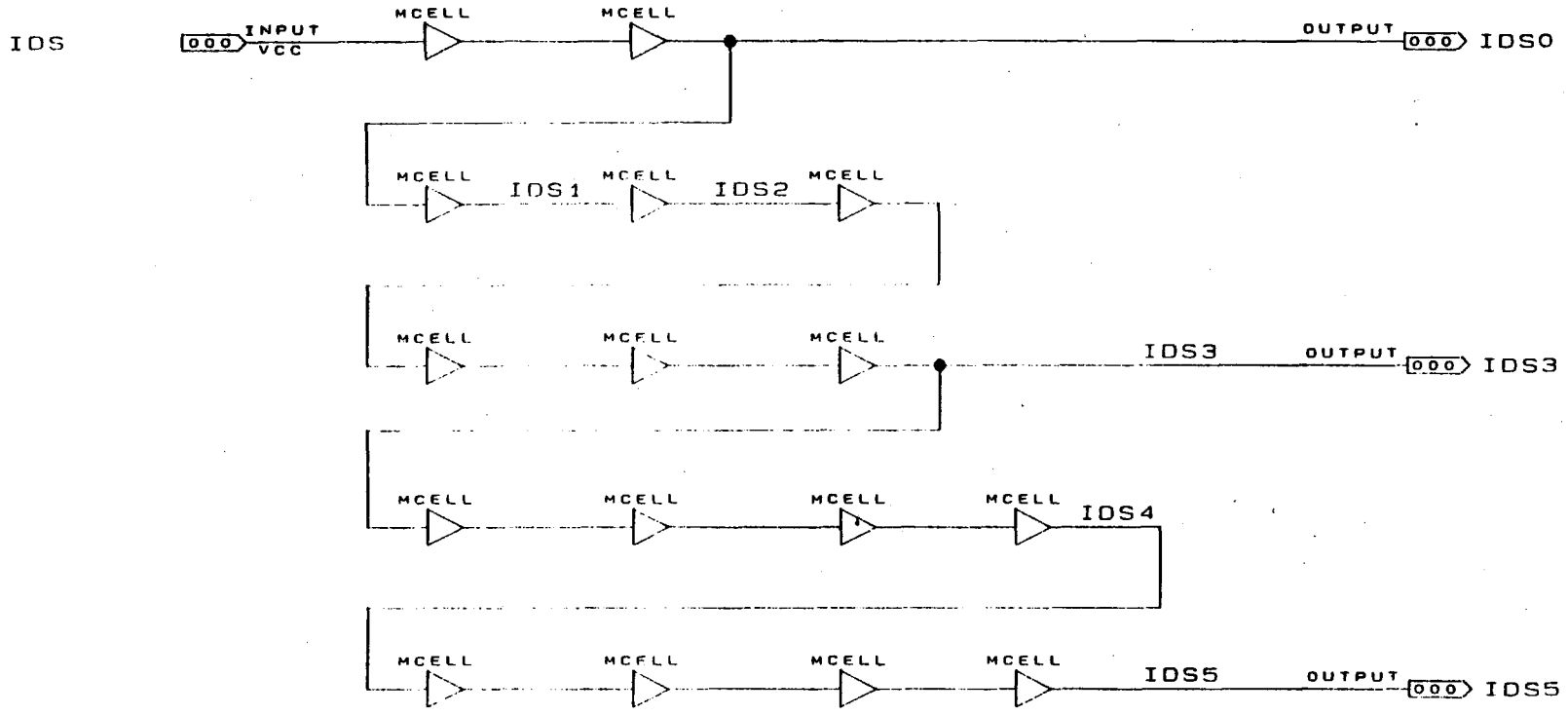
TITLE	PA/CMP NTA REGISTER
COMPANY	Fermilab
DESIGNER	KEN TAEPTOW
STEP D	DPLS FPM8128
DATE	12:14A 10-17-1990
TURBO	ON
NUMBER	A.00 REV
SHEET	OF
SECURITY	OFF



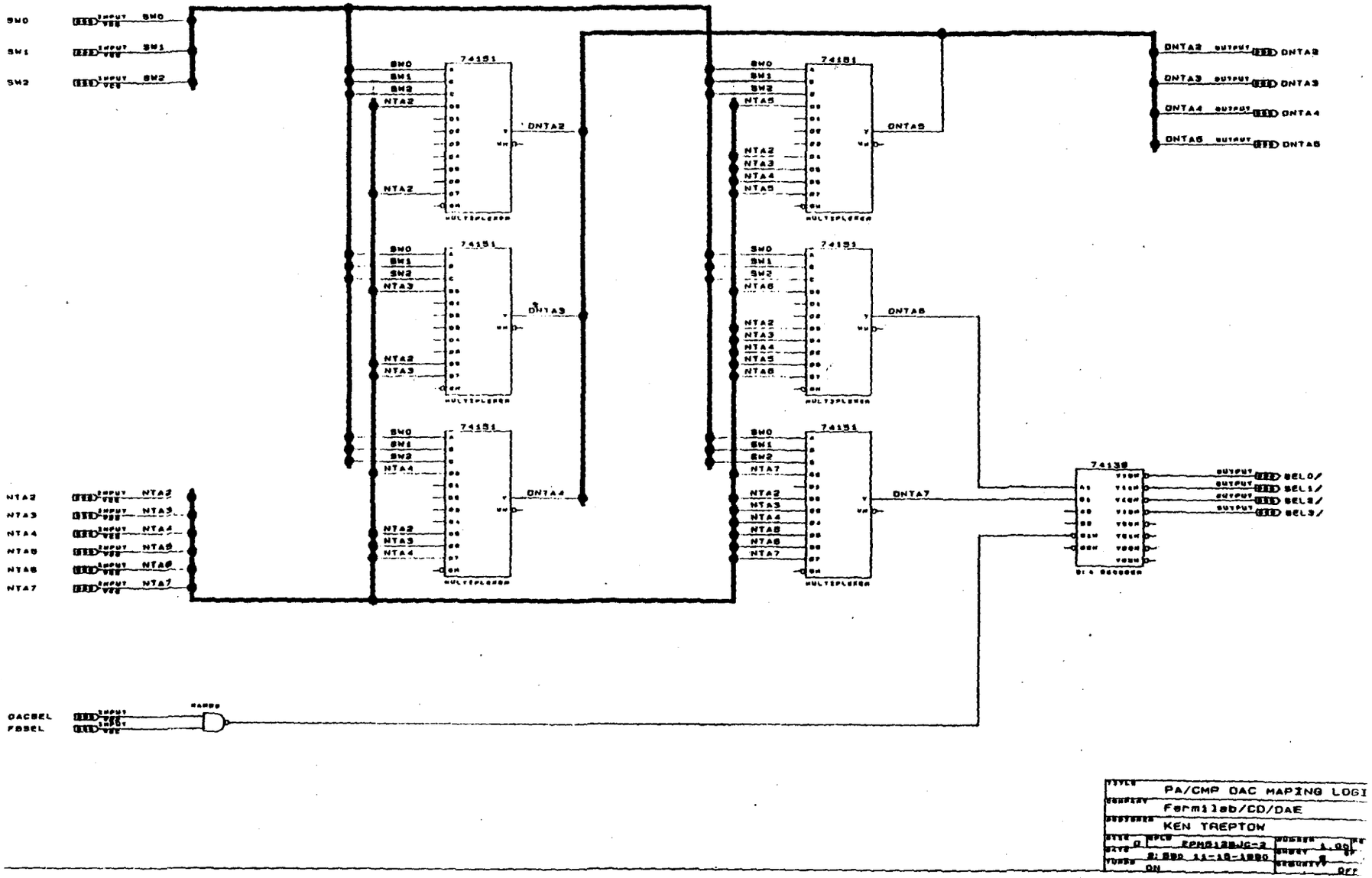
TITLE	PA/COMP CSR0		
COMPANY	FERMILAB/CD/DAE		
DESIGNER	KEN TREPTOW		
SIZE B	EPLD EPM5128	NUMBER 1.00	REV B
DATE 6/26/90		SHEET 6 OF 9	
TURBO ON		SECURITY OFF	



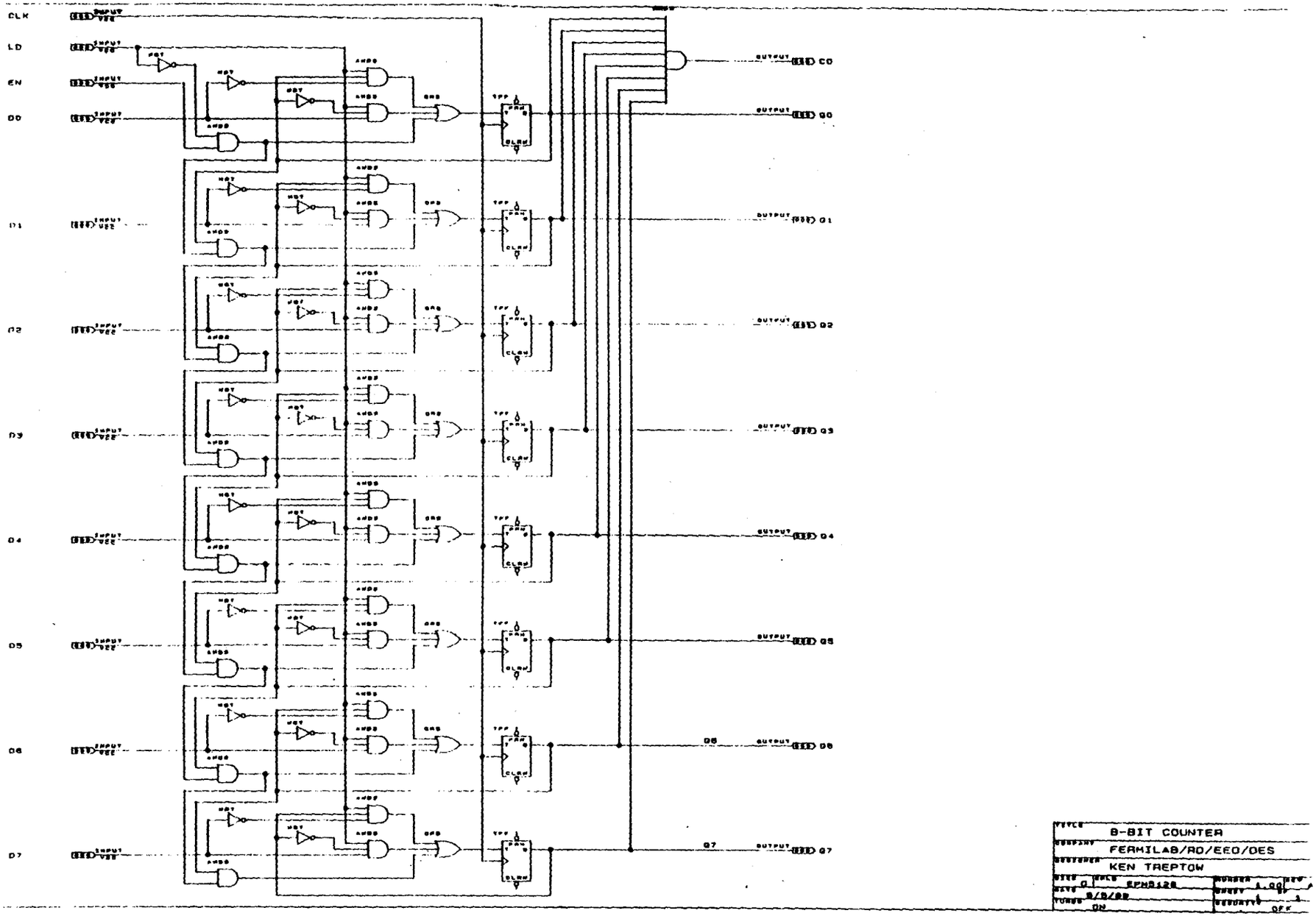
TITLE	PA/CMP DAC ADDR DECODE
COMPANY	Fermilab
DESIGNER	KEN TREPTOW
SIZE D	APLD EPM5128
DATE	12:17a 10-17-1990
TURBO	ON
NUMBER	1.00
SHEET	7 OF 9
SECURITY	OFF



TITLE	PA/CMP DS DELAYS		
COMPANY	Fermilab		
DESIGNER	KEN TREPTOW		
SIZE B	EPLD EPM5128	NUMBER 1.00	REV A
DATE 10:22a 11-13-1990		SHEET 8 OF 9	
TURBO ON		SECURITY OFF	

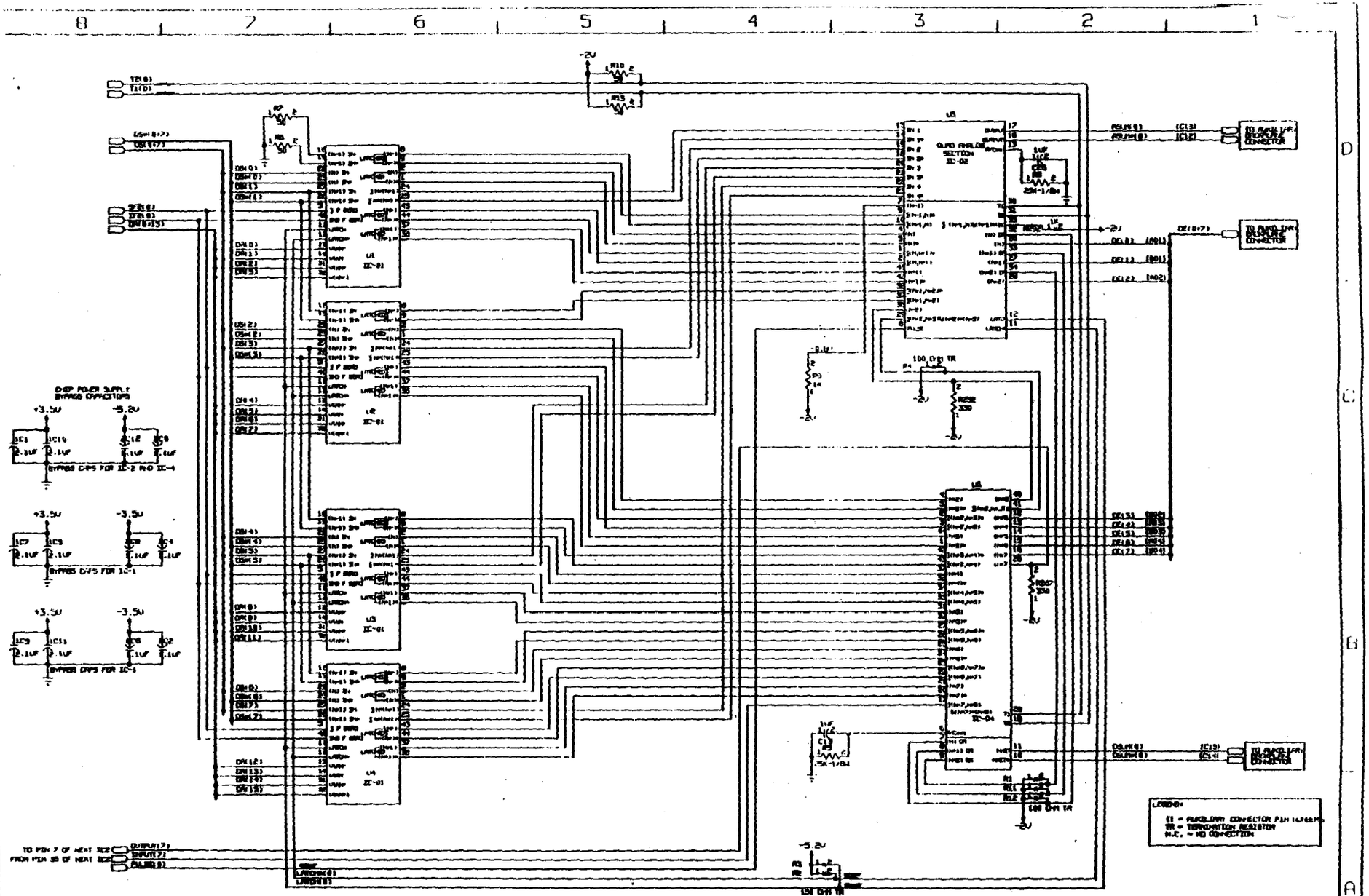


TITLE	PA/CMP DAC MAPING LOGI
COMPANY	Fermilab/CD/DAE
AUTHOR	KEN TREPTOW
SIZE D	MPLD EPM5128JC-2 NUMBER REV 1.00
DATE	D: SUN 11-15-1992 SHEET OF
TIME	ON DRAWING NO. REF



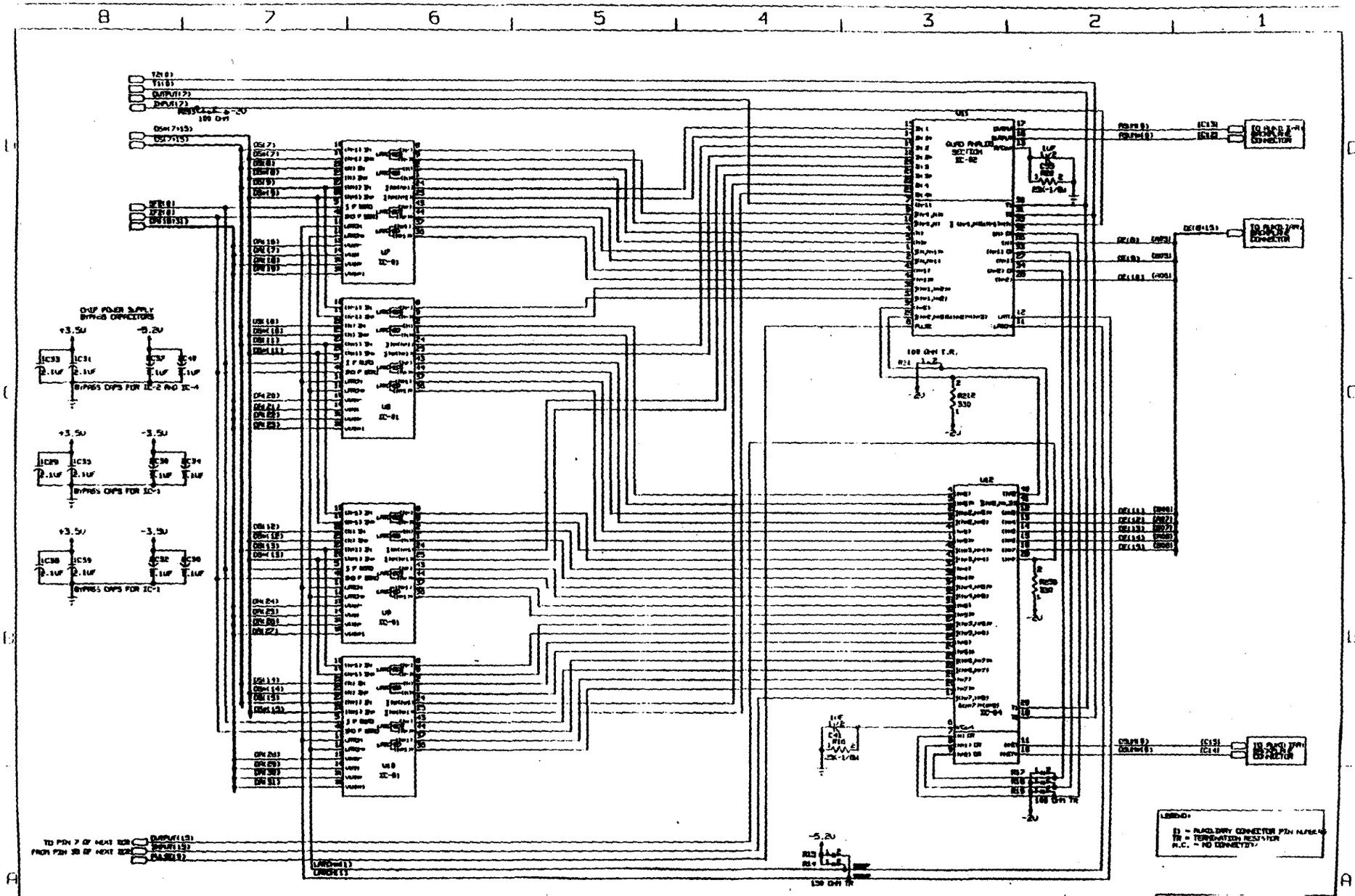
DEVICE	8-BIT COUNTER
DESIGNER	FERMILAB/RD/EED/DES
DRAWN BY	KEN TREPTOW
DATE	8/8/88
TURNS	ON
DEVICE	EPM5128
NUMBER	1.00
SHEET	1 OF 1
SECURITY	OFF

APPENDIX C - POSTAMP/COMPARATOR MODULE - SCHEMATIC



E771 DISCRIMINATOR/POST AMP COMPARATOR
ASIC LOGIC SCHEMATIC
SECTION 1 OF 16
SHEET #1

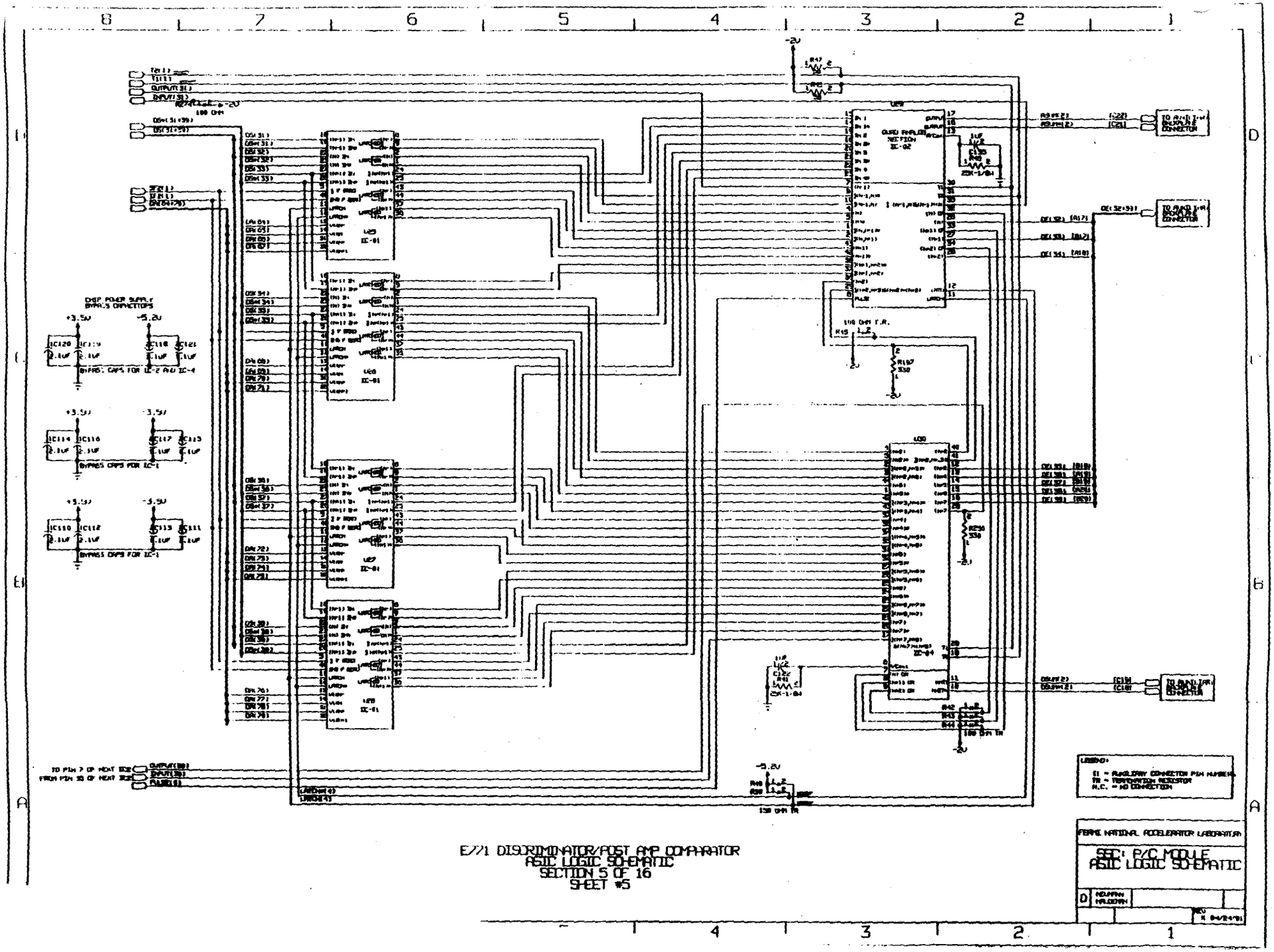
FERMI NATIONAL ACCELERATOR LABORATORY
SSC P/C MODULE
ASIC LOGIC SCHEMATIC



E771 DISCRIMINATOR/POST COMPARATOR
ASIC LOGIC 904
SECTION 2 OF 3
SHEET #2

APPENDIX D - MODULE DRAWINGS

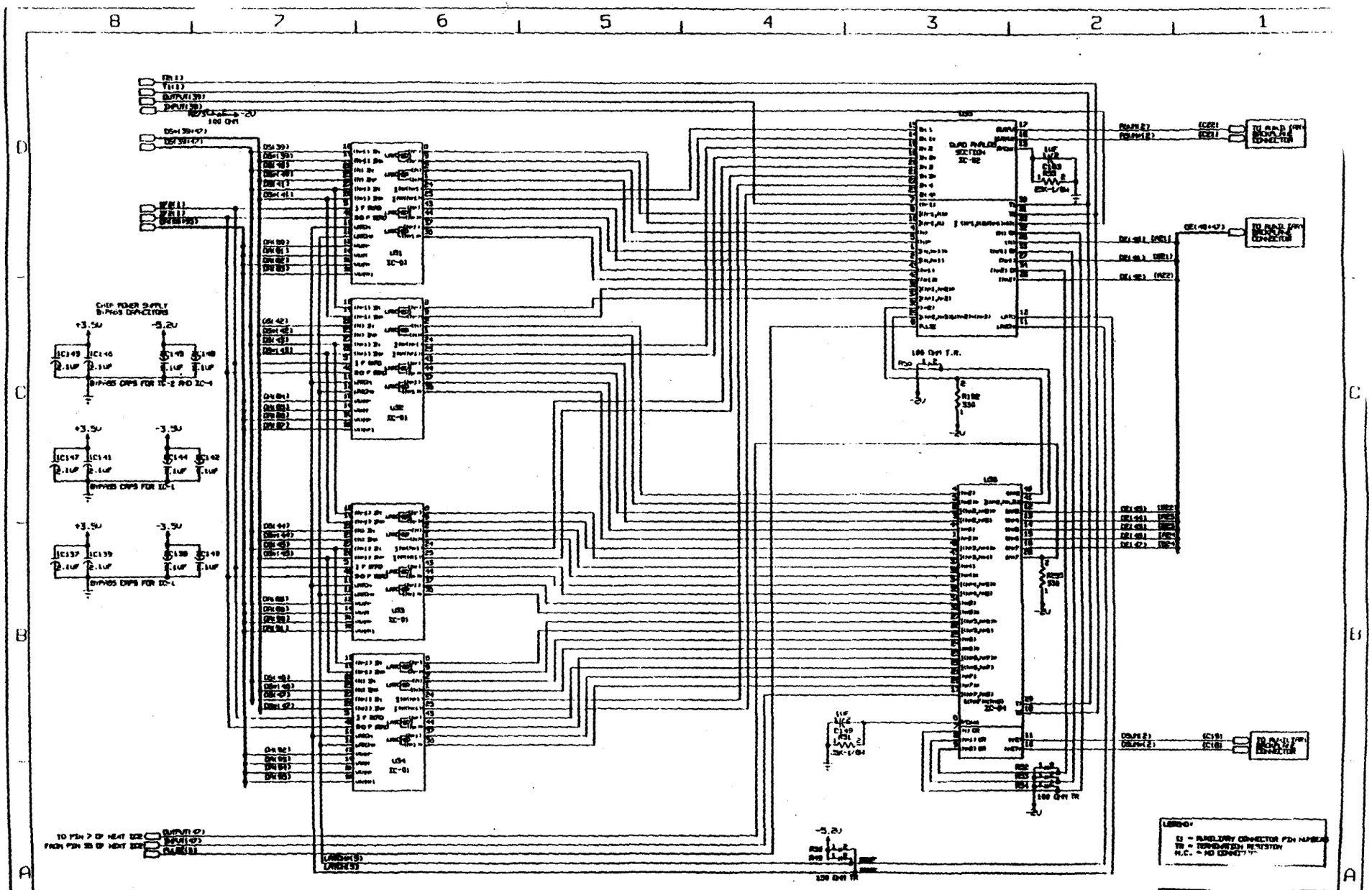
1.	2563.000-ED-215747	EED E771 - POST AMP COMPARATOR VER III SCHEMATIC DIAGRAM REV. I
2.	2563.000-MD-215752	EED E771 - POST AMP COMPARATOR VER III FRONT PANEL & FRONT PANEL AIR GAP FILLER DETAILS REV. A
3.	2563.000-MD-215753	EED E771 - POST AMP COMPARATOR VER III FRONT PANEL SILK-SCREEN REV. C
4.	2563.000-MD-215790	EED E771 - POST AMP COMPARATOR VER III DRILL DRAWING REV. -
5.	2563.000-MD-215802	EED E771 - POST AMP COMPARATOR VER III BOARD LAYUP AND OUTLINE DETAILS REV. -
6.	2563.000-MD-215803	EED E771 - POST AMP COMPARATOR VER III P.C. BOARD ASSEMBLY REV. -
7.	2563.000-MD-215804	EED E771 - POST AMP COMPARATOR VER III BOTTOM SIDE ASSEMBLY REV. -
8.	2563.000-MD-215805	EED E771 - POST AMP COMPARATOR VER III TOP ASSEMBLY REV. A
9.	2563.000-MD-215806	EED E771 - POST AMP COMPARATOR VER III LED AND TEST POINTS WIRING DETAILS REV. -
10.	2563.000-MD-215807	EED E771 - POST AMP COMPARATOR VER III DAC JUMPER DIAGRAM REV. -
11.	2563.000-MD-215808	EED E771 - POST AMP COMPARATOR VER III POWER TEST POINTS DETAILS REV. -



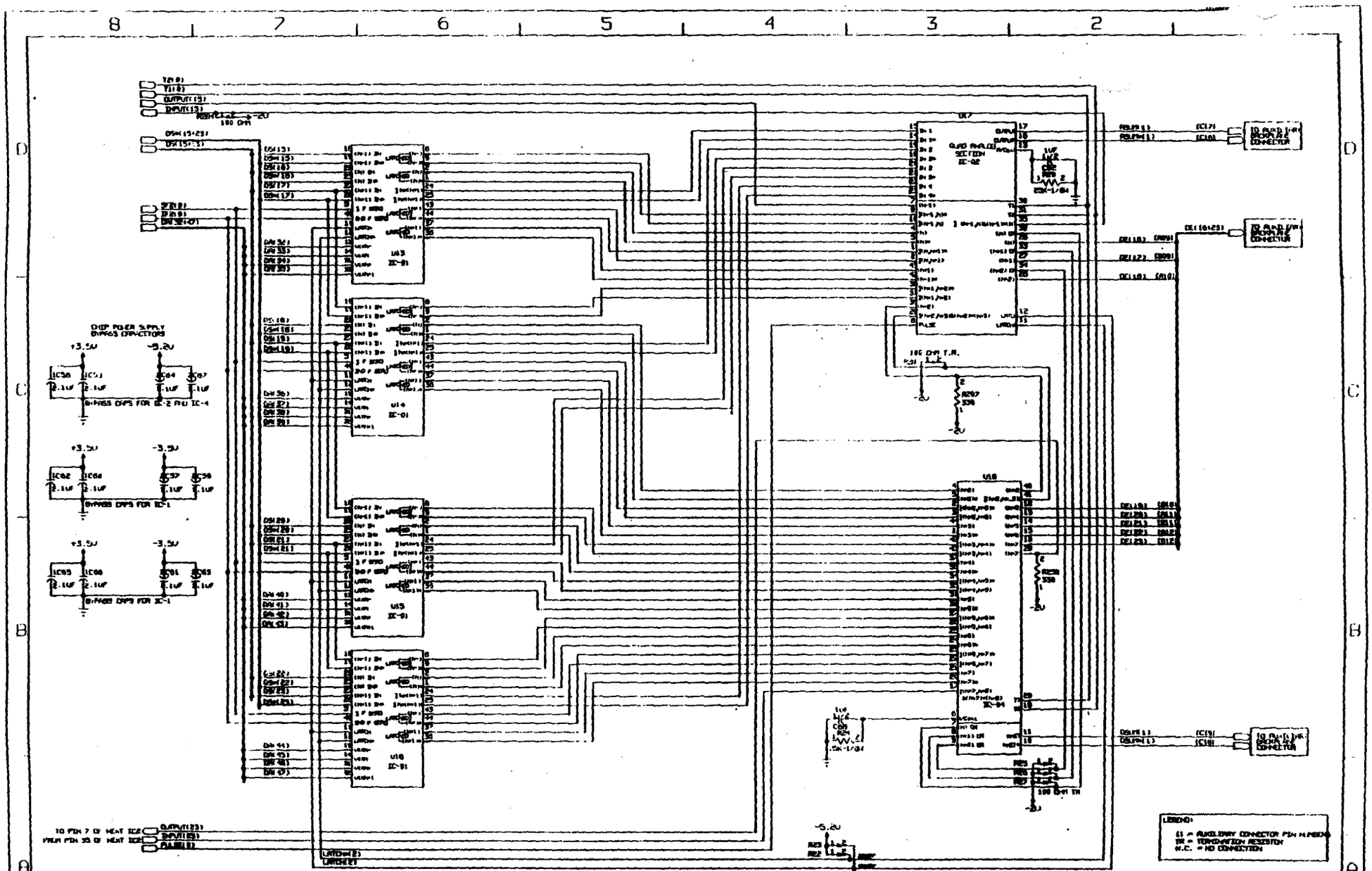
E771 DISCRIMINATOR/POST AMP COMPARATOR
ASIC LOGIC SCHEMATIC
SECTION 5 OF 16
SHEET #5

FERMI NATIONAL ACCELERATOR LABORATORY

SEC: P/C MODULE
ASIC LOGIC SCHEMATIC



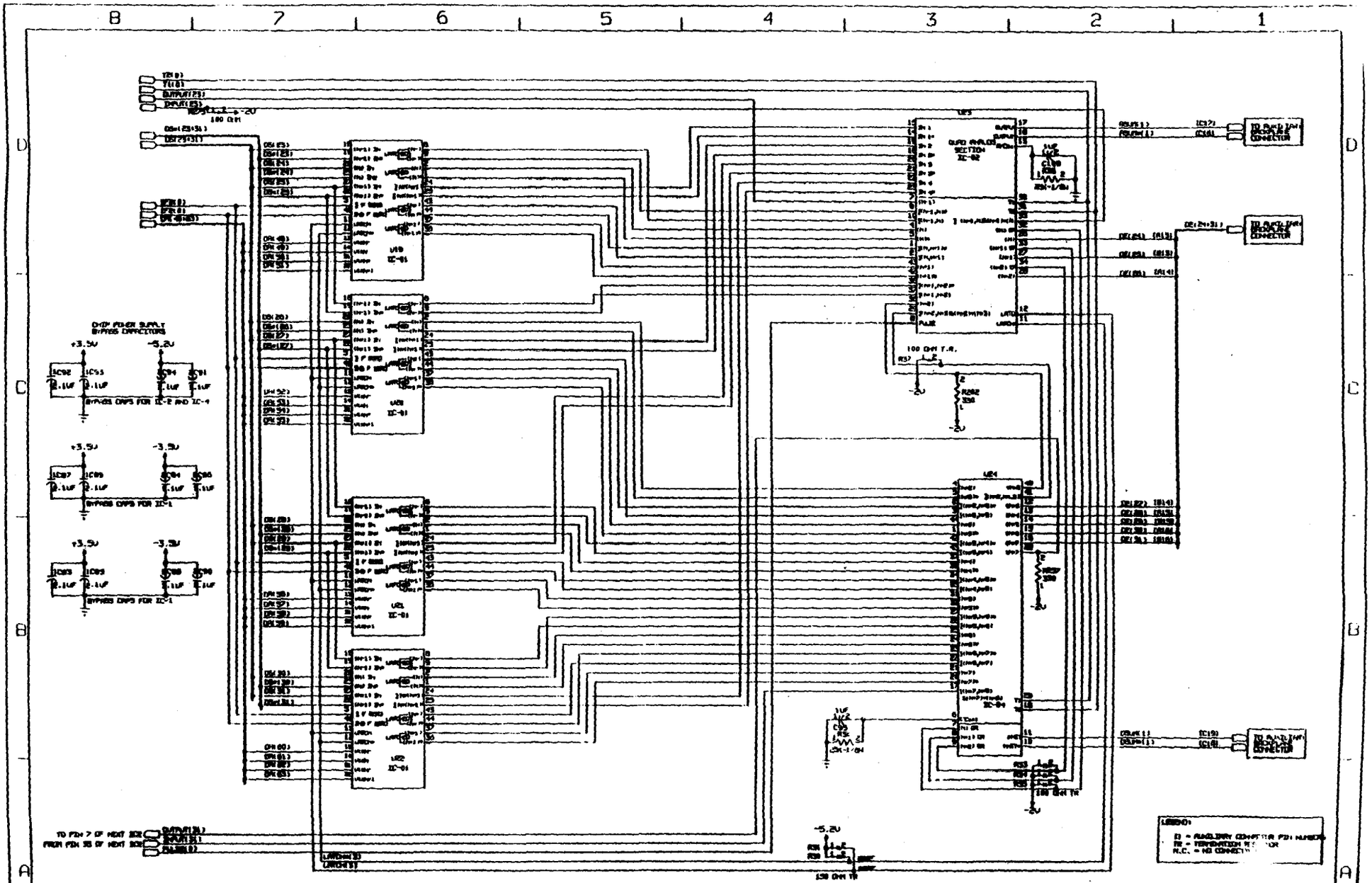
E771 DISCRIMINATOR/POST AMP COMPARATOR
ASIC LOGIC SD[illegible] IIC
SECTION 6 OF
SHEET #6



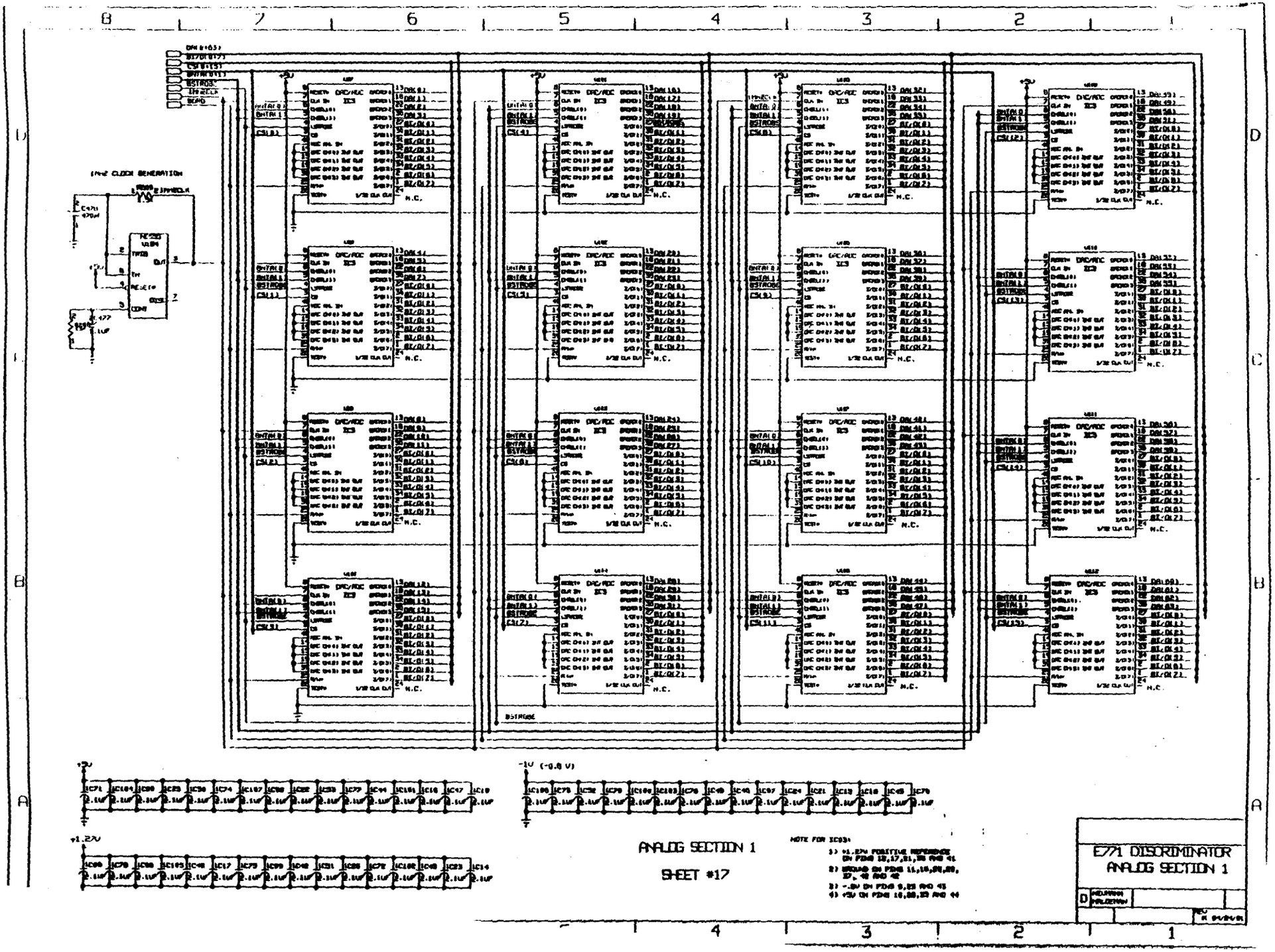
E771 DISCRIMINATOR/POST AMP COMPARATOR
ASIC LOGIC SCHEMATIC
SECTION 3 OF 16
SHEET #3

FERMI NATIONAL ACCELERATOR LABORATORY

SSC P/C MODULE
ASIC LOGIC SCHEMATIC



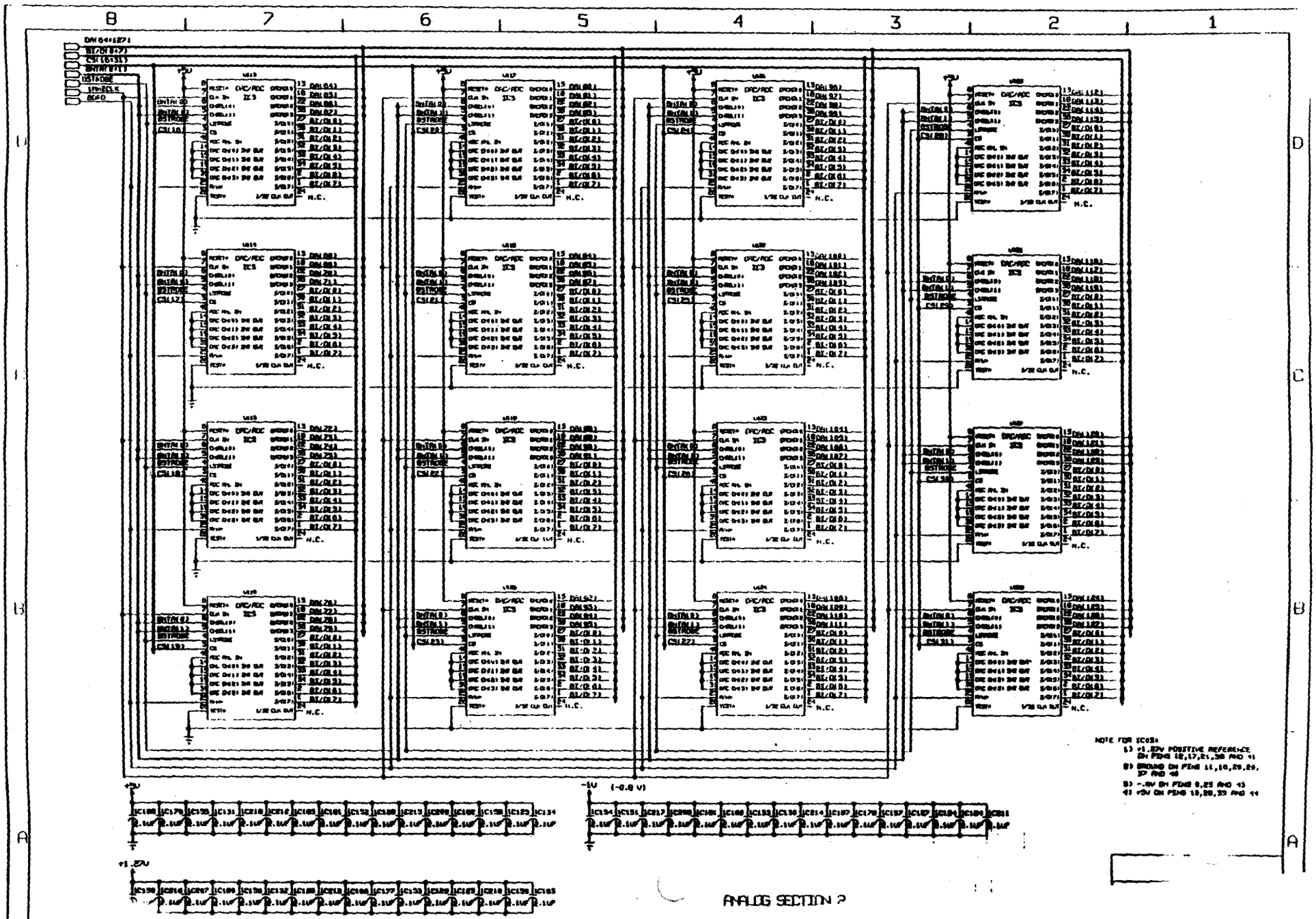
E771 DISCRIMINATOR/POST [illegible] OMPARATOR
ASIC LOGIC SCHEMATIC
SECTION 4 OF [illegible]



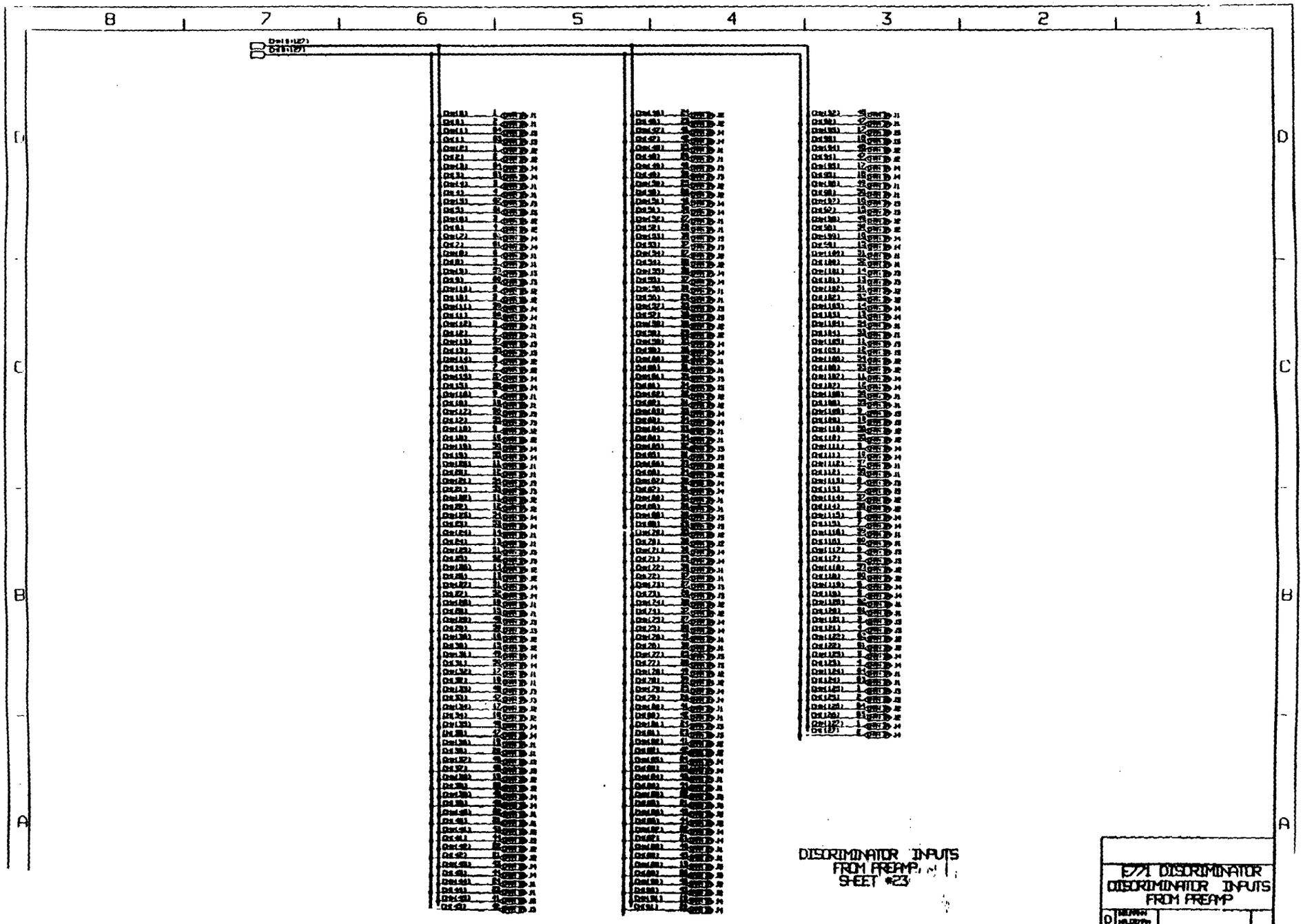
ANALOG SECTION 1

SHEET #17

E771 DISCRIMINATOR
ANALOG SECTION 1

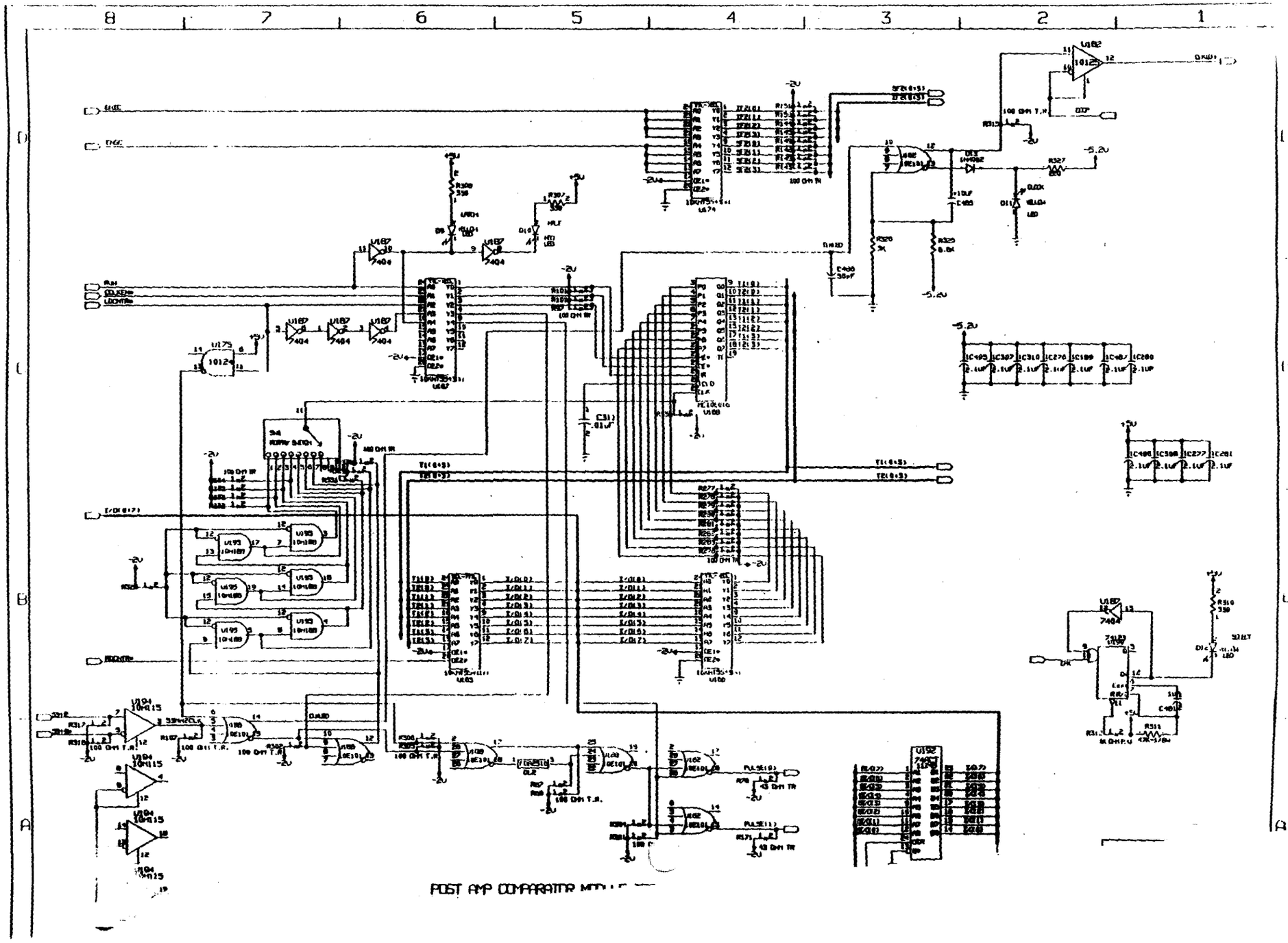


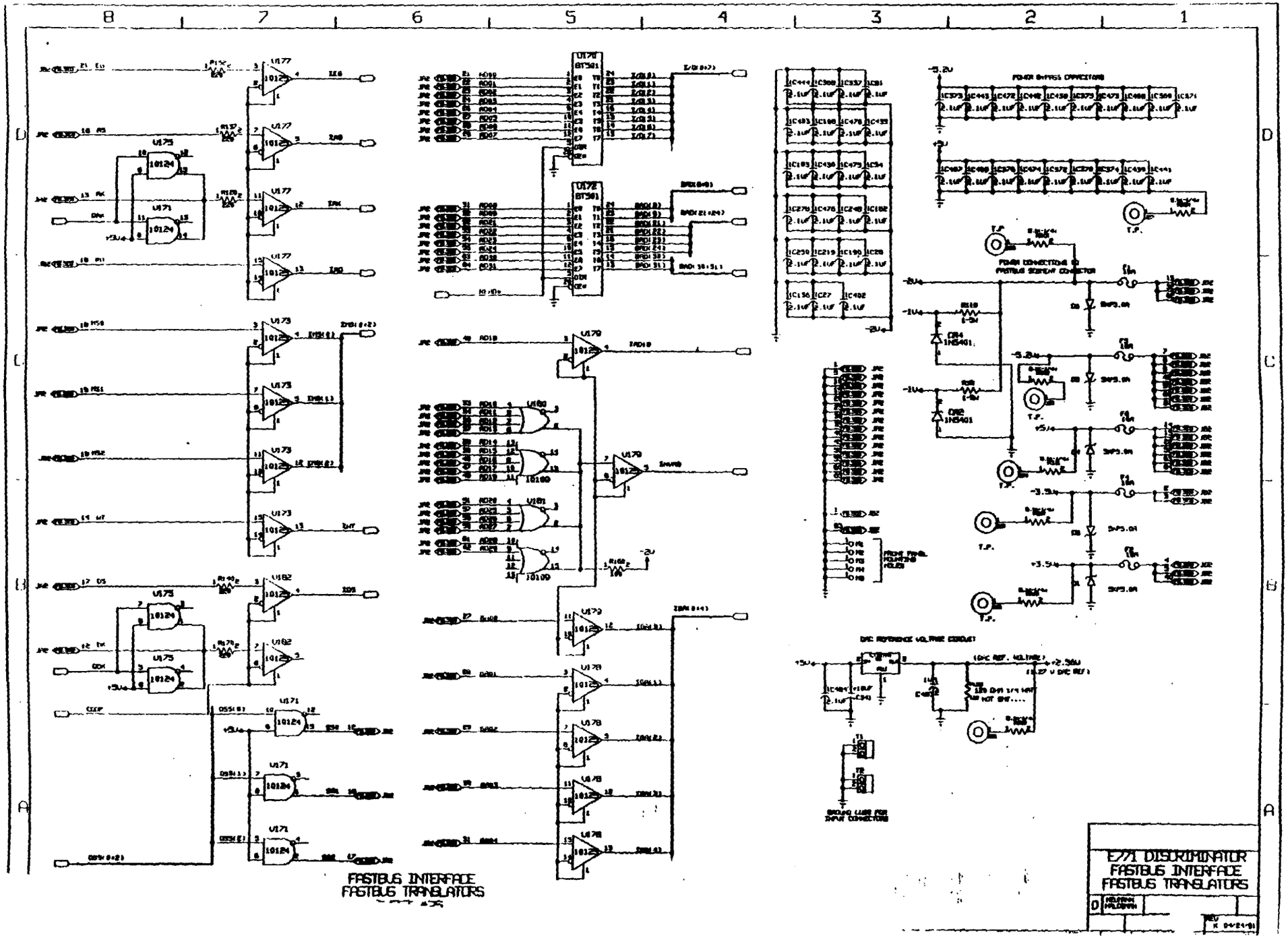
ANALOG SECTION 2

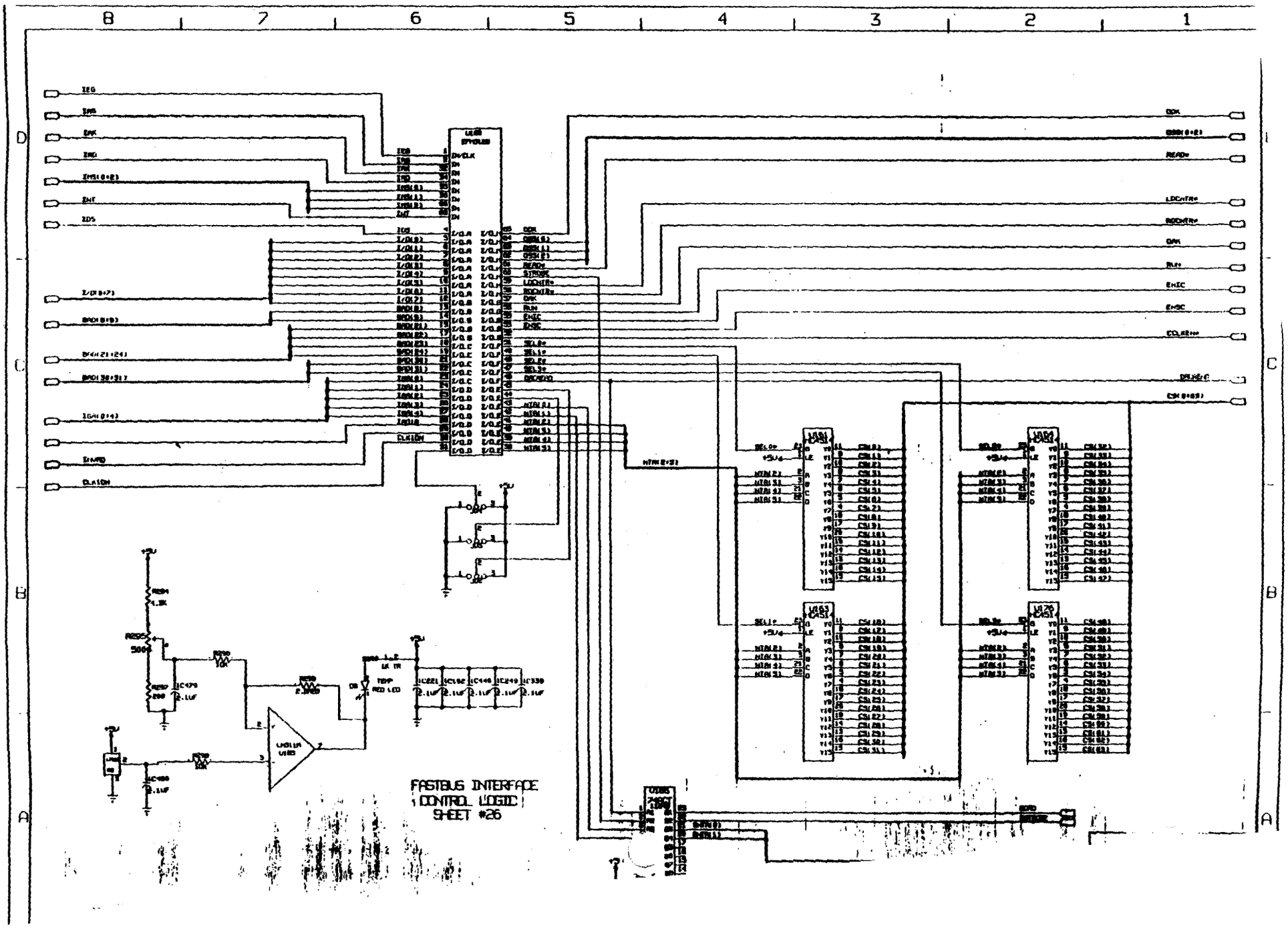


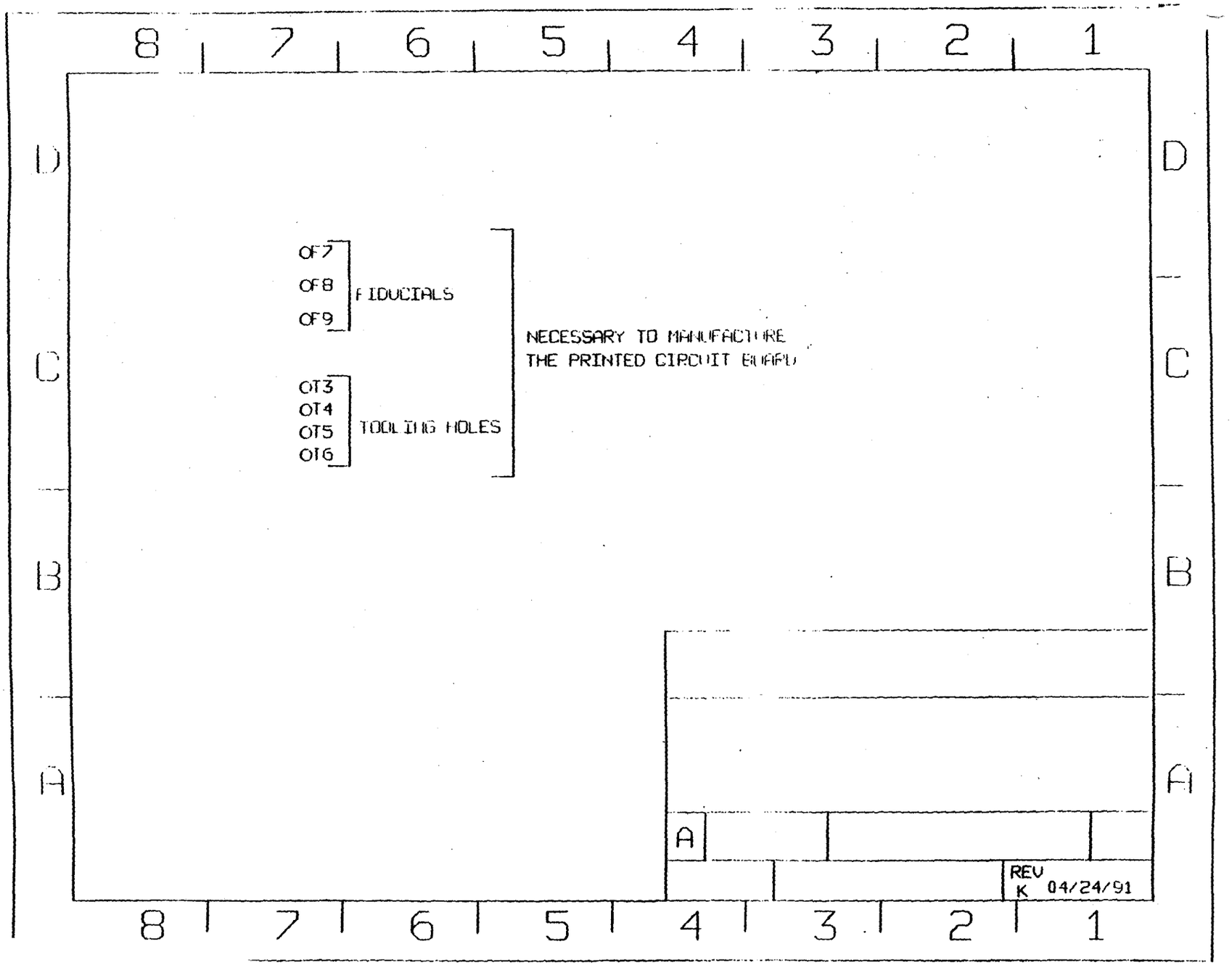
DISCRIMINATOR INPUTS
FROM PREAMP
SHEET #23

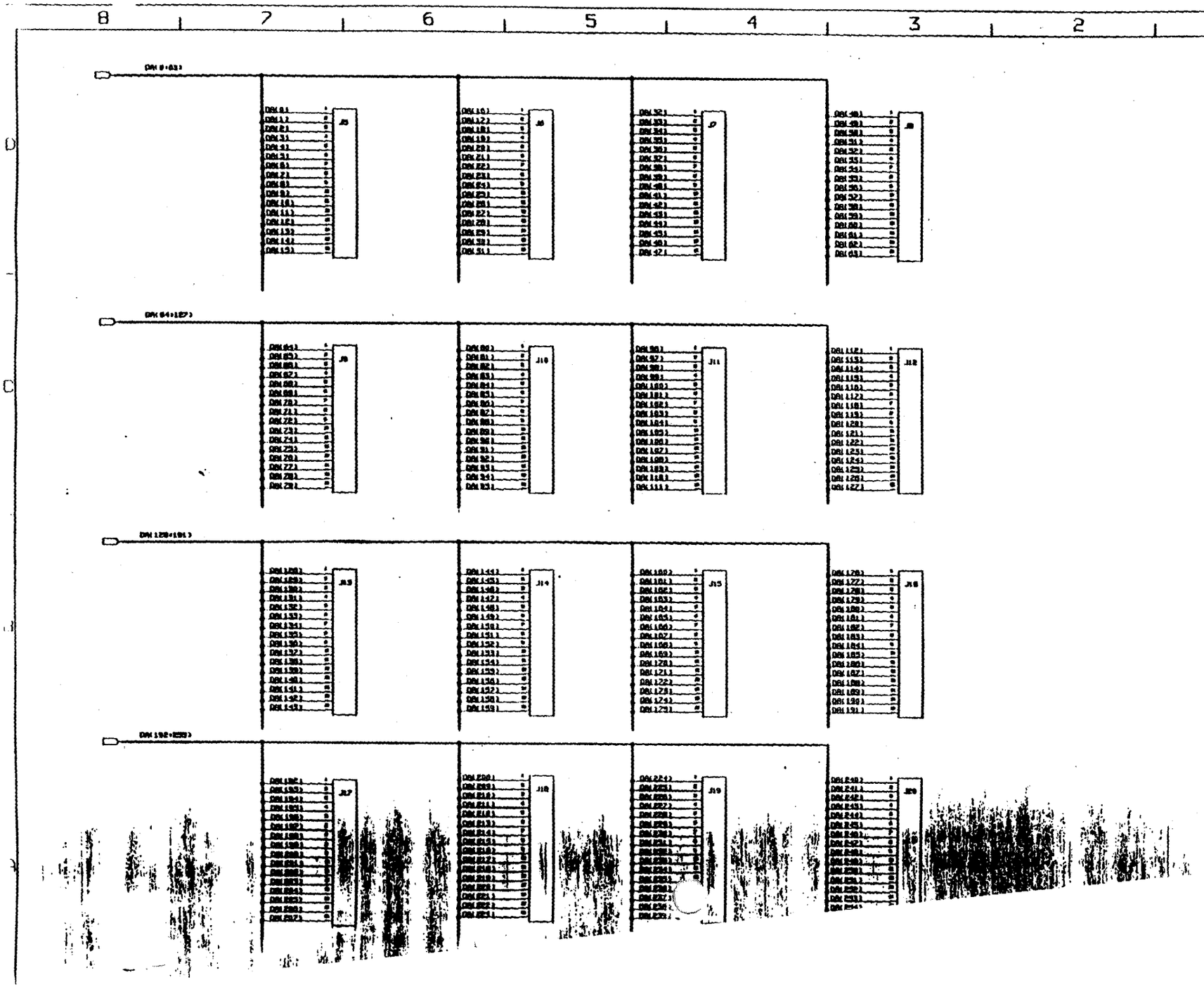
E771 DISCRIMINATOR
DISCRIMINATOR INPUTS
FROM PREAMP


8
7
6
5
4
3
2
1
D
C
B
A
POST AMP COMPARATOR


FASTBUS INTERFACE
FASTBUS TRANSLATORS
E771 DISCRIMINATOR
FASTBUS INTERFACE
FASTBUS TRANSLATORS


FASTBUS INTERFACE
CONTROL LOGIC
SHEET #26

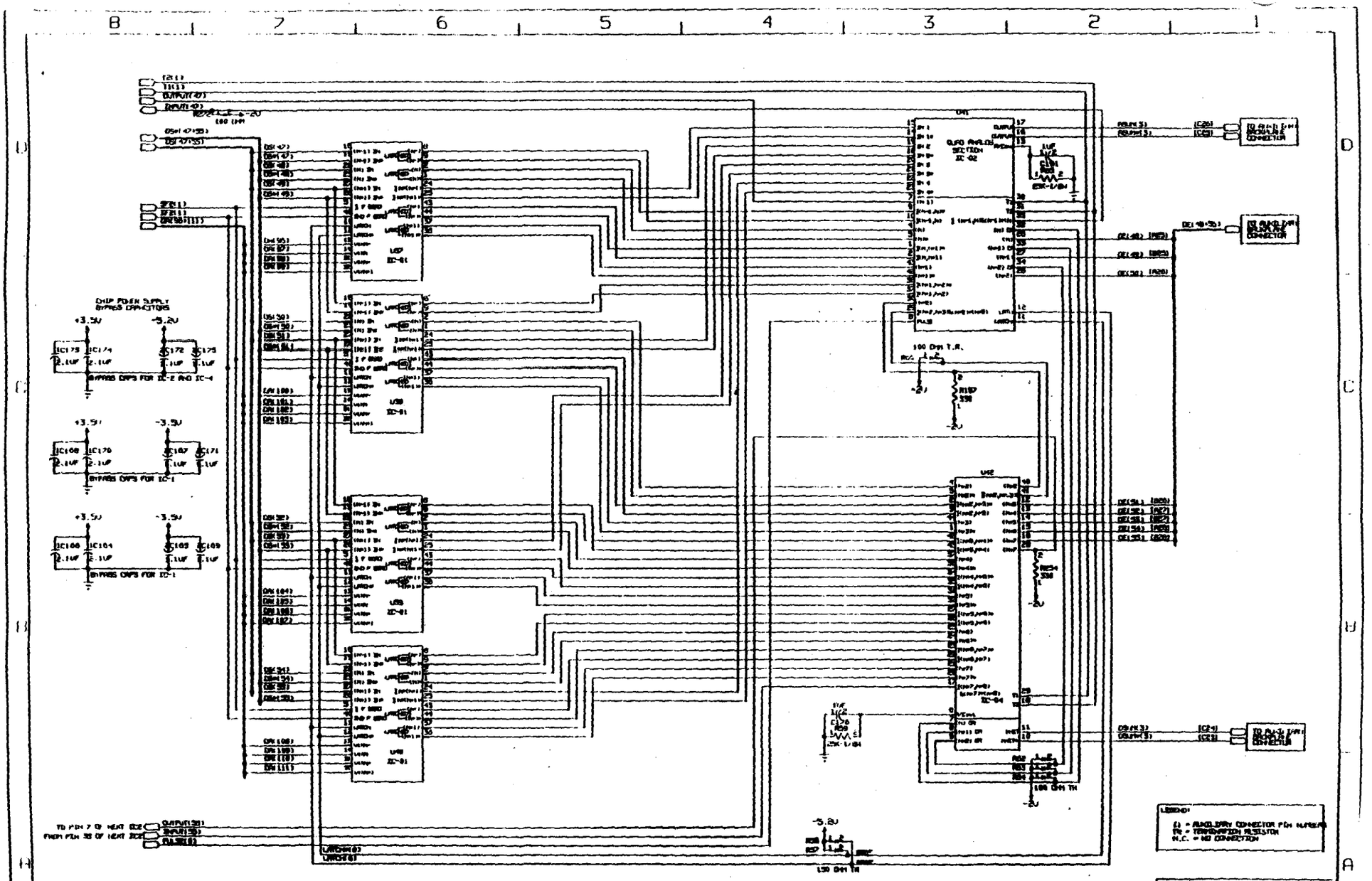

8
7
6
5
4
3
2
1
D
C
B
A
OF7
OF8
OF9
FIDUCIALS
OT3
OT4
OT5
OT6
TOOLING HOLES
NECESSARY TO MANUFACTURE
THE PRINTED CIRCUIT BOARD
A
REV
K
04/24/91


8
7
6
5
4
3
2
1
D
C
B
A

VERSION	NOTES	SHEET #
A		
B		
C		
D		
E		
F	REVERSED INPUTS TO U192 AND CHANGED TO DAC READ	24
G	SWAPPED LABELS FOR E/B AUX CONN C14 & C15	21
H	REMOVED ANALOG GROUND FROM GROUND CONNECTIONS AND REMOVED GROUND FROM F/B AUX CONN C50 ADDED C10	25 21
I	ADDED GNDS C9,C30,C49,C50 C62,C63	21
J	ADDED DELAY CIRCUITRY AND ROTARY SWITCH MODIFIED LED DRIVER	24
K	CHANGED RESISTORS FROM 100 TO 50 OHMS AND MODIFIED BUFFER U186	1,5,9,13 26

E771 DISCRIMINATOR FASTBUS INTERFACE CONTROL LOGIC

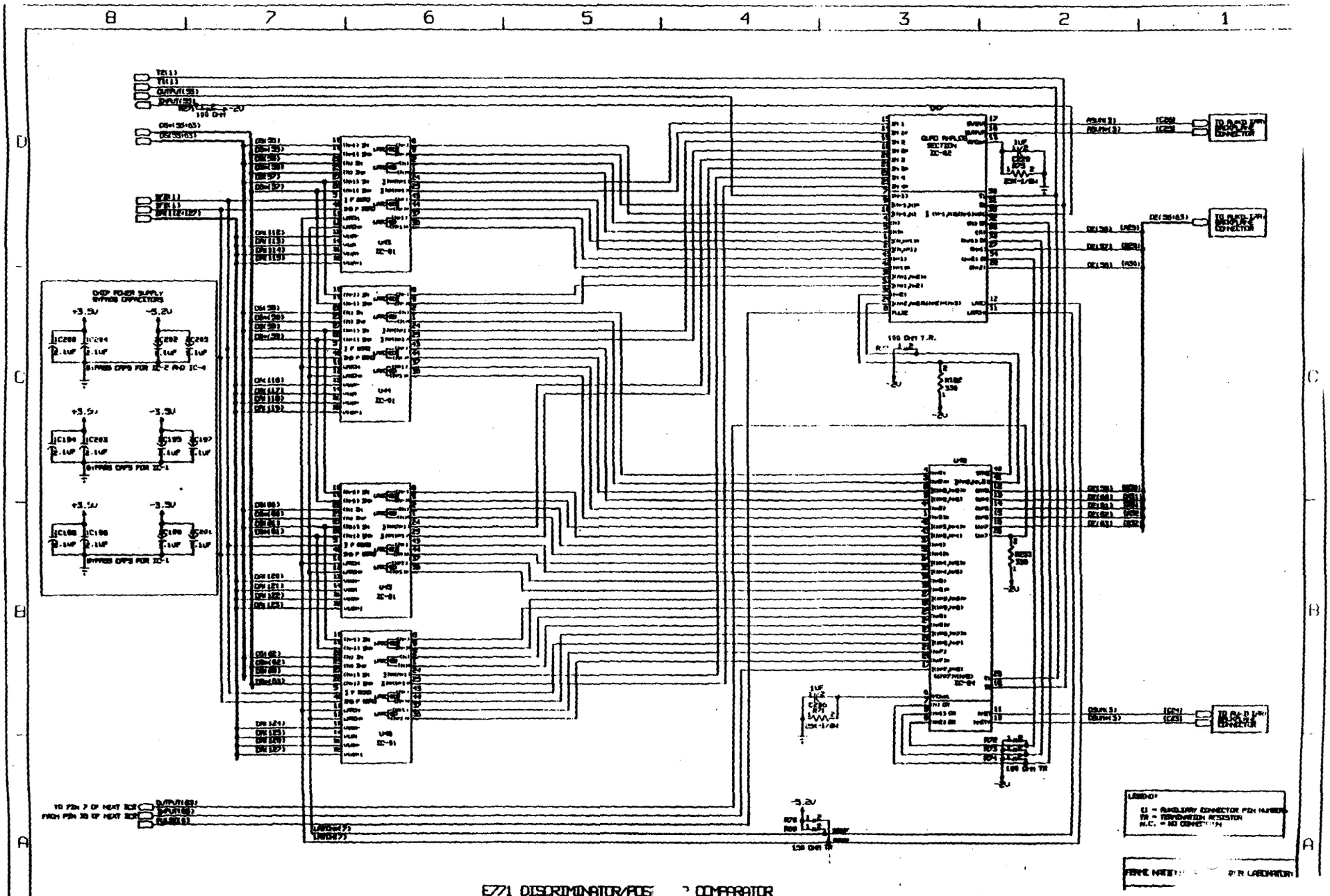
A	NEUMANN HALDEMAN		
			REV K 04/24/91



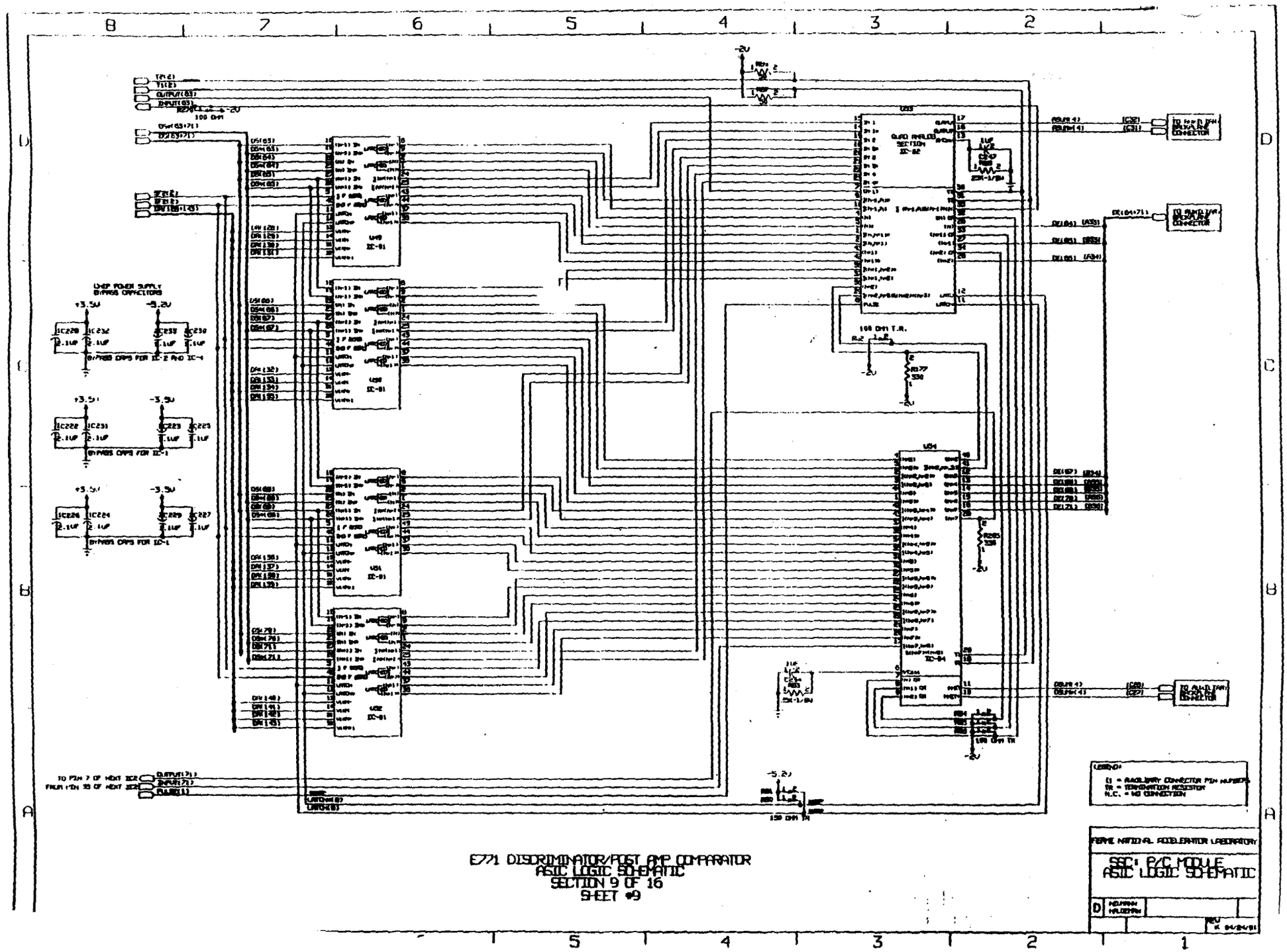
E771 DISCRIMINATOR/POST AMP COMPARATOR
ASIC LOGIC SCHEMATIC
SECTION 7 OF 16
SHEET #7

FERMI NATIONAL ACCELERATOR LABORATORY

SSC P/C MODULE
ASIC LOGIC SCHEMATIC



E771 DISCRIMINATOR/POS[illegible] COMPARATOR
ASIC LOGIC S[illegible]IC
SECTION 8 (1)
SHEET [illegible]

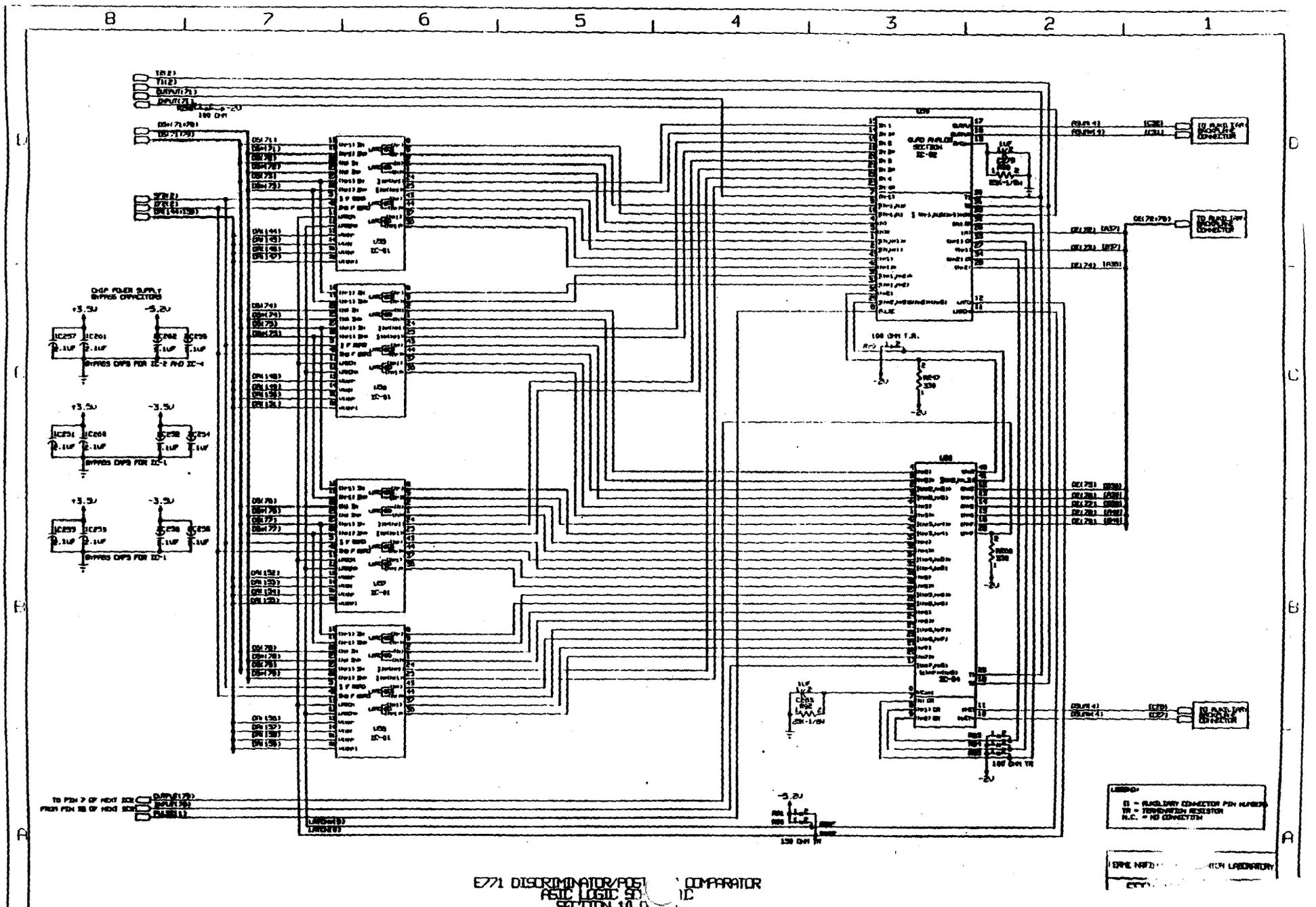


E771 DISCRIMINATOR/POST AMP COMPARATOR
ASIC LOGIC SCHEMATIC
SECTION 9 OF 16
SHEET #9

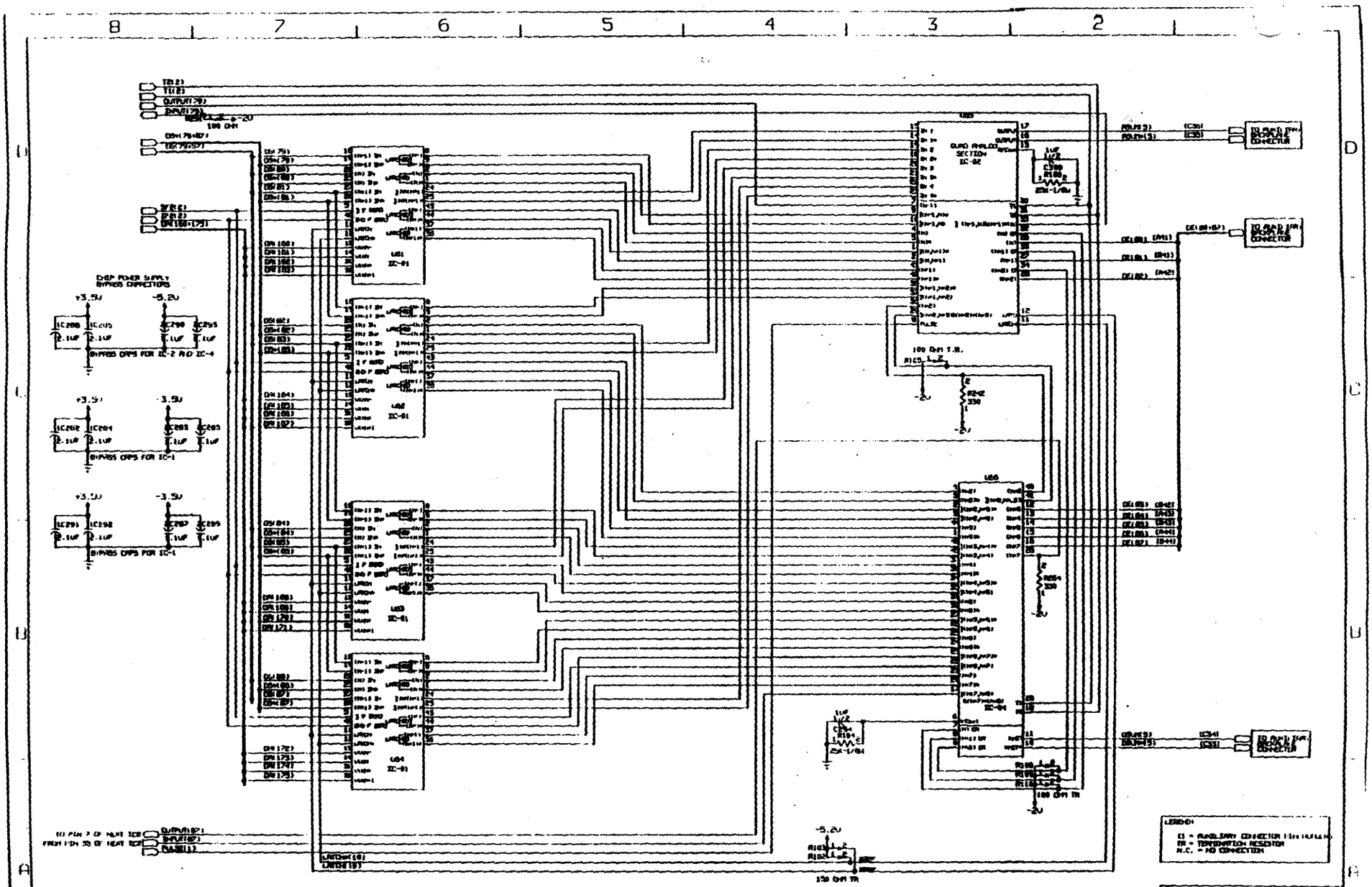
FERMI NATIONAL ACCELERATOR LABORATORY

SSC P/C MODULE
ASIC LOGIC SCHEMATIC

D



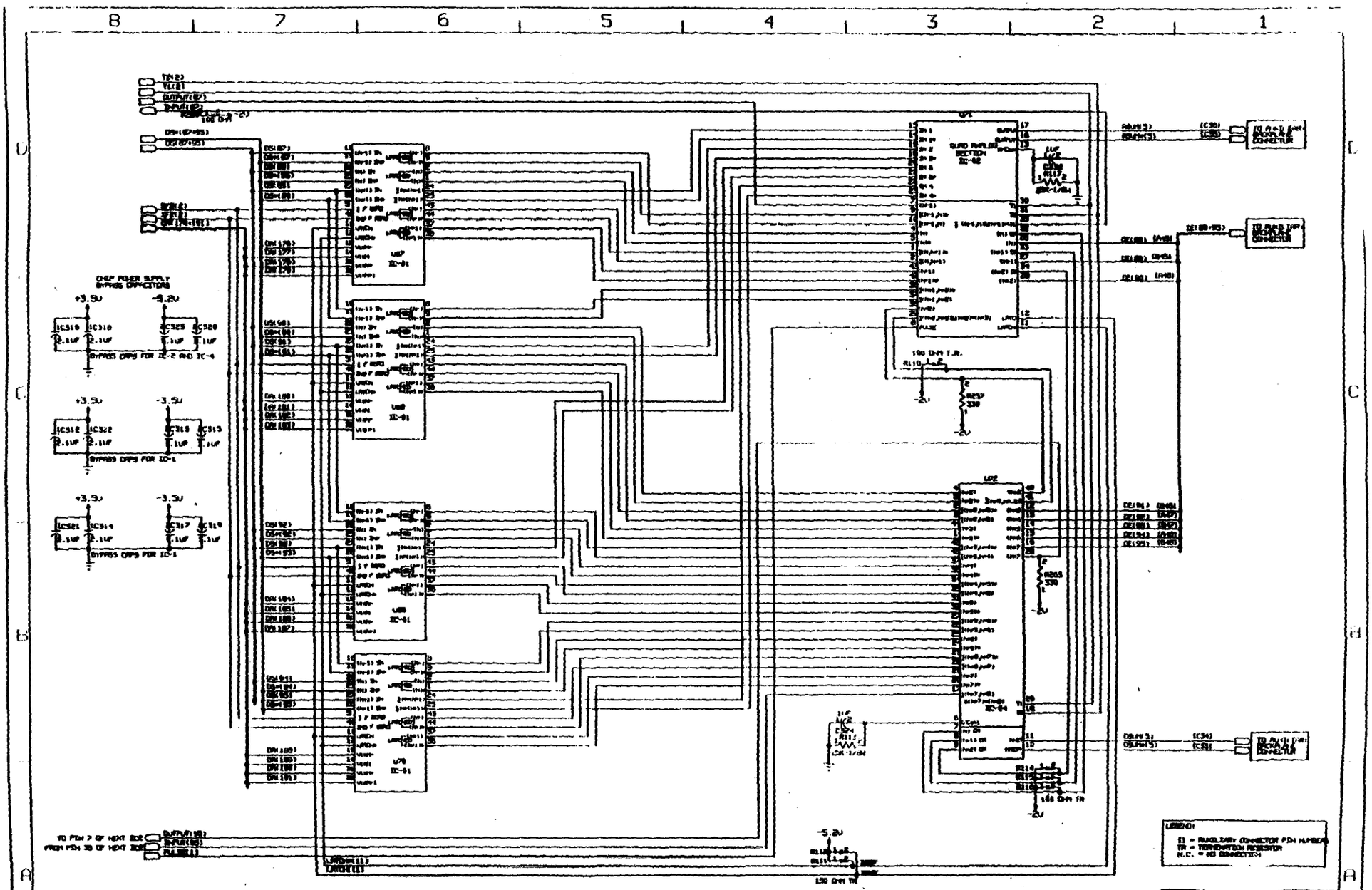
E771 DISCRIMINATOR/POST[illegible] COMPARATOR
ASIC LOGIC SC[illegible]IC
SECTION 10 [illegible]
SHEET [illegible]



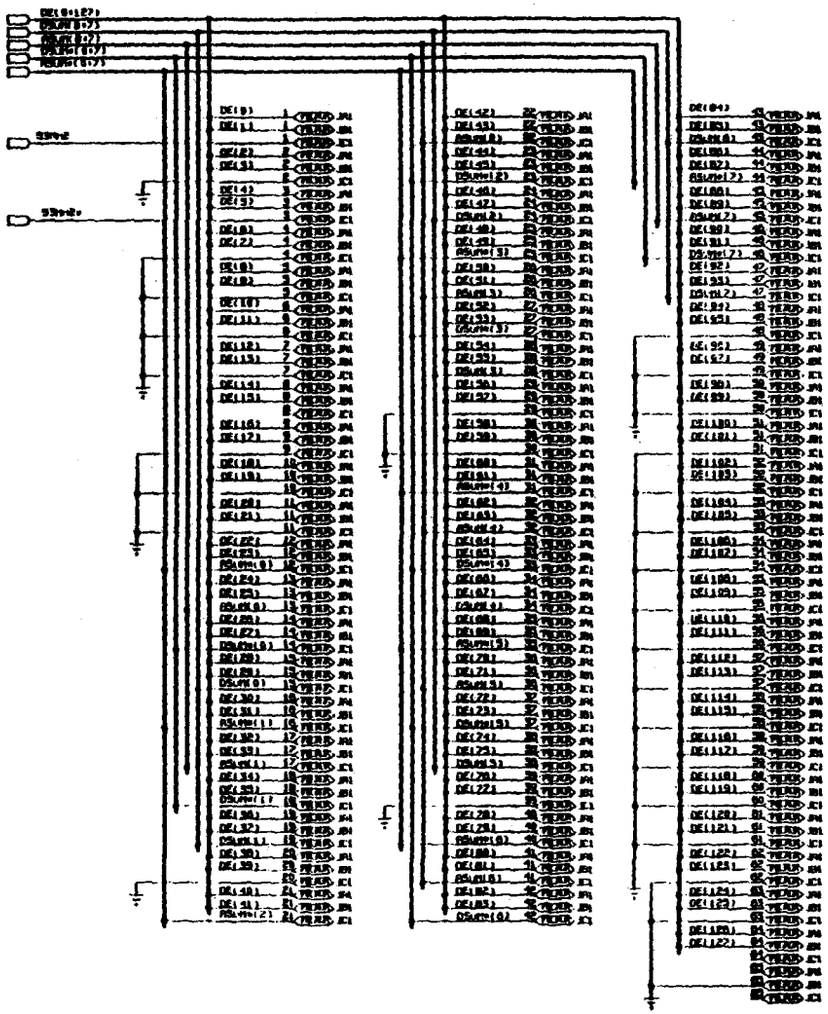
E771 DISCRIMINATOR/POST AMP COMPARATOR
ASIC LOGIC SCHEMATIC
SECTION 11 OF 16
SHEET #11

FERMI NATIONAL ACCELERATOR LABORATORY

SSC P/C MODULE
ASIC LOGIC SCHEMATIC

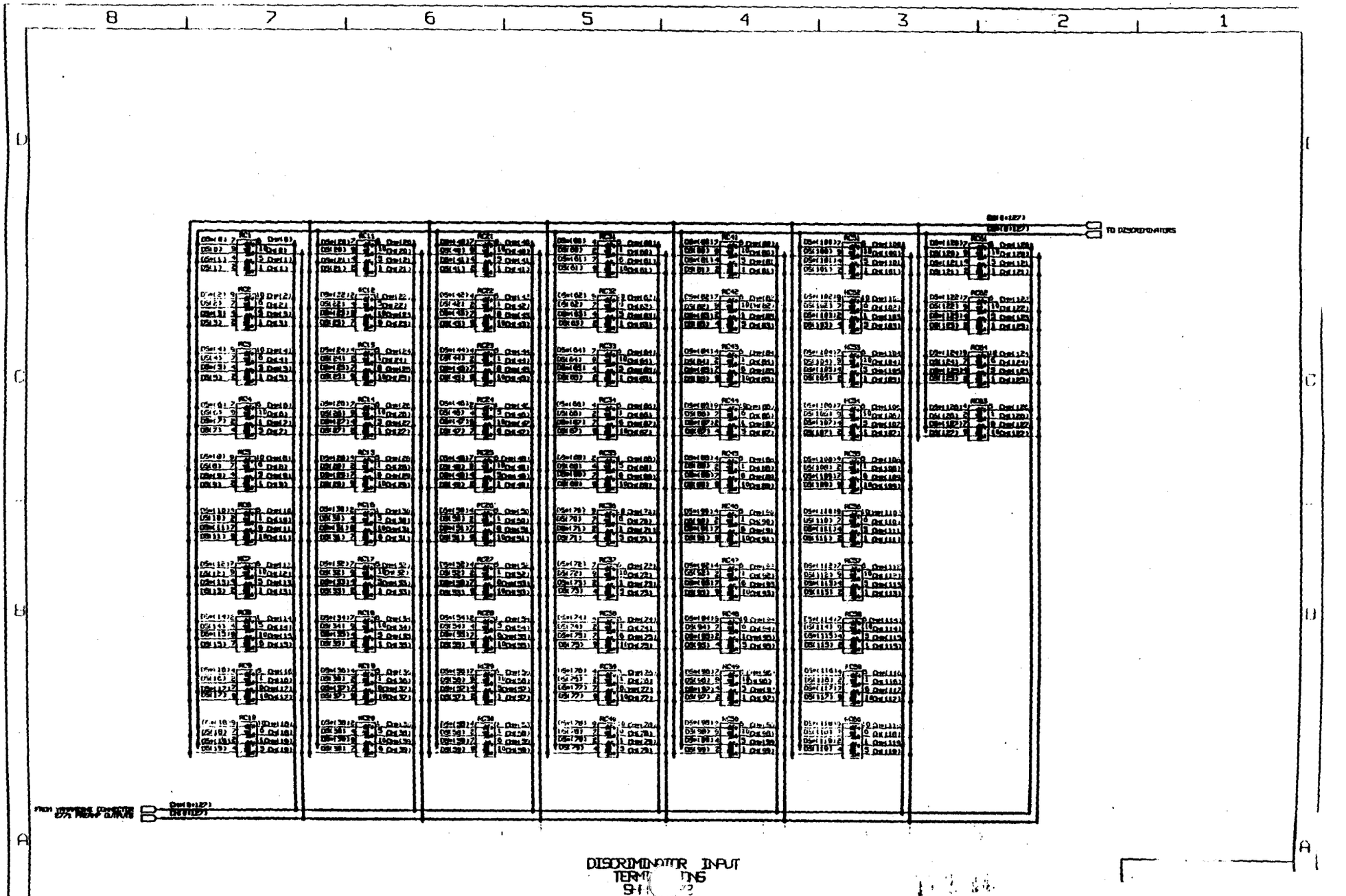


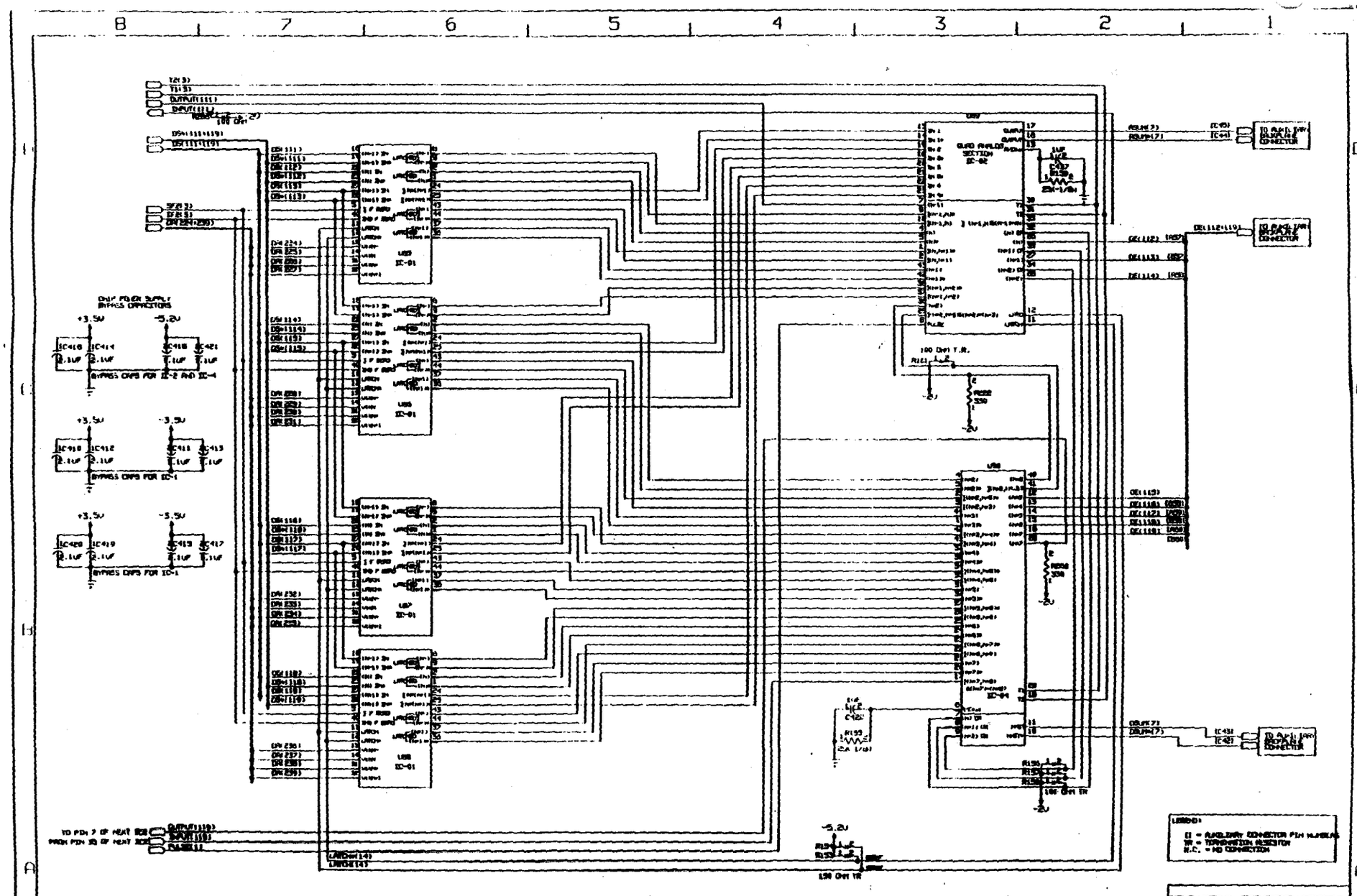
E771 DISCRIMINATOR/POST COMPARATOR
ASIC LOGIC SCH
SECTION 12 OF
SHEET #12



FASTBUS AUXILIARY CONNECTOR
SHEET #21

E771 DISCRIMINATOR
FASTBUS AUXILIARY CONNECTOR

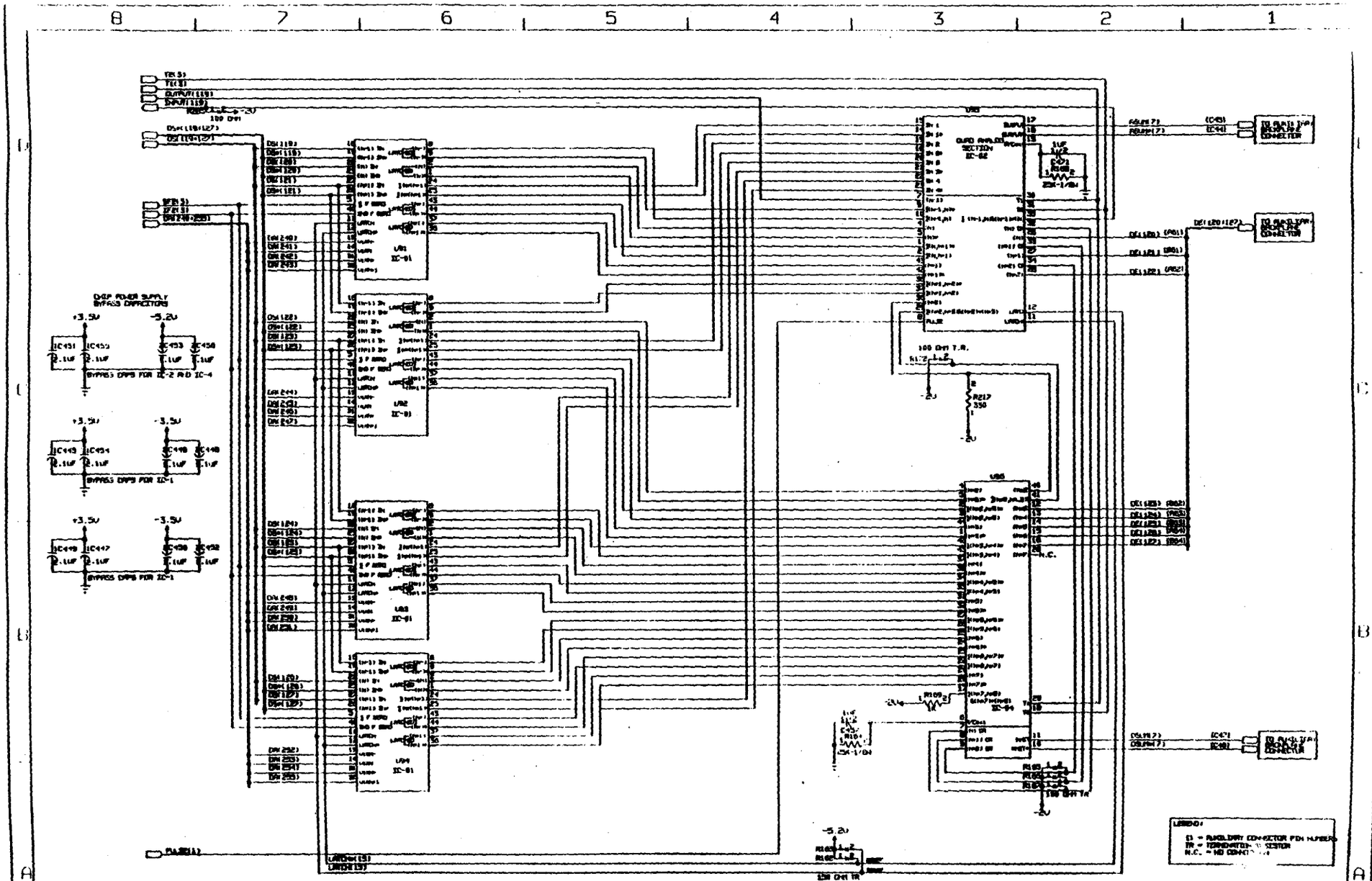

8
7
6
5
4
3
2
1
D
C
B
A
TO DISCRIMINATORS
DISCRIMINATOR INPUT
TERMINATIONS



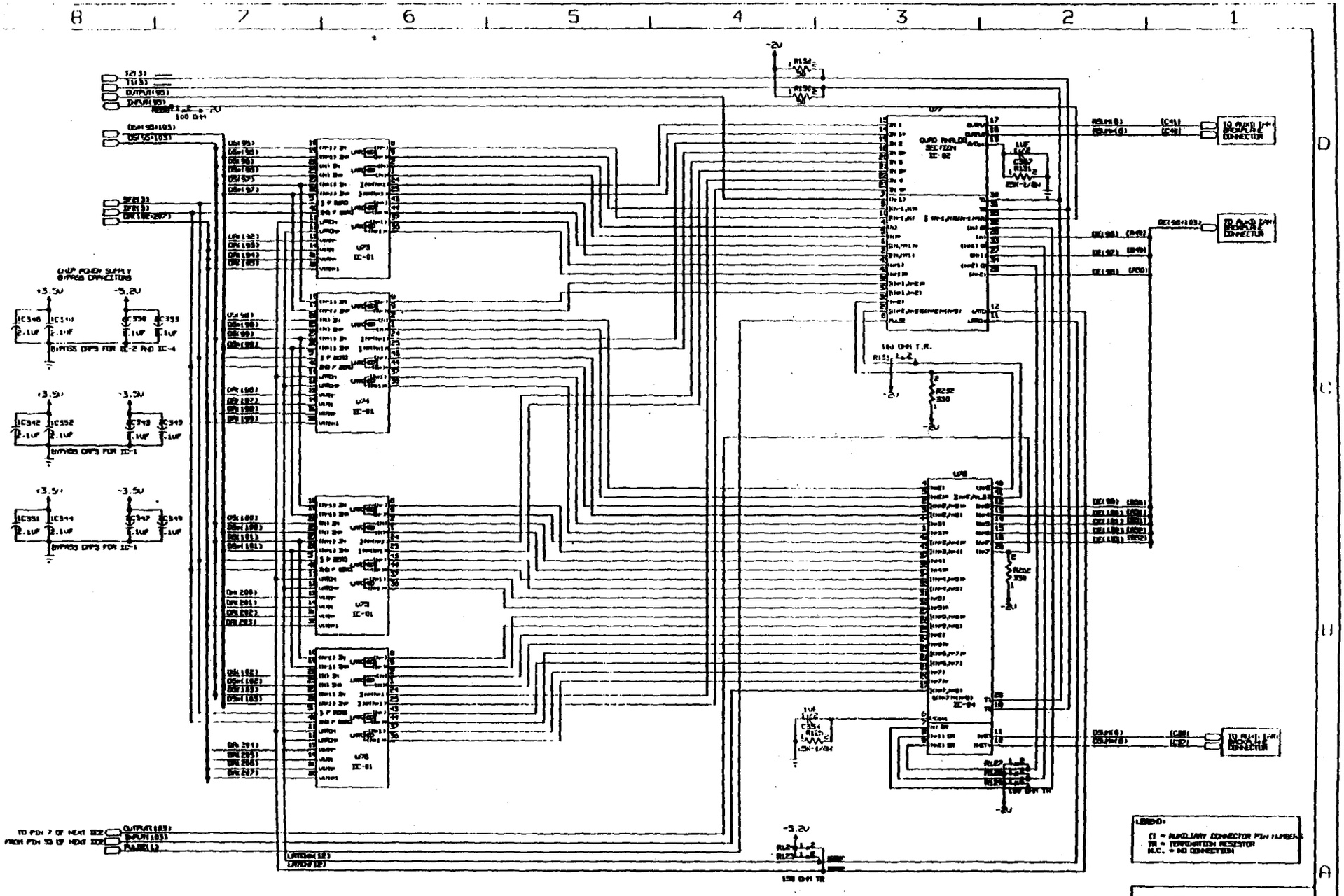
E771 DISCRIMINATOR/POST AMP COMPARATOR
ASIC LOGIC SCHEMATIC
SECTION 15 OF 16
SHEET #15

FERMI NATIONAL ACCELERATOR LABORATORY

SSC P/C MODULE
ASIC LOGIC SCHEMATIC



E771 DISCRIMINATOR/PDL[illegible] COMPARATOR
ASIC LOGIC SD[illegible] IC
SECTION 16 OF [illegible]

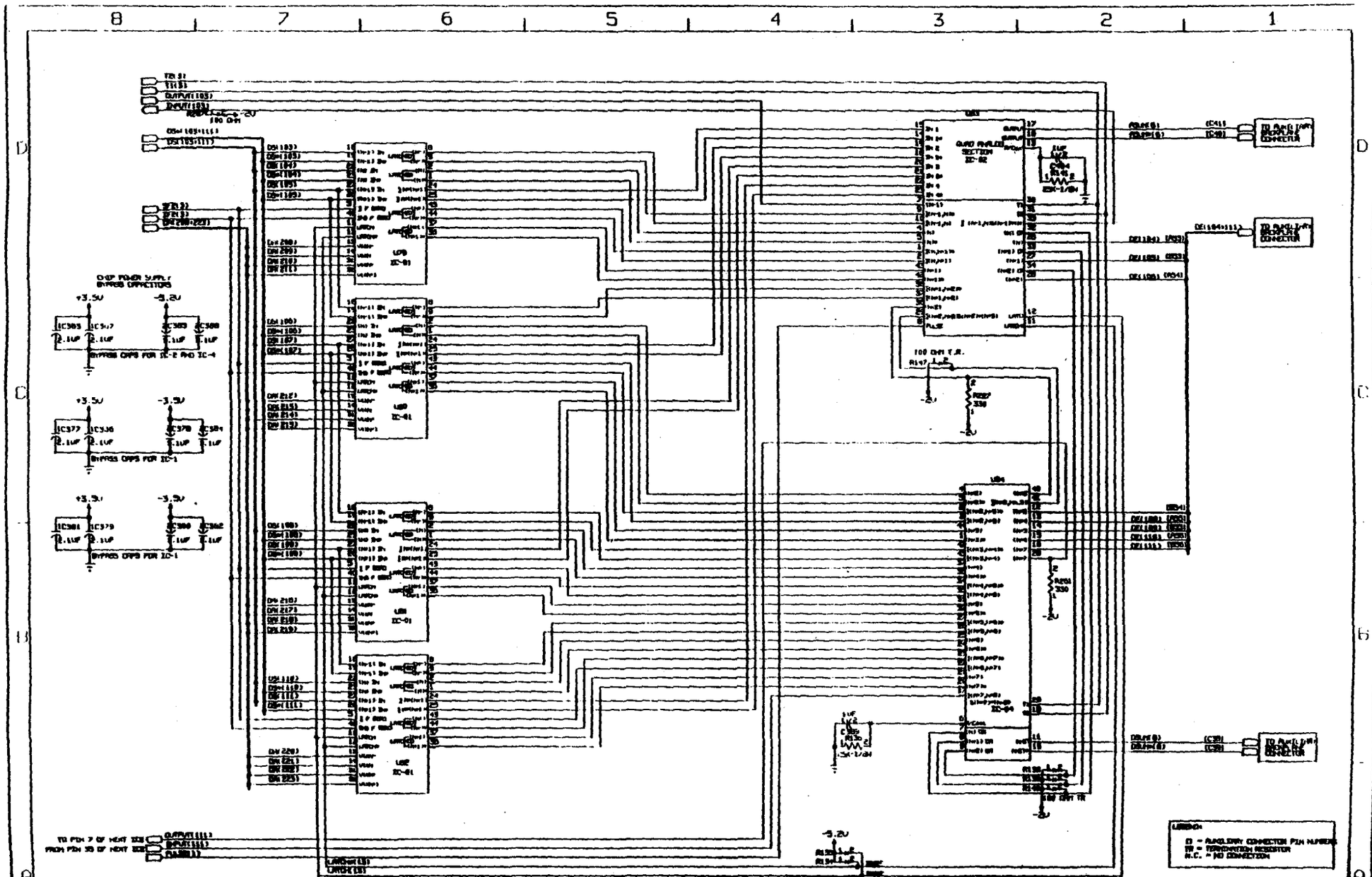


E771 DISCRIMINATOR/POST AMP COMPARATOR
ASIC LOGIC SCHEMATIC
SECTION 13 OF 16
SHEET #13

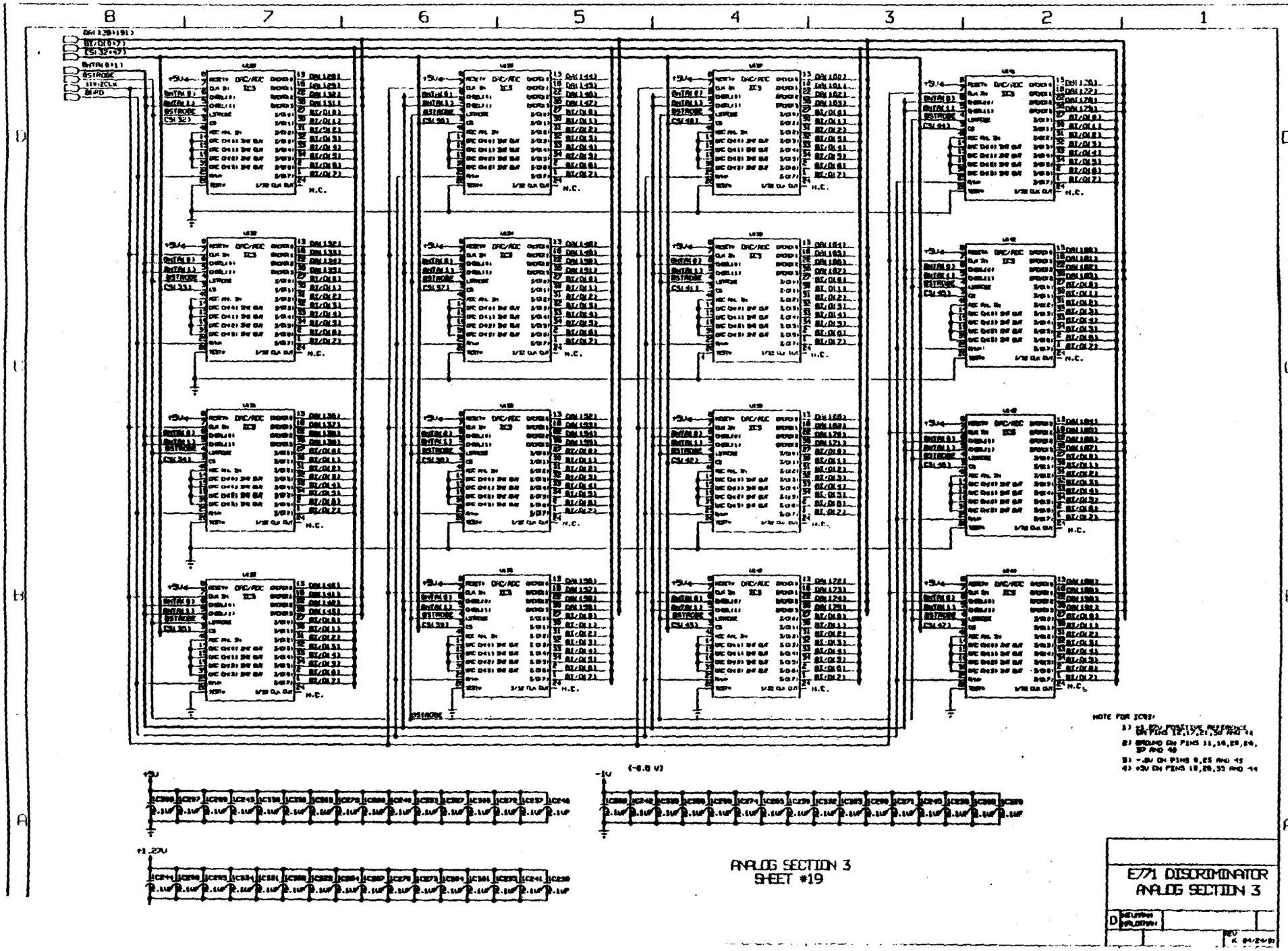
FERMI NATIONAL ACCELERATOR LABORATORY

SSC P/C MODULE
ASIC LOGIC SCHEMATIC

D

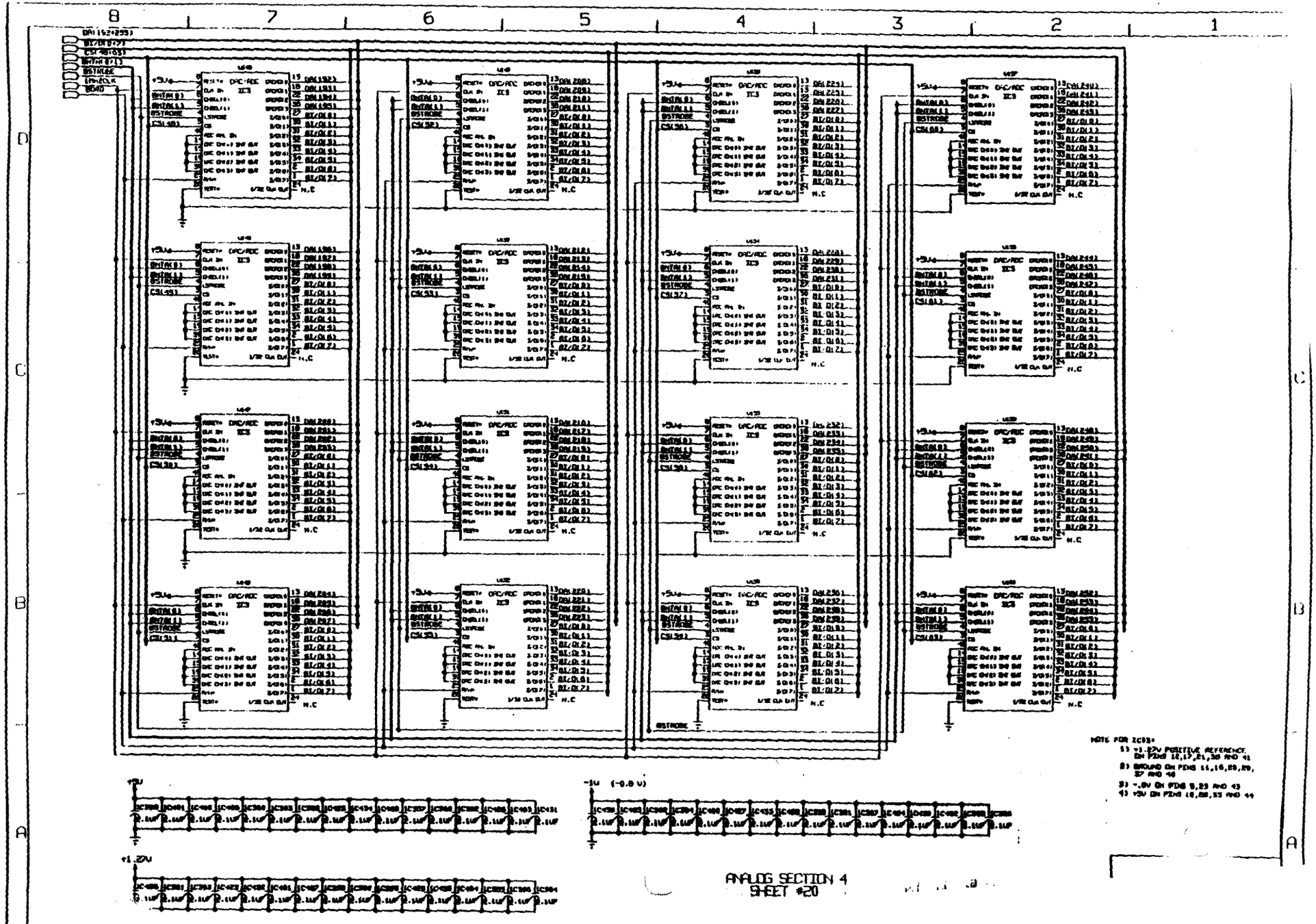


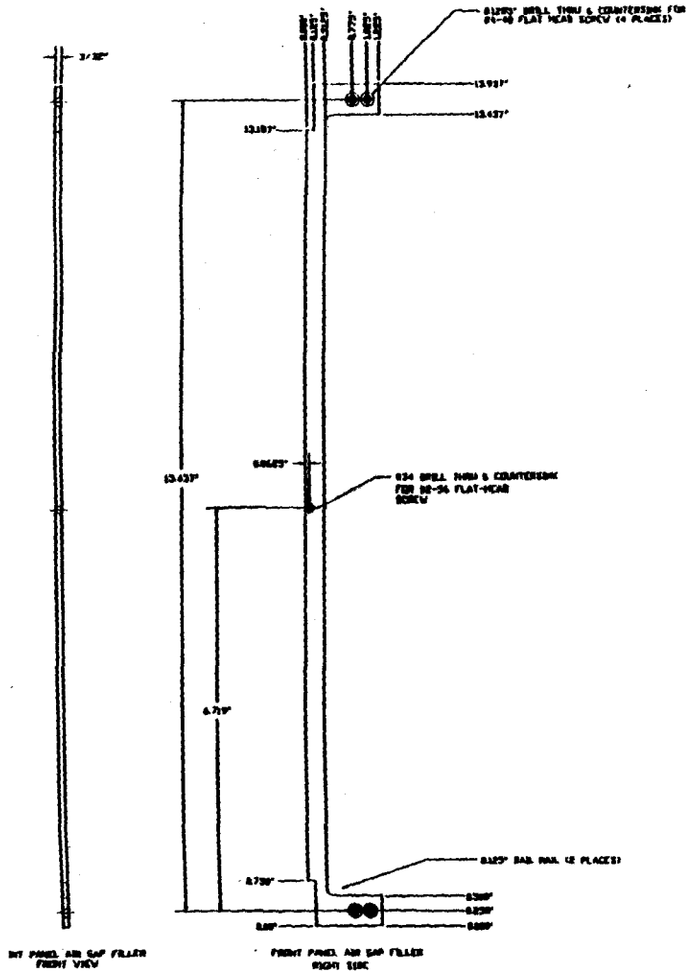
E771 DISCRIMINATOR/POST AMP COMPARATOR
ASIC LOGIC SCHEMATIC
SECTION 14
SHEET



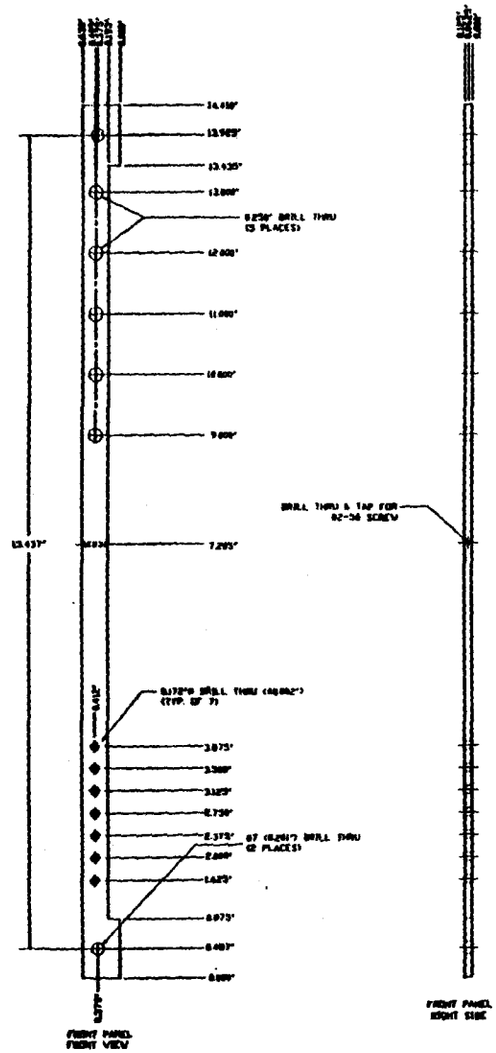
ANALOG SECTION 3
SHEET #19

E771 DISCRIMINATOR
ANALOG SECTION 3


8
7
6
5
4
3
2
1
D
C
B
A
+5V
+1.2V
-5V (-6.8 V)
NOTE FOR IC03:
1) +1.22V POSITIVE REFERENCE ON PINS 12,17,21,36 AND 41
2) GROUND ON PINS 11,16,20,29, 37 AND 40
3) -6V ON PINS 9,23 AND 43
4) +5V ON PINS 10,38,33 AND 44
ANALOG SECTION 4
SHEET #20



E771 - POST AMP COMPARATOR FRONT PANEL AIR GAP FILLER



E771 - POST AMP COMPARATOR FRONT PANEL DETAILS

D FRONT PANEL SILKSCREEN - 2563A00-MB-015753

ITEM NO.	PART NO.	DESCRIPTION OR SIZE
		PARTS LIST

UNLESS OTHERWISE SPECIFIED			DRAWN	S. HALBERMAN	03
DIMENSIONS	DECIMALS	ANGLES	TRACED	S. A. HEMMAN	03
±1/64"	0.005"	±1/4°	CHECKED		
1. BREAK ALL SHARP EDGES 1/64 MAX.			APPROVED	S. HALBERMAN	04
2. ON ONE SIDE ONLY.					
3. DIMENSIONS IN ( ) ARE FOR REFERENCE ONLY.					
4. SMALL GAP BETWEEN SURFACES					

SIZE	CAGE CODE
D	2563A00-MB-015753
MATERIAL	
1/8" 6061-T6 ALUMINUM	

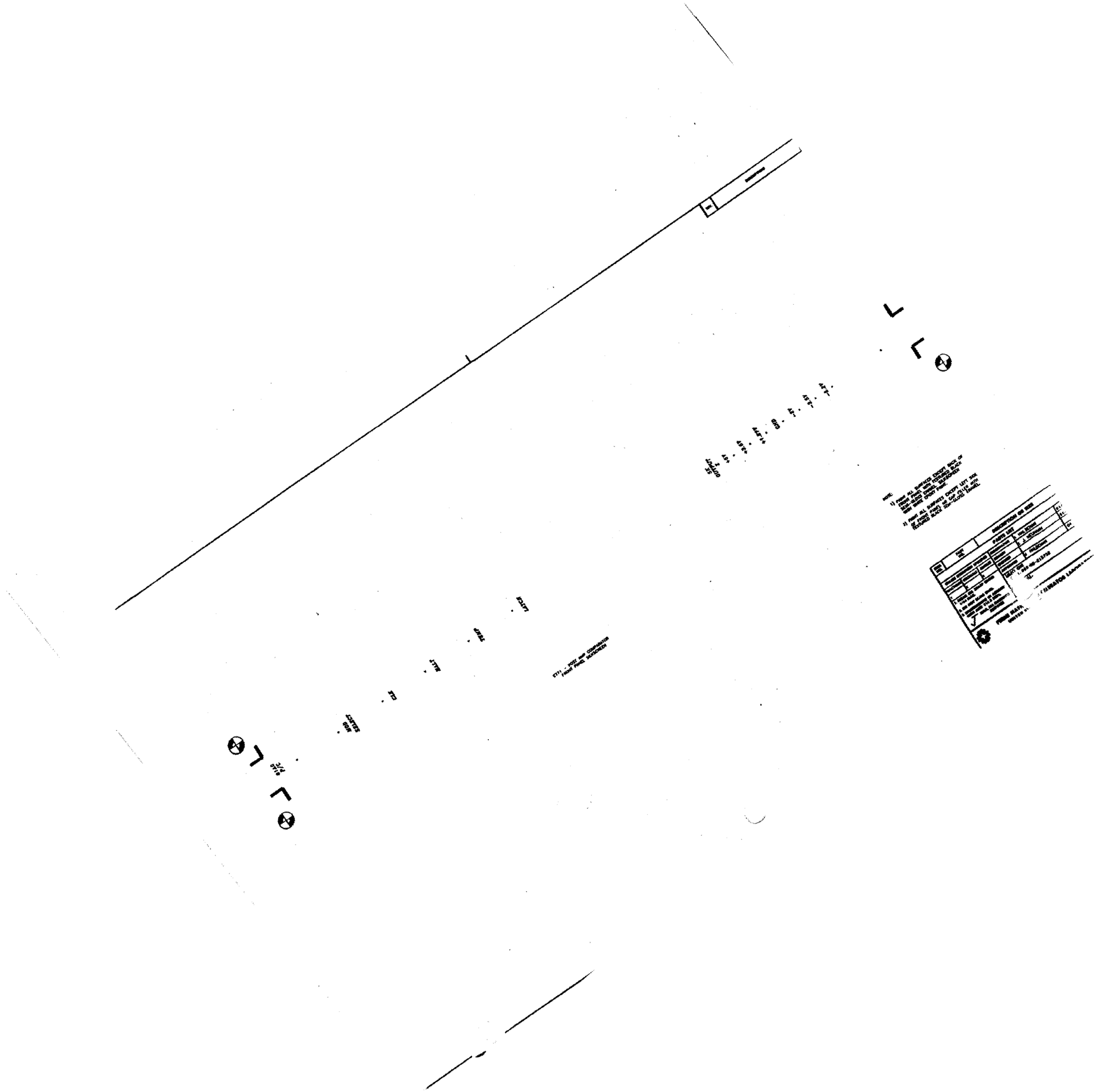
FERMI NATIONAL ACCELERATOR LABORATORY
UNITED STATES DEPARTMENT OF ENERGY

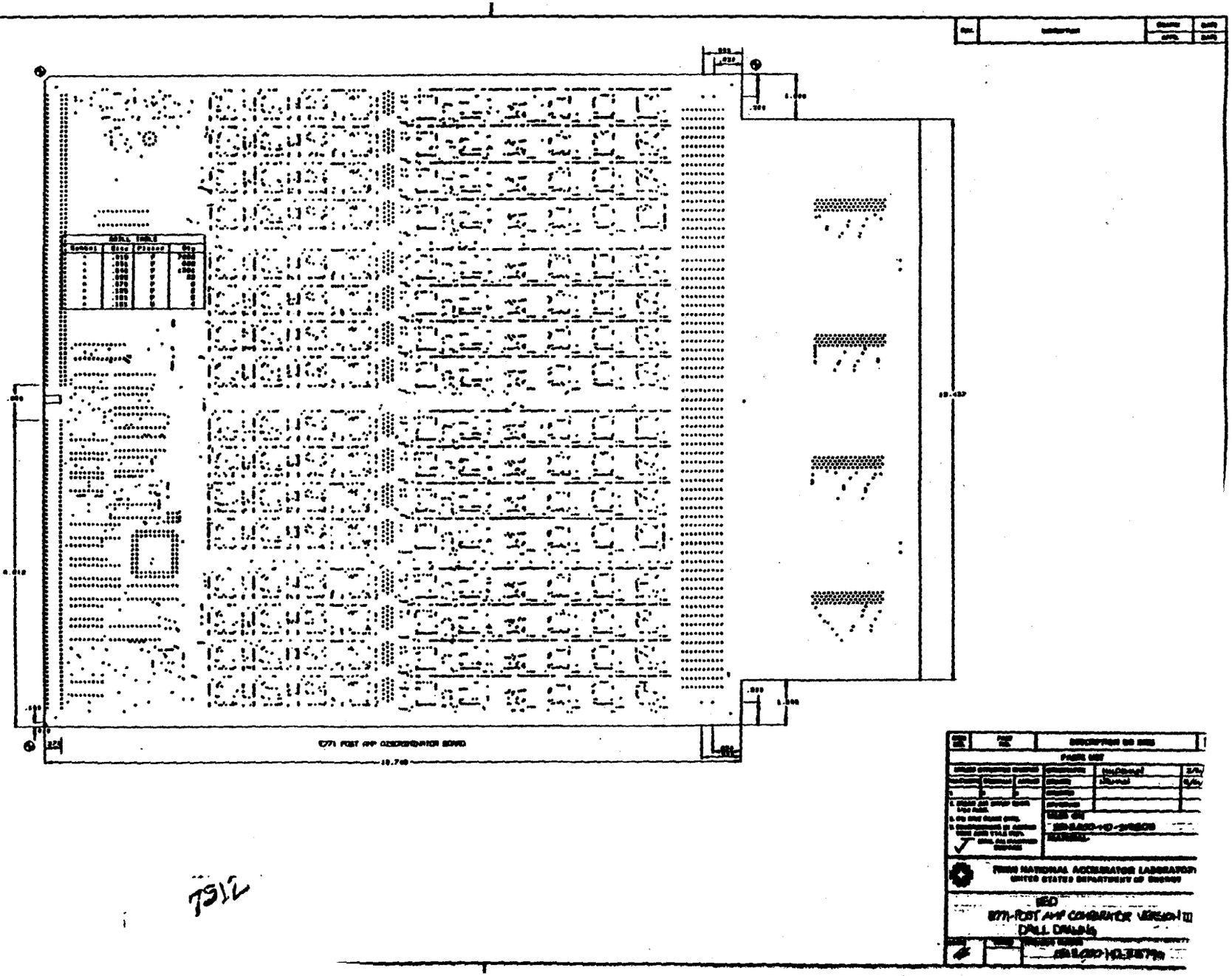
CED

E771 - POST AMP COMPARATOR VER III

FRONT PANEL & FRONT PANEL AIR GAP FILLER

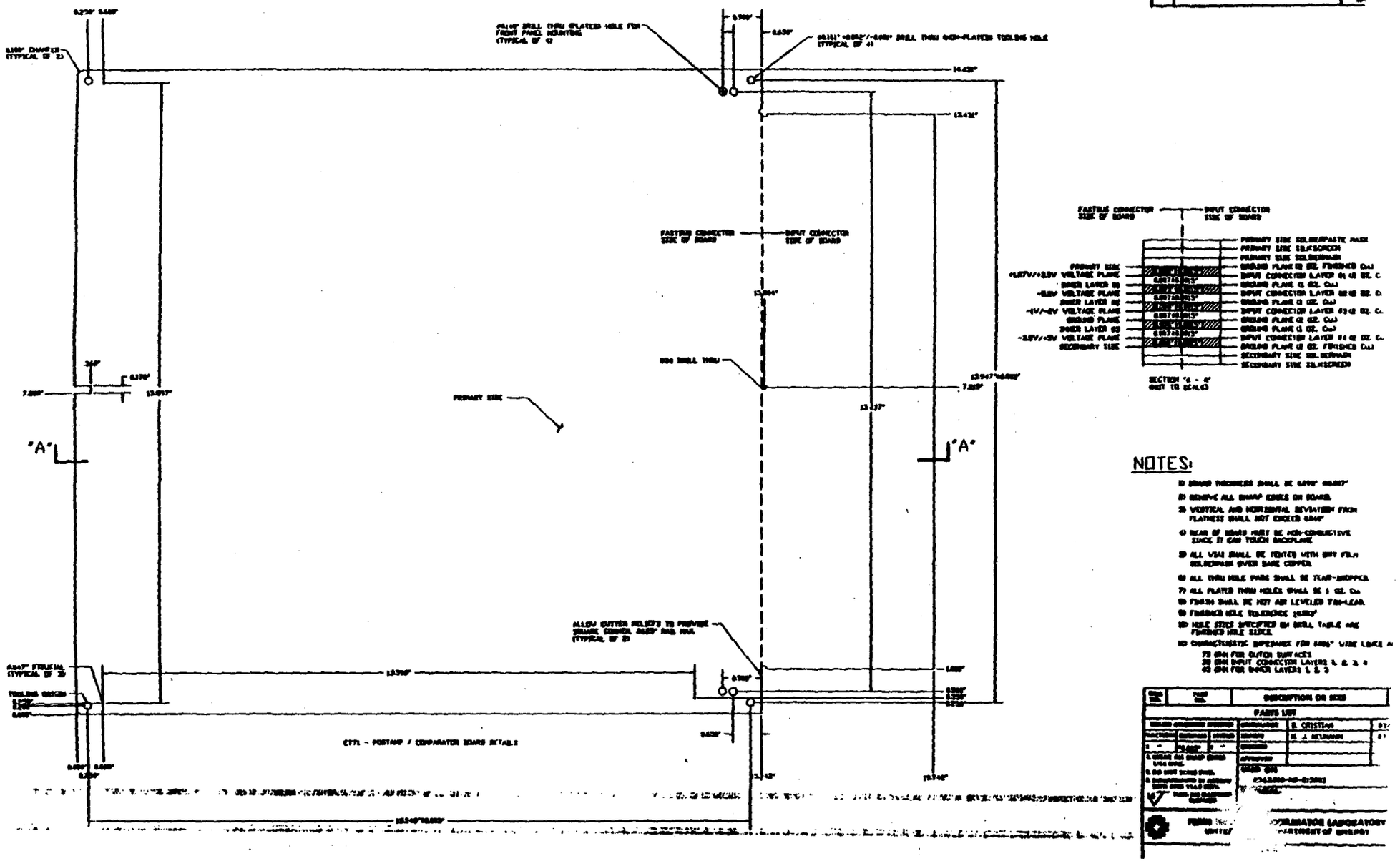
SCALE	SHEET	DRAWING NUMBER
1"=1"		2563A00-MB-015752



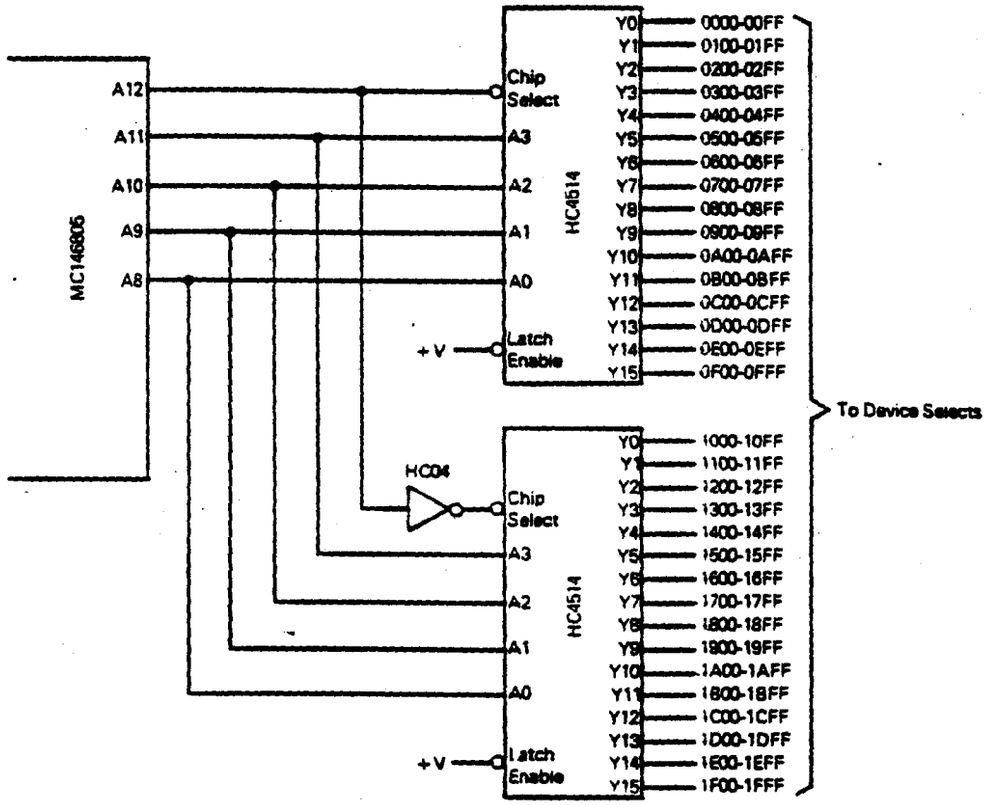

DRILL TABLE
8771 POST AMP DEGENERATOR BOARD
FERMI NATIONAL ACCELERATOR LABORATORY
8771-POST AMP COMPARATOR VERSION III
DRILL DRAWING

24

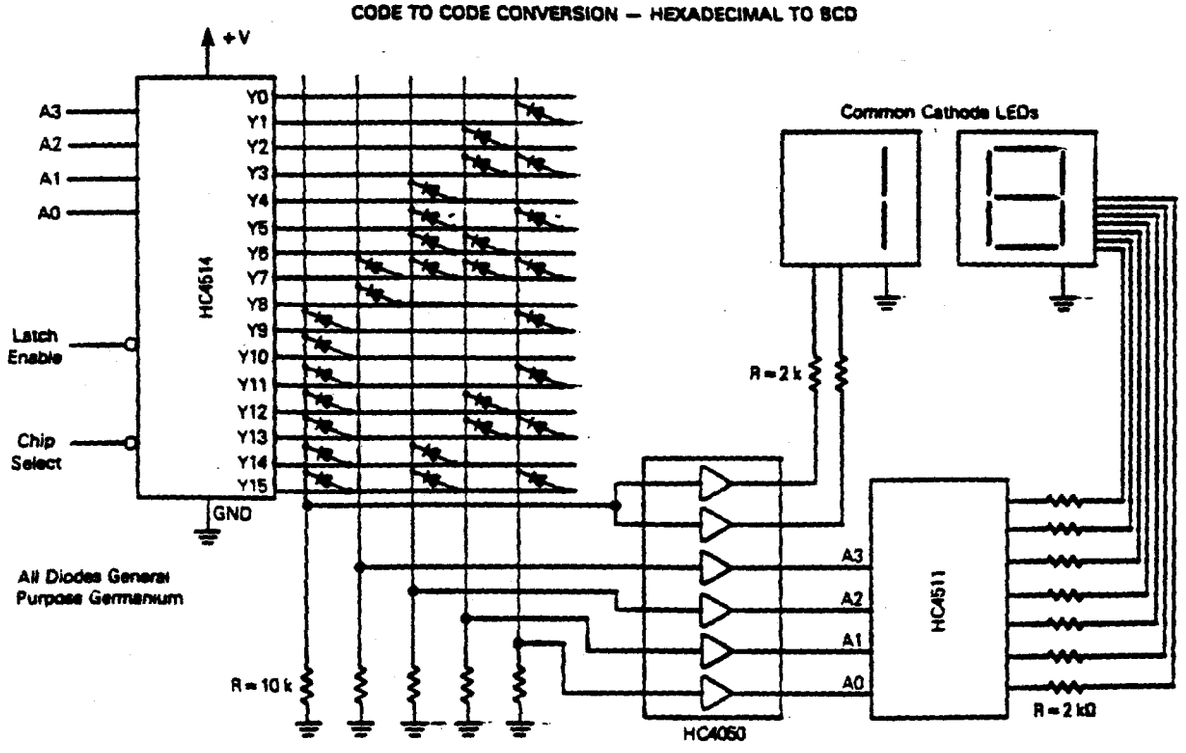
7512


FASTBUS CONNECTOR SIDE OF BOARD
INPUT CONNECTOR SIDE OF BOARD
PRIMARY SIDE
SECONDARY SIDE
SECTION "A - A"
"A"
"A"
NOTES:
PARTS LIST
S. CRISTIAN
M. J. MEHMANN
FERMI NATIONAL ACCELERATOR LABORATORY

MICROPROCESSOR MEMORY DECODING



CODE TO CODE CONVERSION — HEXADECIMAL TO BCD



Bt501
Bt502

ECL / TTL Octal Transceiver and Translator

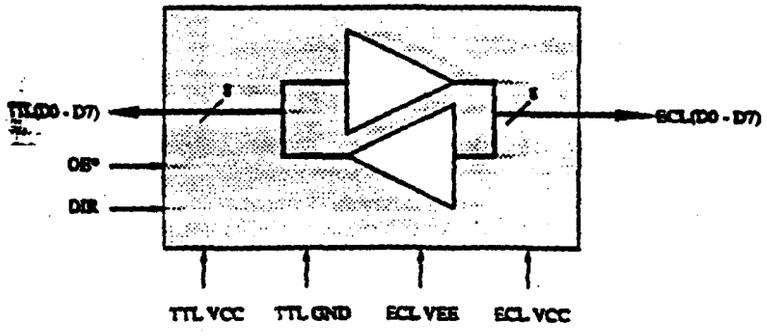
Distinguishing Features

- 10KH or 100K ECL Compatible
- Optional Single +5 V Operation
- Separate TTL and ECL Supply Pins
- Three-Statable TTL Pins
- TTL-Compatible Control Inputs
- 24-pin 0.3" DIP Package
- Typical Power Dissipation: 800 mW

Benefits

- Flexible Power Supply
- Reduced Component Count
- Simplifies PCB Layout
- Reduces PCB Interconnect
- Low Bus Loading

Functional Block Diagram

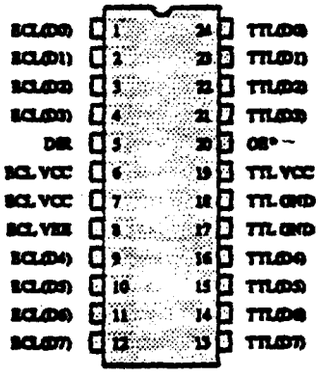


Product Description

The Bt501 and Bt502 are octal ECL/TT bidirectional transceivers and translators. TI Bt501 is 10KH ECL compatible, and the Bt502 100K ECL compatible.

The direction and output enable control inputs ar TTL compatible to simplify interfacing to a standard MPU.

Both devices provide a bidirectional interface between TTL signals and ECL signals. The ECL input/output signals may be generated from normal ECL, single +5 V, or split ECL supplies.



Brooktree Corporation
9950 Barnes Canyon Rd.
San Diego, CA 92121
(619) 452-7580 • (800) VIDEO IC
TLX: 383 596 • FAX: (619) 452-1249
L501001 Rev. F

Brooktree®

5
5

Circuit Description

	Nominal Voltages Applied		
Supply Pin	Single Supply System	Dual Supply System	Split ECL Supply System
TTL VCC	+5.0 V	+5.0 V	+5.0 V
TTL GND	0 V	0 V	0 V
ECL VCC	+5.0 V	0 V	+2.0 V
ECL VEE	0 V	-5.2 V	-3.2 V

Bt501 Supply Operation.

	Nominal Voltages Applied		
Supply Pin	Single Supply System	Dual Supply System	Split ECL Supply System
TTL VCC	+5.0 V	+5.0 V	+5.0 V
TTL GND	0 V	0 V	0 V
ECL VCC	+5.0 V	0 V	+2.0 V
ECL VEE	0 V	-4.5 V	-2.5 V

Bt502 Supply Operation.

Note: The TTL (D0–D7), DIR, and OE* pins are TTL compatible regardless of the ECL power supply parameters. Changing the ECL power supply parameters affects the threshold levels of only the ECL (D0–D7) pins.

DIR	OE*	Function
0	0	TTL (D0–D7) --> ECL (D0–D7)
1	0	ECL (D0–D7) --> TTL (D0–D7)
x	1	TTL (D0–D7) three-stated, ECL (D0–D7) = 0

Control Truth Table.

Bt501—Recommended Operating Conditions

Parameter	Symbol	Min	Typ	Max	Units
TTL Device Ground	TTL GND	0	0	0	Volts
ECL Device Ground	ECL VCC	0	0	0	Volts
TTL Power Supply	TTL VCC	+4.75	+5.0	+5.25	Volts
ECL Power Supply	ECL VEE	-4.9	-5.2	-5.5	Volts
Ambient Operating Temperature	TA	0		+70	°C

Note: Thermal equilibrium is established by applying power for at least 2 minutes while maintaining a transverse air flow of 400 linear feet per minute over the device either mounted in the test socket or on the printed circuit board.

Bt501—Absolute Maximum Ratings

Parameter	Symbol	Min	Typ	Max	Units
ECL VEE (measured to ECL VCC)				-8.0	Volts
TTL VCC (measured to TTL GND)				+7.0	Volts
Voltage on Any ECL Pin		ECL VCC		ECL VEE	Volts
Voltage on Any TTL Pin		TTL GND -0.5		TTL VCC +0.5	Volts
ECL(D0–D7) Output Current				-50	mA
TTL(D0–D7) Short Circuit Output Current	IOS	-50		-150	mA
Ambient Operating Temperature	TA	-55		+125	°C
Storage Temperature	TS	-65		+150	°C
Junction Temperature	TJ			+175	°C
Soldering Temperature (5 seconds, 1/4" from pin)	TSOL			260	°C

Note: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those listed in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Bt501—ECL DC Characteristics

Parameter	Symbol	TA (°C)	Min	Typ	Max	Units
Input High Voltage*	VIH	0 +25 +70	-1170 -1130 -1070		-840 -810 -735	mV mV mV
Input Low Voltage*	VIL	0 +25 +70	-1950 -1950 -1950		-1480 -1480 -1450	mV mV mV
Output High Voltage*	VOH	0 +25 +70	-1020 -980 -920		-840 -810 -735	mV mV mV
Output Low Voltage*	VOL	0 +25 +70	-1950 -1950 -1950		-1630 -1630 -1600	mV mV mV
Input High Current (Vin = VIHmax)	IIH	0 +25 +70			10 10 10	µA µA µA
ECL VEE Supply Current	IEE	0 +25 +70			75 75 75	mA mA mA

Test conditions (unless otherwise specified): "Recommended Operating Conditions" with ECL (D0–D7) loading of 50 Ω to -2.0 V. Typical values are based on nominal temperature, i.e., room, and nominal voltage, i.e., 5 V.

*Relative to ECL VCC.

The specified limits shown can be met only after thermal equilibrium has been established. Thermal equilibrium is established by applying power for at least 2 minutes while maintaining a transverse air flow of 400 linear feet per minute over the device either mounted in the test socket or on the printed circuit board.

Bt501—TTL DC Characteristics

Parameter	Symbol	Min	Typ	Max	Units
Input High Voltage*	VIH	2.0		TTL VCC +0.5	Volts
Input Low Voltage*	VIL	TTL GND –0.5		0.8	Volts
Input High Current (Vin = 2.4 V)	IIH			70	µA
Input Low Current (Vin = 0.4 V)	IIL			–0.7	mA
Output High Voltage* (IOH = –2.0 mA)	VOH	2.5			Volts
Output Low Voltage* (IOL = 20 mA)	VOL			0.5	Volts
Three-State Output Current	IOZ				
Vout = VOHmin				10	µA
Vout = VOLmax				–10	µA
TTL VCC Supply Current	ICC			85	mA

Test conditions (unless otherwise specified): "Recommended Operating Conditions" with ECL (D0–D loading of 50 Ω to –2.0 V. Typical values are based on nominal temperature, i.e., room, and nominal voltage, i.e., 5 V.

*Relative to TTL GND.

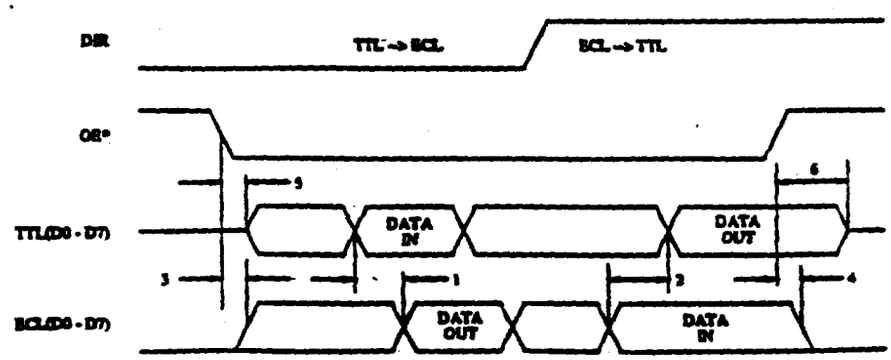
(5)

Bt501—AC Characteristics

Parameter	Symbol	Min	Typ	Max	Units
TTL --> ECL Propagation Delay	1	2		7	ns
ECL --> TTL Propagation Delay	2	5		11	ns
ECL (D0–D7) Enable Time	3	7		13	ns
ECL (D0–D7) Disable Time*	4	7		13	ns
TTL (D0–D7) Enable Time	5	4		10	ns
TTL (D0–D7) Disable Time*	6	6		12	ns

Test conditions (unless otherwise specified): "Recommended Operating Conditions" with ECL (D0–D7) loading of 50 Ω to –2.0 V. TTL input values are 0–3 V, with input rise/fall times ≤ 4 ns, measured between the 10% and 90% points. ECL input values are –0.80 to –2.0 V, with input rise/fall times ≤ 2 ns, measured between the 20% and 80% points. Timing reference points at 50% for inputs and outputs. Typical values are based on *nominal temperature*, i.e., *room*, and *nominal voltage*, i.e., 5 V.

*Subject to capacitive loading.

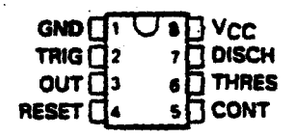


Input/Output Timing.

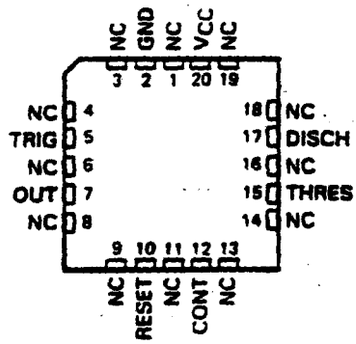
- Timing from Microseconds to Hours
- Astable or Monostable Operation
- Adjustable Duty Cycle
- TTL-Compatible Output Can Sink or Source Up to 200 mA
- Functionally Interchangeable with the Signetics SE555, SE555C, SA555, NE555; Have Same Pinout

SE555C FROM TI IS NOT RECOMMENDED FOR NEW DESIGNS

SE555, SE555C . . . JG PACKAGE
SA555, NE555 . . . D, JG, OR P PACKAGE
(TOP VIEW)



SE555, SE555C . . . FK PACKAGE
(TOP VIEW)



NC—No internal connection

description

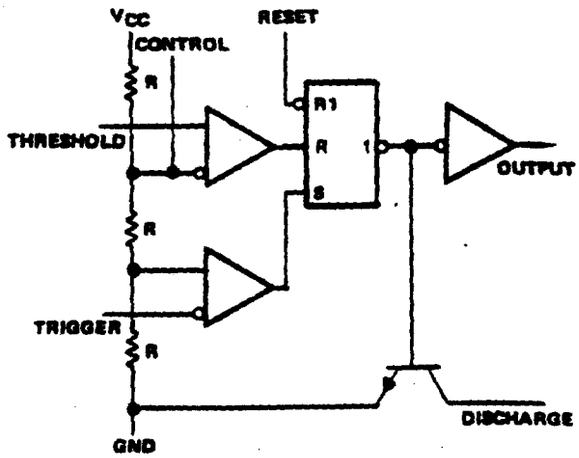
These devices are monolithic timing circuits capable of producing accurate time delays or oscillation. In the time-delay or monostable mode of operation, the timed interval is controlled by a single external resistor and capacitor network. In the astable mode of operation, the frequency and duty cycle may be independently controlled with two external resistors and a single external capacitor.

The threshold and trigger levels are normally two-thirds and one-third, respectively, of V_{CC}. These levels can be altered by use of the control voltage terminal. When the trigger input falls below the trigger level, the flip-flop is set and the output goes high. If the trigger input is above the trigger level and the threshold input is above the threshold level, the flip-flop is reset and the output is low. The reset input can override all other inputs and can be used to initiate a new timing cycle. When the reset input goes low, the flip-flop is reset and the output goes low. Whenever the output is low, a low-impedance path is provided between the discharge terminal and ground.

The output circuit is capable of sinking or sourcing current up to 200 mA. Operation is specified for supplies of 5 to 15 V. With a 5-V supply, output levels are compatible with TTL inputs.

The SE555 and SE555C are characterized for operation over the full military range of −55°C to 125°C. The SA555 is characterized for operation from −40°C to 85°C, and the NE555 is characterized for operation from 0°C to 70°C.

functional block diagram



Reset can override Trigger, which can override Threshold.

Special Functions

AVAILABLE OPTIONS

T_A RANGE	V_{thres} MAX V_{CC} = 15 V	PACKAGE			
		SMALL OUTLINE (D)	CHIP CARRIER (FK)	CERAMIC DIP (JG)	PLASTIC DIP (P)
0°C to 70°C	11.2 V	NE555D		NE555JG	NE555P
−40°C to 85°C	11.2 V	SA555D		SA555JG	SA555P
−55°C to 125°C	10.8 V 11.2 V		SE555FK SE555CFK	SE555JG SE555CJG	

The D package is available taped and reeled. Add the suffix R to the device type (e.g., NE555DR).

FUNCTION TABLE

RESET	TRIGGER VOLTAGE†	THRESHOLD VOLTAGE†	OUTPUT	DISCHARGE SWITCH
Low	Irrelevant	Irrelevant	Low	On
High	< 1/3 V_{DD}	Irrelevant	High	Off
High	> 1/3 V_{DD}	> 2/3 V_{DD}	Low	On
High	> 1/3 V_{DD}	< 2/3 V_{DD}	As previously established	

†Voltage levels shown are nominal.

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} (See Note 1) . . . . . 18 V
Input voltage (control, reset, threshold, and trigger) . . . . . V_{CC}
Output current . . . . . ±225 mA
Continuous total dissipation . . . . . see Dissipation Rating Table
Operating free-air temperature range: SE555, SE555C . . . . . −55°C to 125°C
SA555 . . . . . −40°C to 85°C
NE555 . . . . . 0°C to 70°C
Storage temperature range . . . . . −65°C to 150°C
Case temperature for 60 seconds: FK package . . . . . 260°C
Lead temperature 1,6 mm (1/16 inch) from case for 60 seconds: JG package . . . . . 300°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds: D or P package . . . . . 260°C

NOTE 1: All voltage values are with respect to network ground terminal.

DISSIPATION RATING TABLE

PACKAGE	$T_A \le 25°C$ POWER RATING	DERATING FACTOR ABOVE T_A = 25°C	T_A = 70°C POWER RATING	T_A = 85°C POWER RATING	T_A = 125°C POWER RATING
D	725 mW	5.8 mW/°C	464 mW	377 mW	N/A
FK	1375 mW	11.0 mW/°C	880 mW	715 mW	275 mW
JG (SE555, SE555C)	1050 mW	8.4 mW/°C	672 mW	546 mW	210 mW
JG (SA555, NE555)	825 mW	6.6 mW/°C	528 mW	429 mW	N/A
P	1000 mW	8.0 mW/°C	640 mW	520 mW	N/A

recommended operating conditions

	SE555		SE555C		SA555		NE555		UNIT
	MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX	
Supply voltage, V_{CC}	4.5	18	4.5	18	4.5	18	4.5	18	V
Input voltage (control, reset, threshold, and trigger)		V_{CC}		V_{CC}		V_{CC}		V_{CC}	V
Output current		±200		±200		±200		±200	m
Operating free-air temperature, T_A	−55	125	−55	125	−40	85	0	70	°

4 Spec... Functions

TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

electrical characteristics at 25°C free-air temperature, V_{CC} = 5 V to 15 V (unless otherwise n

PARAMETER	TEST CONDITIONS		SE555			SE555C, SA555, NE555			U
			MIN	TYP	MAX	MIN	TYP	MAX	
Threshold voltage level	V_{CC} = 15 V		9.4	10	10.6	8.8	10	11.2	
	V_{CC} = 5 V		2.7	3.3	4	2.4	3.3	4.2	
Threshold current (see Note 2)				30	250		30	250	n
Trigger voltage level	V_{CC} = 15 V		4.8	5	5.2	4.5	5	5.6	V
	V_{CC} = 5 V		1.45	1.67	1.9	1.1	1.67	2.2	
Trigger current	Trigger at 0 V			0.5	0.9		0.5	2	µA
Reset voltage level			0.3	0.7	1	0.3	0.7	1	V
Reset current	Reset at V_{CC}			0.1	0.4		0.1	0.4	mA
	Reset at 0 V			−0.4	−1		−0.4	−1.5	
Discharge switch off-state current				20	100		20	100	nA
Control voltage (open circuit)	V_{CC} = 15 V		9.6	10	10.4	9	10	11	V
	V_{CC} = 5 V		2.9	3.3	3.8	2.6	3.3	4	
Low-level output voltage	V_{CC} = 15 V	I_{OL} = 10 mA		0.1	0.15		0.1	0.25	
		I_{OL} = 50 mA		0.4	0.5		0.4	0.75	
		I_{OL} = 100 mA		2	2.2		2	2.5	
		I_{OL} = 200 mA		2.5			2.5		
	V_{CC} = 5 V	I_{OL} = 5 mA		0.1	0.2		0.1	0.35	
		I_{OL} = 8 mA		0.15	0.25		0.15	0.4	
High-level output voltage	V_{CC} = 15 V	I_{OH} = −100 mA	13	13.3		12.75	13.3		V
		I_{OH} = −200 mA		12.5			12.5		
	V_{CC} = 5 V	I_{OH} = −100 mA	3	3.3		2.75	3.3		
Supply current	Output low, No load	V_{CC} = 15 V		10	12		10	15	mA
		V_{CC} = 5 V		3	5		3	6	
	Output high, No load	V_{CC} = 15 V		9	10		9	13	
		V_{CC} = 5 V		2	4		2	5	

NOTE 2: This parameter influences the maximum value of the timing resistors R_A and R_B in the circuit of Figure 12. For example, when V_{CC} = 5 V, the maximum value is $R = R_A + R_B \approx 3.4$ MΩ, and for V_{CC} = 15 V, the maximum value is 10 MΩ.

operating characteristics, V_{CC} = 5 V and 15 V

PARAMETER		TEST CONDITIONS†	SE555			SE555C, SA555, NE555			UNIT
			MIN	TYP	MAX	MIN	TYP	MAX	
Initial error of timing interval‡	Each timer, monostable§	T_A = 25°C		0.5	1.5		1	3	%
	Each timer, astable¶			1.5			2.25		
Temperature coefficient of timing interval	Each timer, monostable§	T_A = MIN to MAX		30	100		50		ppm/°C
	Each timer, astable¶			90			150		
Supply voltage sensitivity of timing interval	Each timer, monostable§	T_A = 25°C		0.05	0.2		0.1	0.5	%/V
	Each timer, astable¶			0.15			0.3		
Output pulse rise time		C_L = 15 pF, T_A = 25°C		100	200		100	300	ns
Output pulse fall time				100	200		100	300	

†For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡Timing interval error is defined as the difference between the measured value and the average value of a random sample from each proce run.
§Values specified are for a device in a monostable circuit similar to Figure 9, with component values as follow: R_A = 2 kΩ to
C = 0.1 µF.
¶Values specified are for a device in an astable circuit similar to Figure 12, with component values as follow: R_A = 1 kΩ to 100 kΩ, C = 0.1 µF.

TYPICAL CHARACTERISTICS†

LOW-LEVEL OUTPUT VOLTAGE
vs
LOW-LEVEL OUTPUT CURRENT

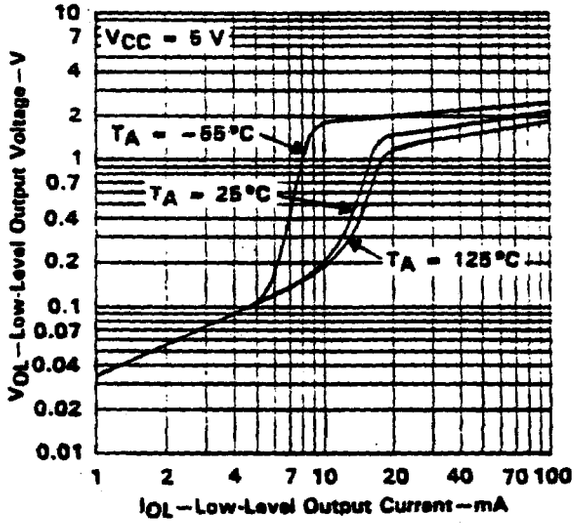


FIGURE 1

LOW-LEVEL OUTPUT VOLTAGE
vs
LOW-LEVEL OUTPUT CURRENT

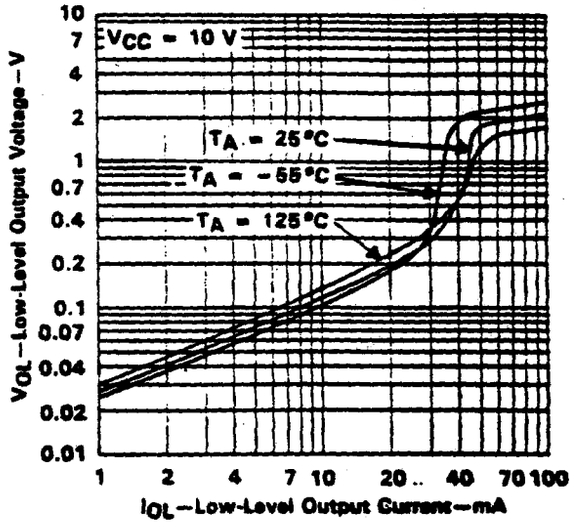


FIGURE 2

LOW-LEVEL OUTPUT VOLTAGE
vs
LOW-LEVEL OUTPUT CURRENT

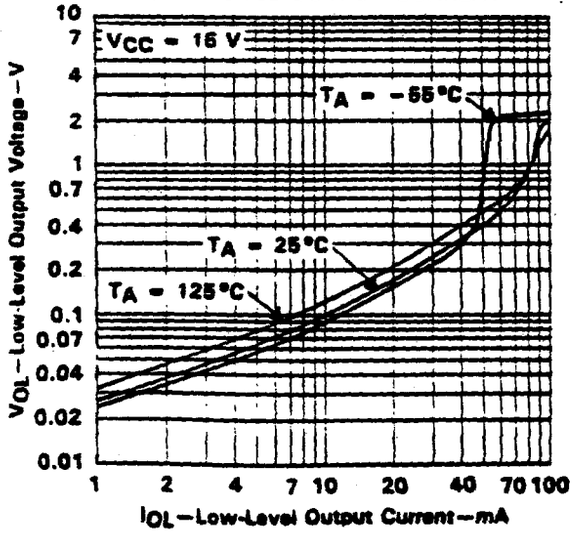


FIGURE 3

DROP BETWEEN SUPPLY VOLTAGE AND OUTPUT
vs
HIGH-LEVEL OUTPUT CURRENT

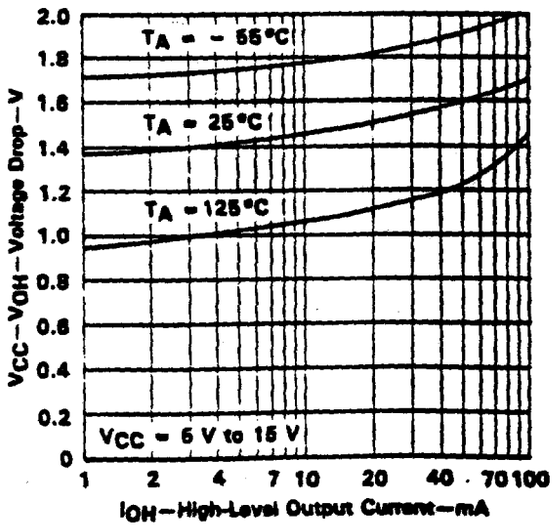


FIGURE 4

†Data for temperatures below 0°C and above 70°C are applicable for SE555 circuits only.

TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

TYPICAL CHARACTERISTICS†

SUPPLY CURRENT
vs
SUPPLY VOLTAGE

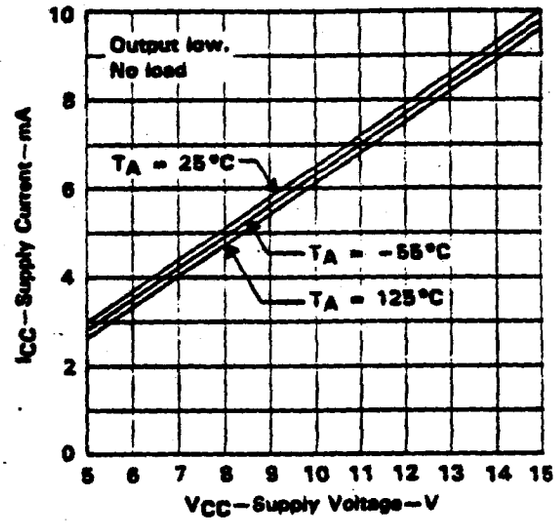


FIGURE 5

NORMALIZED OUTPUT PULSE DURATION
(MONOSTABLE OPERATION)
vs
SUPPLY VOLTAGE

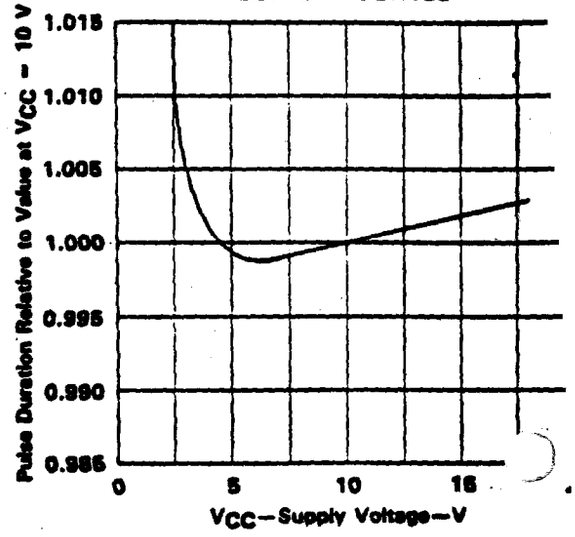


FIGURE 6

NORMALIZED OUTPUT PULSE DURATION
(MONOSTABLE OPERATION)
vs
FREE-AIR TEMPERATURE

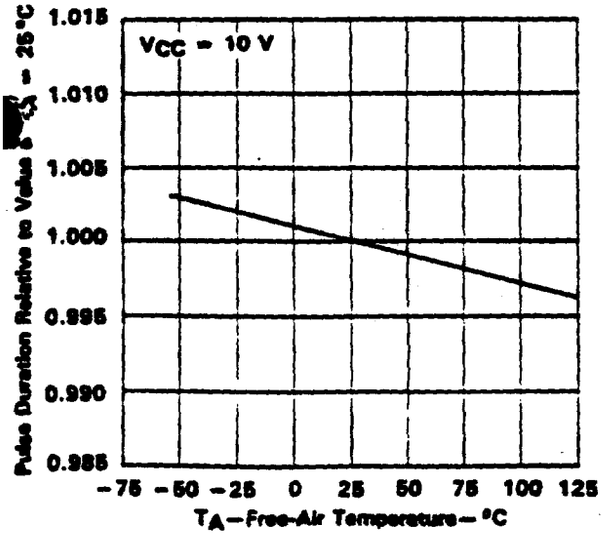


FIGURE 7

PROPAGATION DELAY TIME
vs
LOWEST VOLTAGE LEVEL
OF TRIGGER PULSE

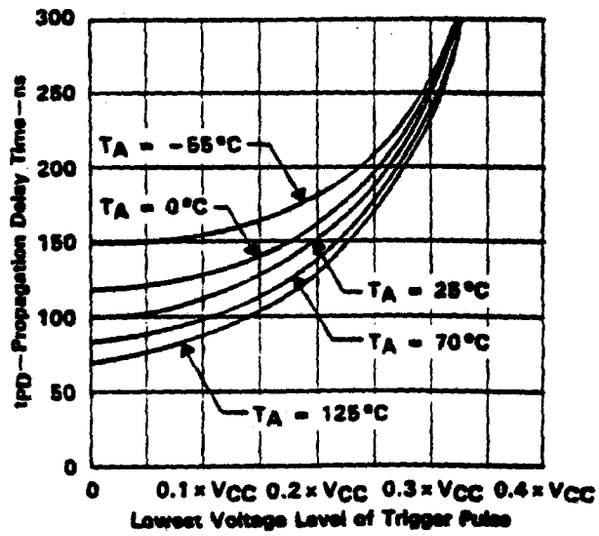


FIGURE 8

†Data for temperatures below 0°C and above 70°C are applicable for SE555 circuits only.

TYPICAL APPLICATION DATA

monostable operation

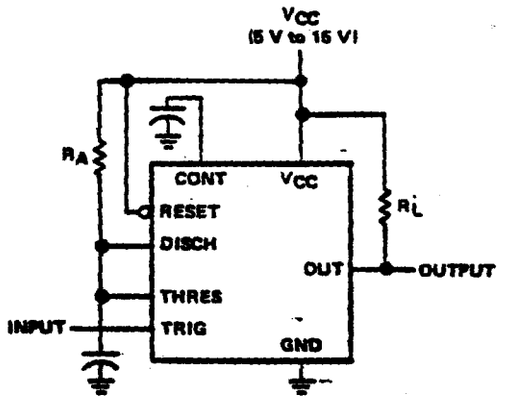


FIGURE 9. CIRCUIT FOR MONOSTABLE OPERATION

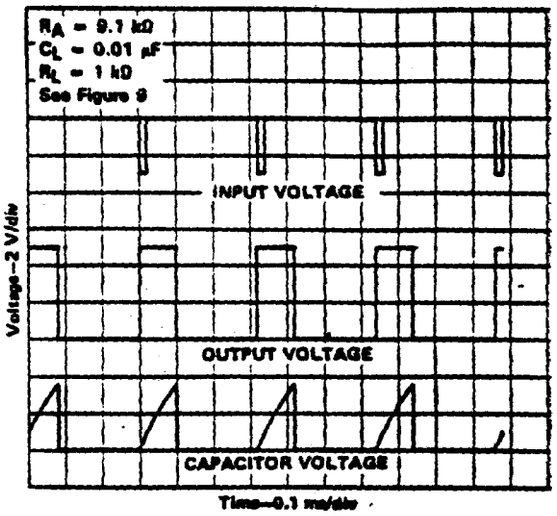


FIGURE 10. TYPICAL MONOSTABLE WAVEFORMS

For monostable operation, any of these timers may be connected as shown in Figure 9. If the output is low, application of a negative-going pulse to the trigger input sets the flip-flop ($\overline{Q}$ goes low), drives the output high, and turns off Q1. Capacitor C is then charged through R_A until the voltage across the capacitor reaches the threshold voltage of the threshold input. If the trigger input has returned to a high level, the output of the threshold comparator will reset the flip-flop ($\overline{Q}$ goes high), drive the output low, and discharge C through Q1.

Monostable operation is initiated when the trigger input voltage falls below the trigger threshold. Once initiated, the sequence ends only if the trigger input is high at the end of the timing interval. Because of the threshold level and saturation voltage of Q1, the output pulse duration is approximately $t_w = 1.1\ R_AC$. Figure 11 is a plot of the time constant for various values of R_A and C. The threshold levels and charge rates are both directly proportional to the supply voltage, V_{CC}. The timing interval is therefore independent of the supply voltage, so long as the supply voltage is constant during the time interval.

Applying a negative-going trigger pulse simultaneously to the reset and trigger terminals during the timing interval discharges C and re-initiates the cycle, commencing on the positive edge of the reset pulse. The output is held low as long as the reset pulse is low. To prevent false triggering, when the reset input is not used, it should be connected to V_{CC}.

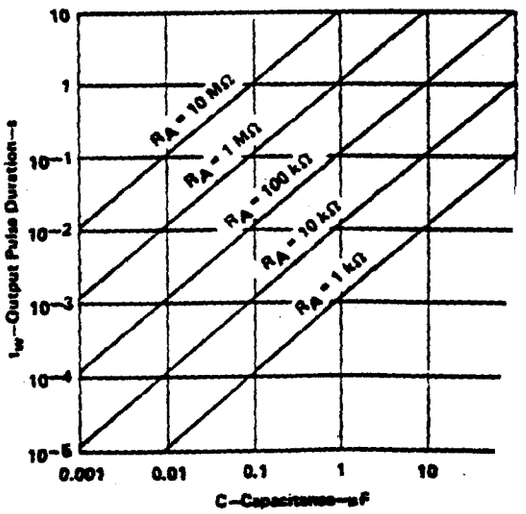
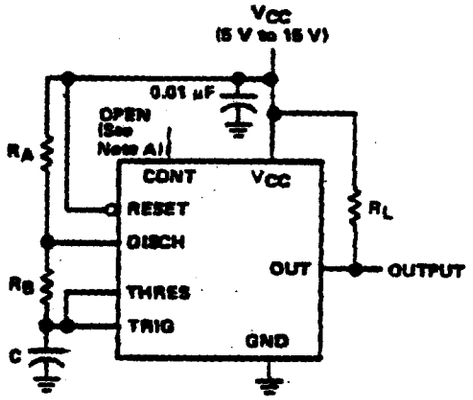


FIGURE 11. OUTPUT PULSE DURATION vs CAPACITANCE

TYPICAL APPLICATION DATA

astable operation



NOTE A: Decoupling the control voltage input to ground with a capacitor may improve operation. This should be evaluated for individual applications.

FIGURE 12. CIRCUIT FOR ASTABLE OPERATION

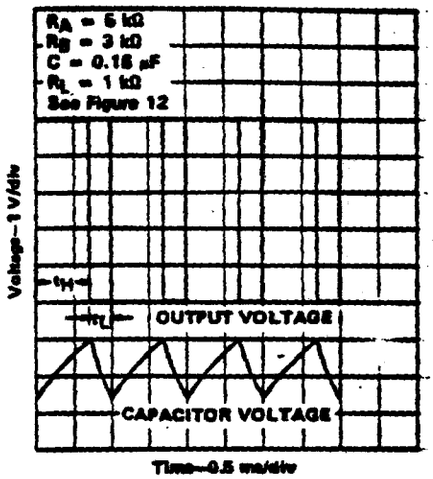


FIGURE 13. TYPICAL ASTABLE WAVEFO

As shown in Figure 12, adding a second resistor, R_B, to the circuit of Figure 9 and connecting the trigger input to the threshold input causes the timer to self-trigger and run as a multivibrator. The capacitor C will charge through R_A and R_B and then discharge through R_B only. The duty cycle may be controlled, therefore, by the values of R_A and R_B.

This astable connection results in capacitor C charging and discharging between the threshold-voltage level ($\approx 0.67 \cdot V_{CC}$) and the trigger-voltage level ($\approx 0.33 \cdot V_{CC}$). As in the monostable circuit, charge and discharge times (and therefore the frequency and duty cycle) are independent of the supply voltage.

Figure 13 shows typical waveforms generated during astable operation. The output high-level duration t_H and low-level duration t_L may be calculated as follows:

$$t_H = 0.693\ (R_A + R_B)\ C$$

$$t_L = 0.693\ (R_B)\ C$$

Other useful relationships are shown below.

$$\text{period} = t_H + t_L = 0.693\ (R_A + 2R_B)\ C$$

$$\text{frequency} = \frac{1.44}{(R_A + 2R_B)\ C}$$

$$\text{Output driver duty cycle} = \frac{t_L}{t_H + t_L} = \frac{R_B}{R_A + 2R_B}$$

$$\text{Output waveform duty cycle} = \frac{t_H}{t_H + t_L} = 1 - \frac{R_B}{R_A + 2R_B}$$

$$\text{Low-to-high ratio} = \frac{t_L}{t_H} = \frac{R_B}{R_A + R_B}$$

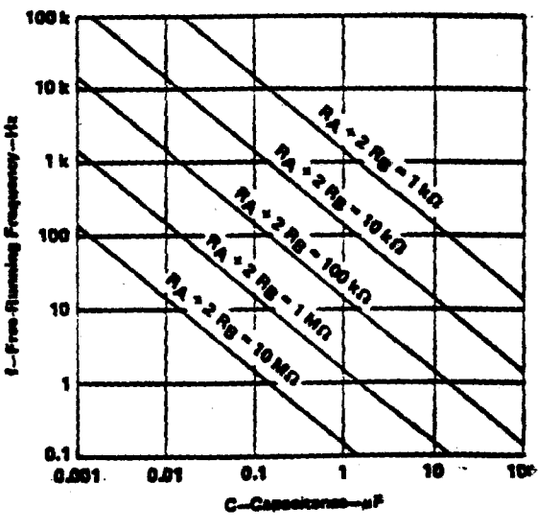


FIGURE 14. FREE-RUNNING FREQUENCY

TYPICAL APPLICATION DATA

missing-pulse detector

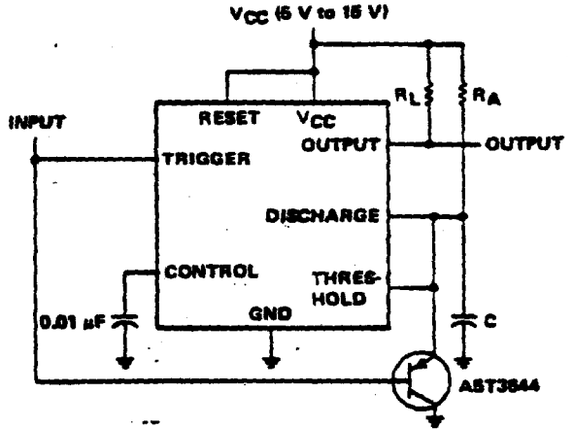


FIGURE 15. CIRCUIT FOR MISSING-PULSE DETECTOR

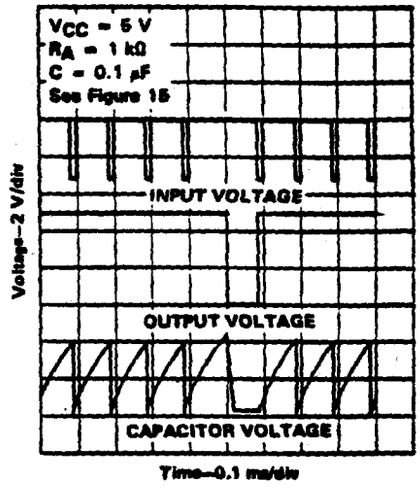


FIGURE 16. MISSING-PULSE DETECTOR WAVEFORMS

The circuit shown in Figure 15 may be used to detect a missing pulse or abnormally long spacing between consecutive pulses in a train of pulses. The timing interval of the monostable circuit is continuously retriggered by the input pulse train as long as the pulse spacing is less than the timing interval. A longer pulse spacing, missing pulse, or terminated pulse train permits the timing interval to be completed, thereby generating an output pulse as illustrated in Figure 16.

frequency divider

By adjusting the length of the timing cycle, the basic circuit of Figure 9 can be made to operate as a frequency divider. Figure 17 illustrates a divide-by-3 circuit that makes use of the fact that retriggering cannot occur during the timing cycle.

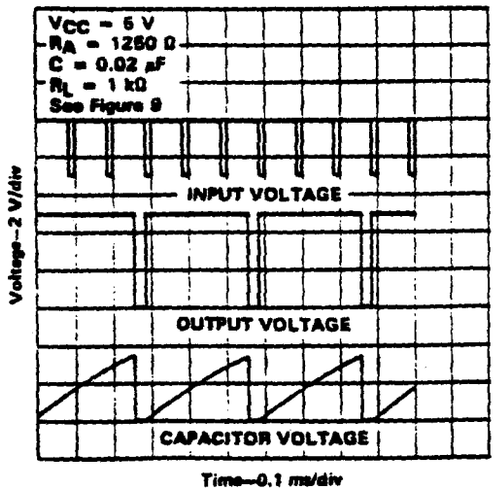
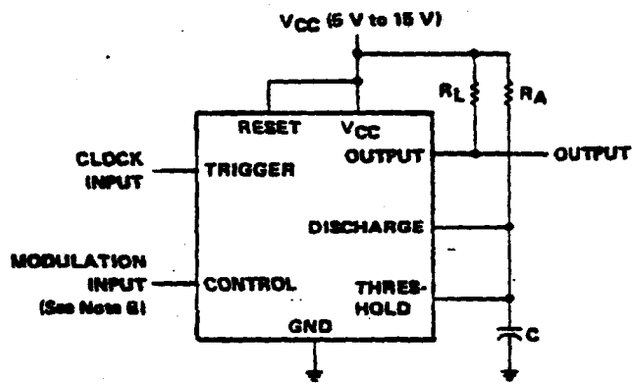


FIGURE 17. DIVIDE-BY-THREE CIRCUIT WAVEFORMS

TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

TYPICAL APPLICATION DATA

pulse-width modulation



NOTE B: The modulating signal may be direct or capacitively coupled to the control terminal. For direct coupling, the effects of modulation source voltage and impedance on the bias of the timer should be considered.

FIGURE 18. CIRCUIT FOR PULSE-WIDTH MODULATION

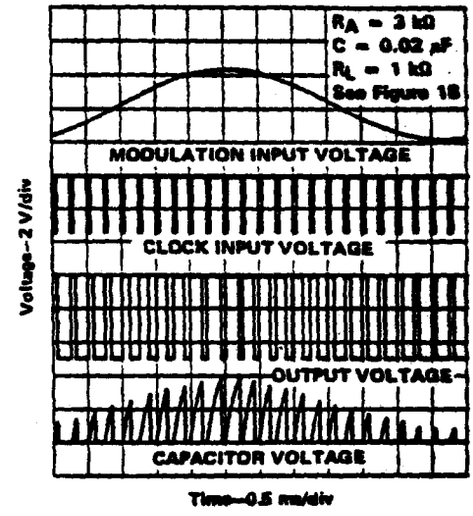
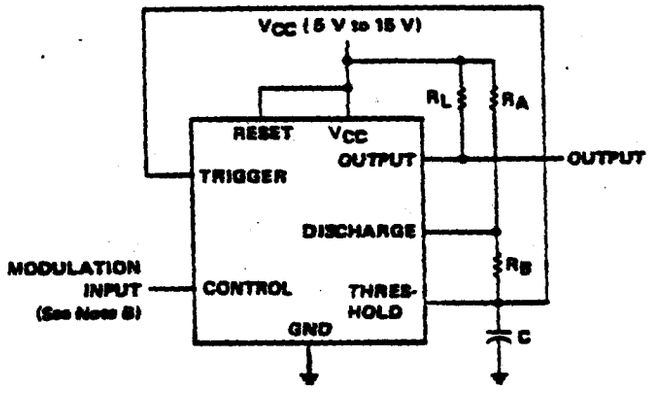


FIGURE 19. PULSE-WIDTH MODULATION WAVEFORMS

The operation of the timer may be modified by modulating the internal threshold and trigger voltages, which is accomplished by applying an external voltage (or current) to the control pin. Figure 18 shows a circuit for pulse-width modulation. A continuous input pulse train triggers the monostable circuit, and a control signal modulates the threshold voltage. Figure 19 illustrates the resulting output pulse-width modulation. While a sine-wave modulation signal is illustrated, any wave shape could be used.

TYPICAL APPLICATION DATA

pulse-position modulation



NOTE B: The modulating signal may be direct or capacitively coupled to the control terminal. For direct coupling, the effects of modulation source voltage and impedance on the bias of the timer should be considered.

FIGURE 20. CIRCUIT FOR PULSE-POSITION MODULATION

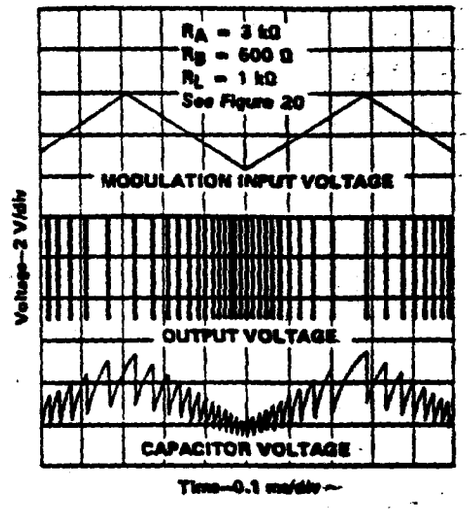


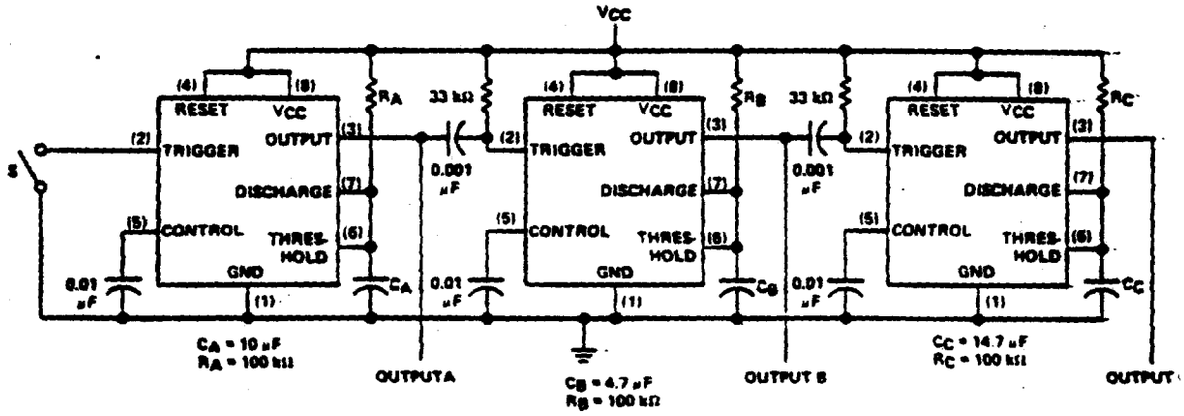
FIGURE 21. PULSE POSITION-MODULATION WAVEFORMS

As shown in Figure 20, any of these timers may be used as a pulse-position modulator. This applicatio modulates the threshold voltage, and thereby the time delay, of a free-running oscillator. Figure 21 illustrate a triangular-wave modulation signal for such a circuit; however, any wave shape could be used.

TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

TYPICAL APPLICATION DATA

sequential timer



S closes momentarily at t = 0.

FIGURE 22. SEQUENTIAL TIMER CIRCUIT

Many applications, such as computers, require signals for initializing conditions during start-up. Other applications, such as test equipment, require activation of test signals in sequence. These timing circuits may be connected to provide such sequential control. The timers may be used in various combinations of astable or monostable circuit connections, with or without modulation, for extremely flexible waveform control. Figure 22 illustrates a sequencer circuit with possible applications in many systems, and Figure 23 shows the output waveforms.

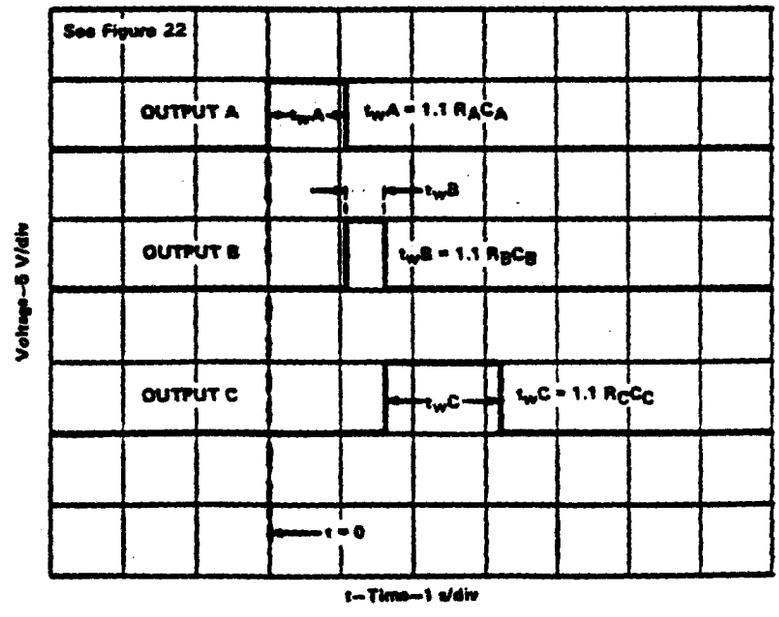


FIGURE 23. SEQUENTIAL TIMER WAVEFORMS

Product Preview

MC10E016
MC100E016

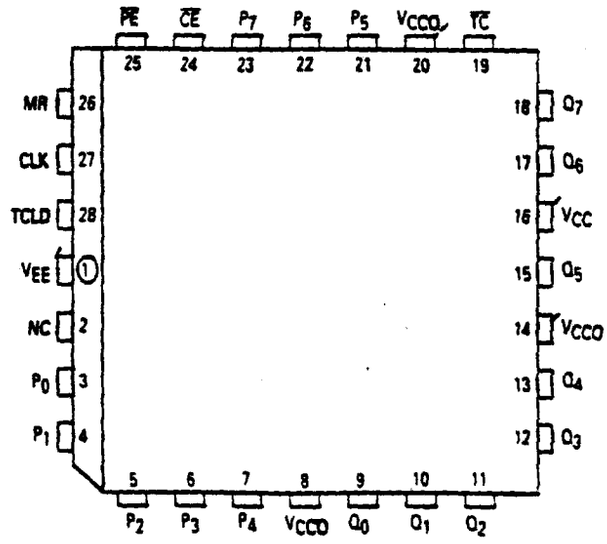
8-BIT SYNCHRONOUS BINARY UP COUNTER

- 700 MHz Min. Count Frequency
- 1000 ps CLK to Q, $\overline{TC}$
- Internal $\overline{TC}$ Feedback (Gated)
- 8-Bit
- Fully Synchronous Counting and $\overline{TC}$ Generation
- Asynchronous Master Reset

The MC10E/100E016 is a high-speed synchronous, presettable, cascadable 8-bit binary counter. Architecture and operation are the same as the MC10H016 in the MECL 10KH family, extended to 8-bits, as shown in the logic symbol.

The counter features internal feedback of $\overline{TC}$, gated by the TCLD (terminal count load) pin. When TCLD is LOW (or left open, in which case it is pulled LOW by the internal pull-downs), the $\overline{TC}$ feedback is disabled, and counting proceeds continuously, with $\overline{TC}$ going LOW to indicate an all-one state. When TCLD is HIGH, the $\overline{TC}$ feedback causes the counter to automatically re-load upon $\overline{TC}$ = LOW, thus functioning as a programmable counter.

PINOUT: 28-LEAD PLCC (TOP VIEW)



FUNCTION TABLE

$\overline{CE}$	$\overline{PE}$	TCLD	MR	CLK	Function
L	L	X	L	Z	Load Parallel (P_n to Q_n)
H	L	X	L	Z	Load Parallel (P_n to Q_n)
L	H	L	L	Z	Continuous Count
L	H	H	L	Z	Count: Load Parallel on $\overline{TC}$ = LOW
H	H	X	L	Z	Hold
X	X	X	L	ZZ	Masters Respond, Slaves Hold
X	X	X	H	Z	Reset (Q_n := LOW, $\overline{TC}$:= HIGH)

Z = clock pulse (low to high);
ZZ = clock pulse (high to low)

PIN NAMES

Pin	Function
P_0–P_7	Parallel Data (Preset) Inputs
Q_0–Q_7	Data Outputs
$\overline{CE}$	Count Enable Control Input
$\overline{PE}$	Parallel Load Enable Control Input
MR	Master Reset
CLK	Clock
$\overline{TC}$	Terminal Count Output
TCLD	TC-Load Control Input

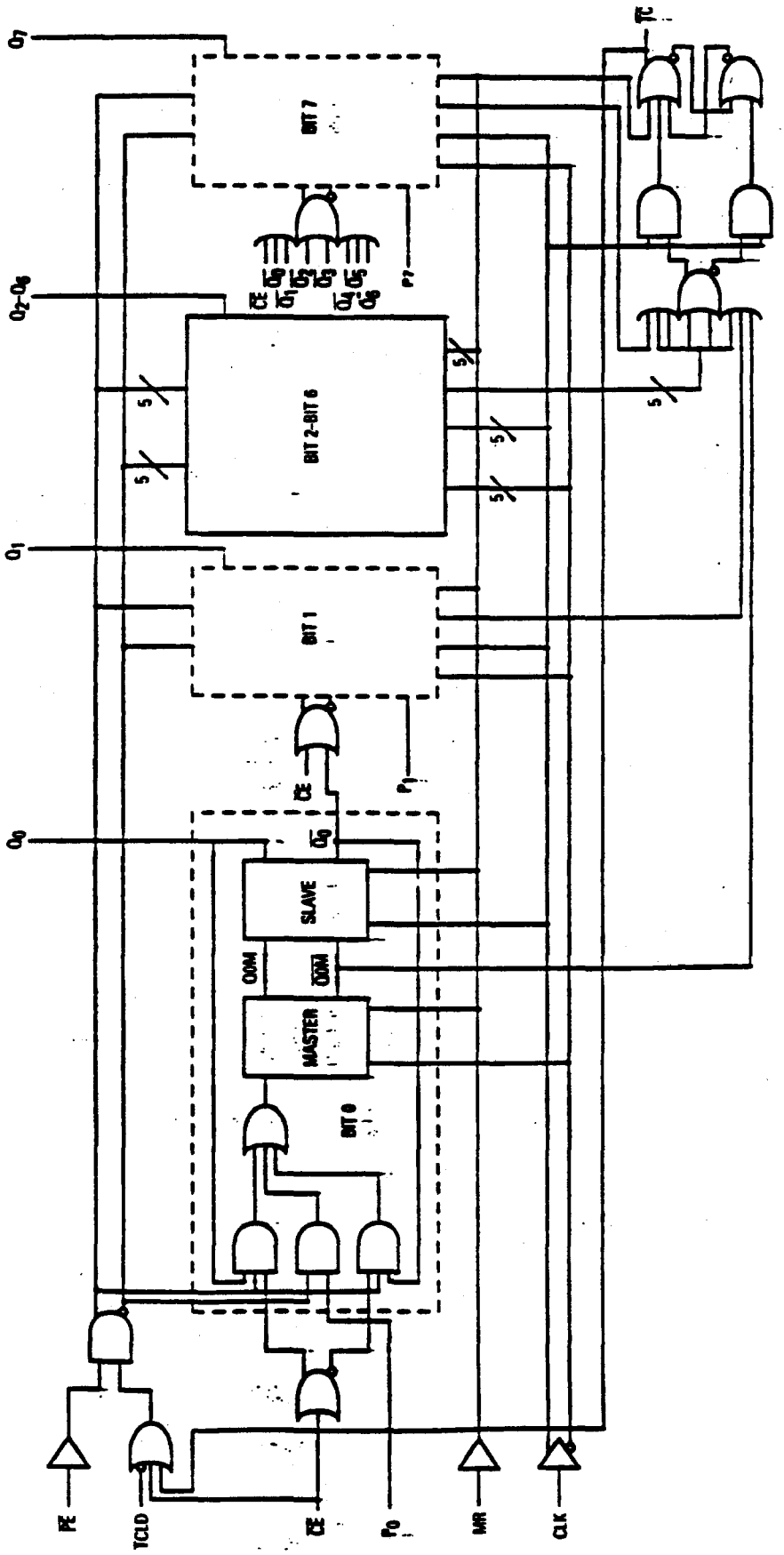
DC CHARACTERISTICS: V_{EE} = V_{EE} (min) to V_{EE} (max); V_{CC} = V_{CCO} = GND

Symbol	Characteristic	T_A = 0°C			T_A = 25°C			T_A = 85°C			Unit	Conditions
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max		
I_{IH}	Input HIGH Current			150			150			150	μA	
I_{EE}	Power Supply Current										mA	
	10E		151	181		151	181		151	181		
	100E		151	181		151	181		174	208		

AC CHARACTERISTICS: V_{EE} = V_{EE} (min) to V_{EE} (max); V_{CC} = V_{CCO} = GND

Symbol	Characteristic	T_A = 0°C			T_A = 25°C			T_A = 85°C			Unit	Conditions
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max		
f_{COUNT}	Max. Count Frequency	700	900		700	900		700	900		MHz	
t_{PLH} t_{PHL}	Propagation Delay to Output										ps	
	CLK to Q	450		1000	450		1000	450		1000		
	MR to Q	450		1000	450		1000	450		1000		
	CLK to $\overline{TC}$	400		900	400		900	400		900		
	MR to $\overline{TC}$	400		900	400		900	400		900		

8-BIT BINARY COUNTER LOGIC DIAGRAM



Note that this diagram is provided for understanding of logic operation only. It should not be used for propagation delays as many gate functions are achieved internally without incurring a full gate del

3 WKS

MOTOROLA
SEMICONDUCTOR
TECHNICAL DATA

25
13 10

SCG
11 20

XC 10E101

Product Preview

MC10E101
MC100E101

- 500 ps Max. Propagation Delay

The MC10E/100E101 is a quad 4-input OR/NOR gate.

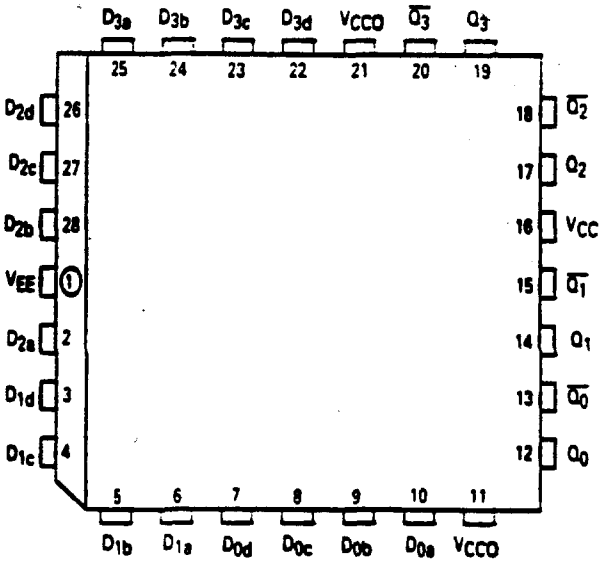
QUAD 4-INPUT OR/NOR GATE

490-9500

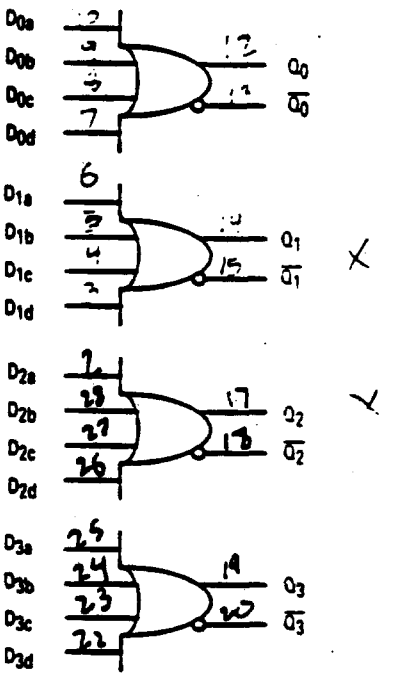
PAM KELEBA

KULESZA

PINOUT: 28-LEAD PLCC (TOP VIEW)



LOGIC SYMBOL



X

X

PIN NAMES

Pin	Function
D_{0a}–D_{3d}	Data Inputs
Q_0–Q_3	True Outputs
$\overline{Q_0}$–$\overline{Q_3}$	Inverting Outputs

500
20.10

SOIC PKG

OLD 5 WKS

23 50

8

DC CHARACTERISTICS: V_{EE} = V_{EE} (min) to V_{EE} (max); V_{CC} = V_{CCO} = GND

Symbol	Characteristic	T_A = 0°C			T_A = 25°C			T_A = 85°C			Unit	Conditions
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max		
I_{IH}	Input HIGH Current D			150			150			150	µA	
I_{EE}	Power Supply Current 10E 100E		30 30	36 36		30 30	36 36		30 35	36 42	mA	

AC CHARACTERISTICS: V_{EE} = V_{EE} (min) to V_{EE} (max); V_{CC} = V_{CCO} = GND

Symbol	Characteristic	T_A = 0°C			T_A = 25°C			T_A = 85°C			Unit	Conditions
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max		
t_{PLH} t_{PHL}	Propagation Delay to Output D	150		500	150		500	150		500	ps	

MOTOROLA

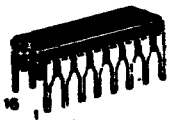
MC10H115

QUAD LINE RECEIVER

The MC10H115 is a quad differential amplifier designed for use in sensing differential signals over long lines. This 10KH part is a functional/pinout duplication of the standard MECL 10K family part, with 100% improvement in counting frequency and no increase in power-supply current.

The base bias supply (V_{BB}) is made available at Pin 9 to make the device useful as a Schmitt trigger, or in other applications where a stable reference voltage is necessary. Active current sources provide the MC10H115 with excellent common mode rejection. If any amplifier in a package is not used, one input of that amplifier must be connected to V_{BB} (Pin 9) to prevent upsetting the current source bias network.

- Propagation Delay, 1.0 ns Typical
- Power Dissipation 110 mW Typ/Pkg (No Load)
- Improved Noise Margin 150 mV (Over Operating Voltage and Temperature Range)
- Voltage Compensated
- MECL 10K-Compatible

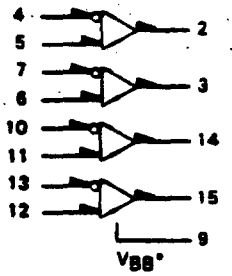


L SUFFIX
CERAMIC PACKAGE
CASE 620

P SUFFIX
PLASTIC PACKAGE
CASE 648

FN SUFFIX
PLCC
CASE 775

LOGIC DIAGRAM



When input pin with bubble goes positive, the output goes negative.

*V_{BB} to be used to supply bias to the MC10H115 only and bypassed (when used) with 0.01 μF to 0.1 μF capacitor.

V_{CC1} = Pin 1
V_{CC2} = Pin 16
V_{EE} = Pin 8

MAXIMUM RATINGS

Characteristic	Symbol	Rating	Unit
Power Supply (V_{CC} = 0)	V_{EE}	−8.0 to 0	Vdc
Input Voltage (V_{CC} = 0)	V_I	0 to V_{EE}	Vdc
Output Current — Continuous — Surge	I_{out}	50 100	mA
Operating Temperature Range	T_A	0–75	°C
Storage Temperature Range — Plastic — Ceramic	T_{stg}	−55 to 150 −55 to 165	°C °C

ELECTRICAL CHARACTERISTICS (V_{EE} = −5.2 V ±5%) (See Note)

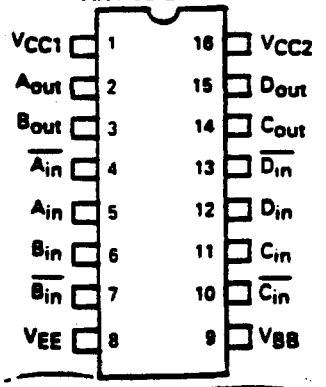
Characteristic	Symbol	0° Min	0° Max	25° Min	25° Max	75° Min	75° Max	Unit
Power Supply Current	I_E	—	29	—	26	—	29	mA
Input Current	I_{inH}	—	150	—	95	—	95	μA
	I_{CBO}	—	1.5	—	1.0	—	1.0	μA
High Output Voltage	V_{OH}	−1.02	−0.84	−0.98	−0.81	−0.92	−0.735	Vdc
Low Output Voltage	V_{OL}	−1.95	−1.63	−1.95	−1.63	−1.95	−1.60	Vdc
High Input Voltage	V_{IH}	−1.17	−0.84	−1.13	−0.81	−1.07	−0.735	Vdc
Low Input Voltage	V_{IL}	−1.95	−1.48	−1.95	−1.48	−1.95	−1.45	Vdc
Reference Voltage	V_{BB}	−1.38	−1.27	−1.35	−1.25	−1.31	−1.19	Vdc

AC PARAMETERS

Propagation Delay	t_{pd}	0.4	1.3	0.4	1.3	0.45	1.45	ns
Rise Time	t_r	0.5	1.4	0.5	1.5	0.5	1.6	ns
Fall Time	t_f	0.5	1.4	0.5	1.5	0.5	1.6	ns

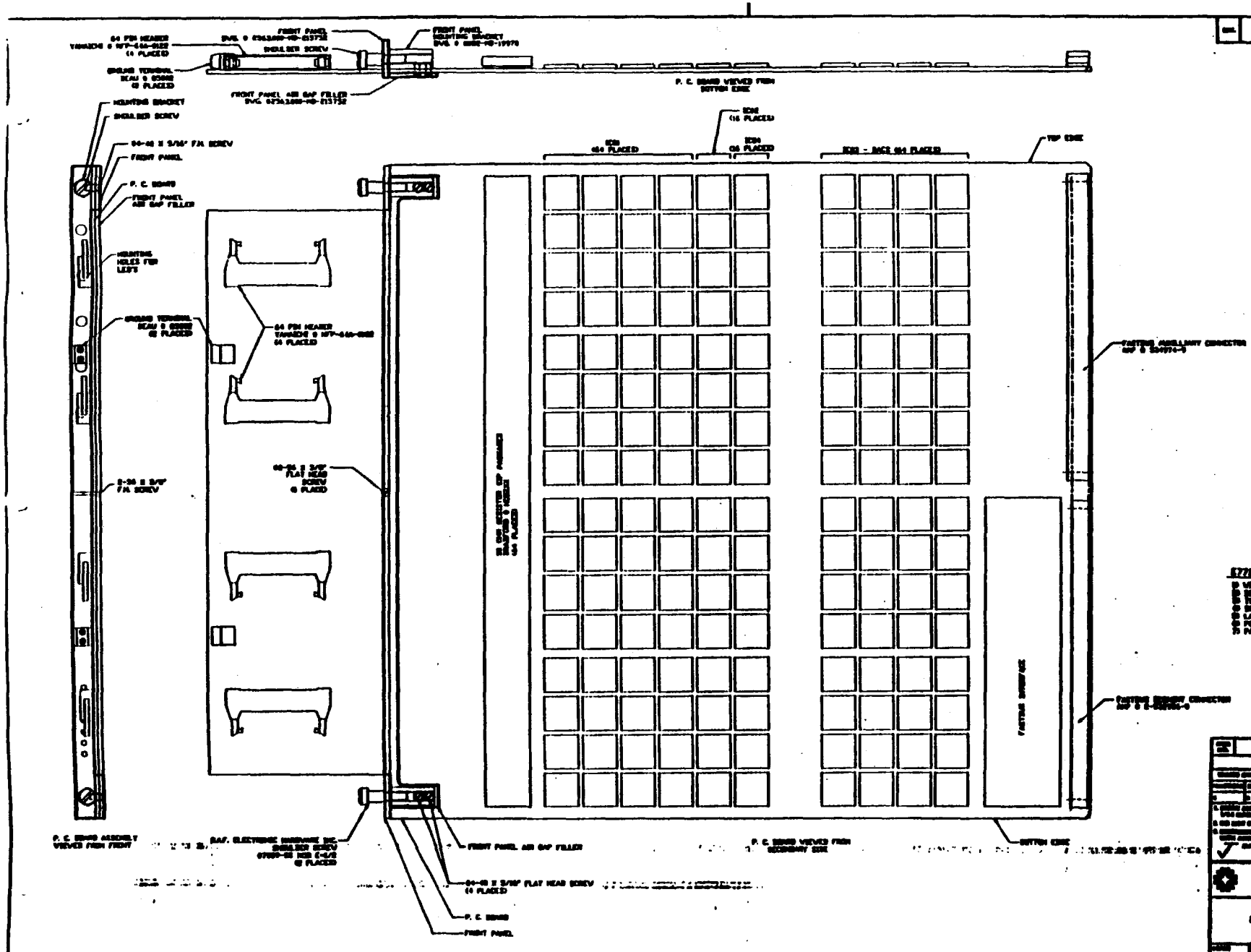
NOTE:
Each MECL 10KH series circuit has been designed to meet the dc specifications shown in the test table, after thermal equilibrium has been established. The circuit is in a test socket or mounted on a printed circuit board and transverse air flow greater than 500 lfpm is maintained. Outputs are terminated through a 50 ohm resistor to −2.0 volts.

DIP PIN ASSIGNMENT



Pin assignment is for Dual-in-line Package.
For PLCC pin assignment, see tables on page 1-31.

2



E771 - POST AMP COMPARATOR DRAWING LIST

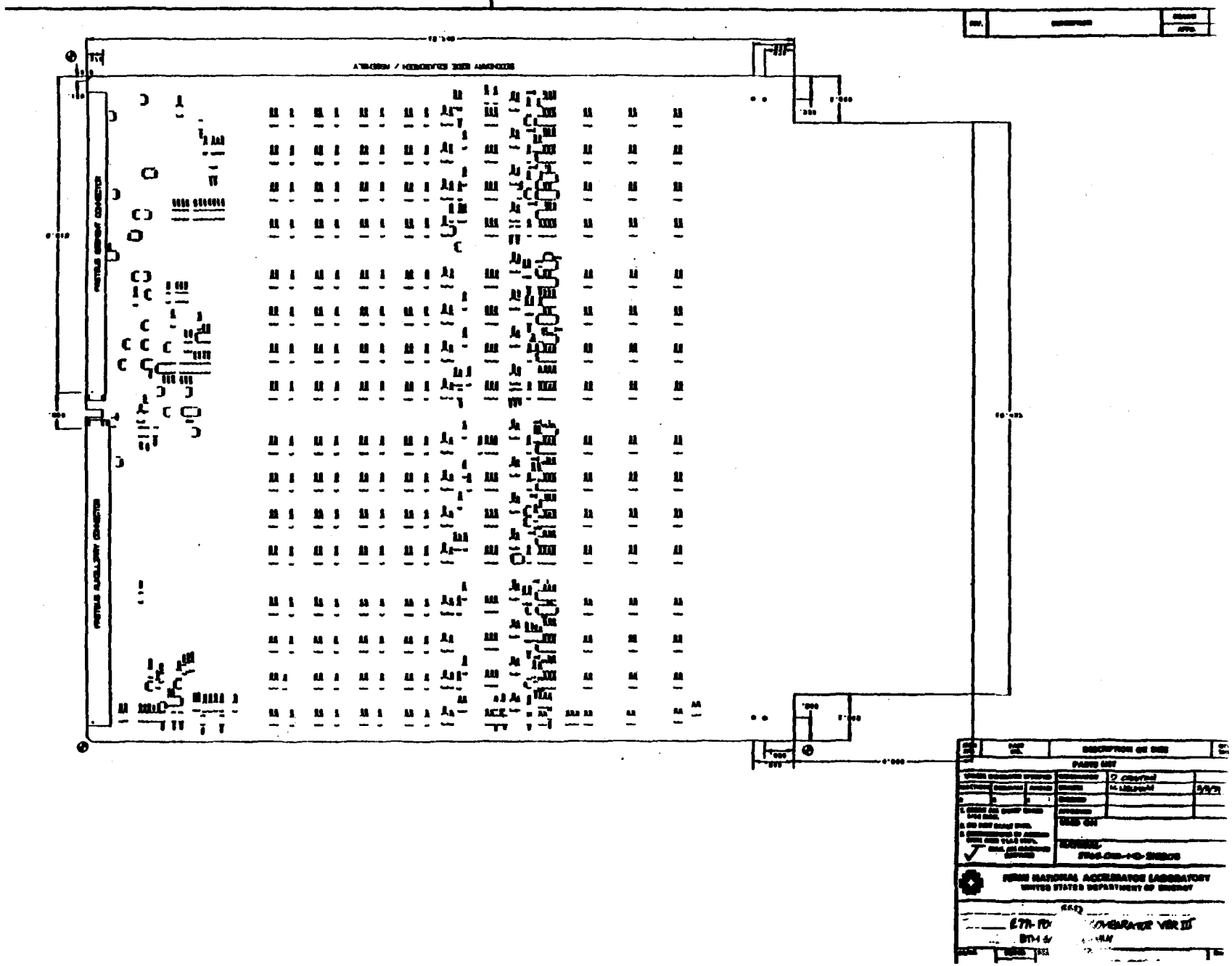
FERMI NATIONAL ACCELERATOR LABORATORY
UNITED STATES DEPARTMENT OF ENERGY

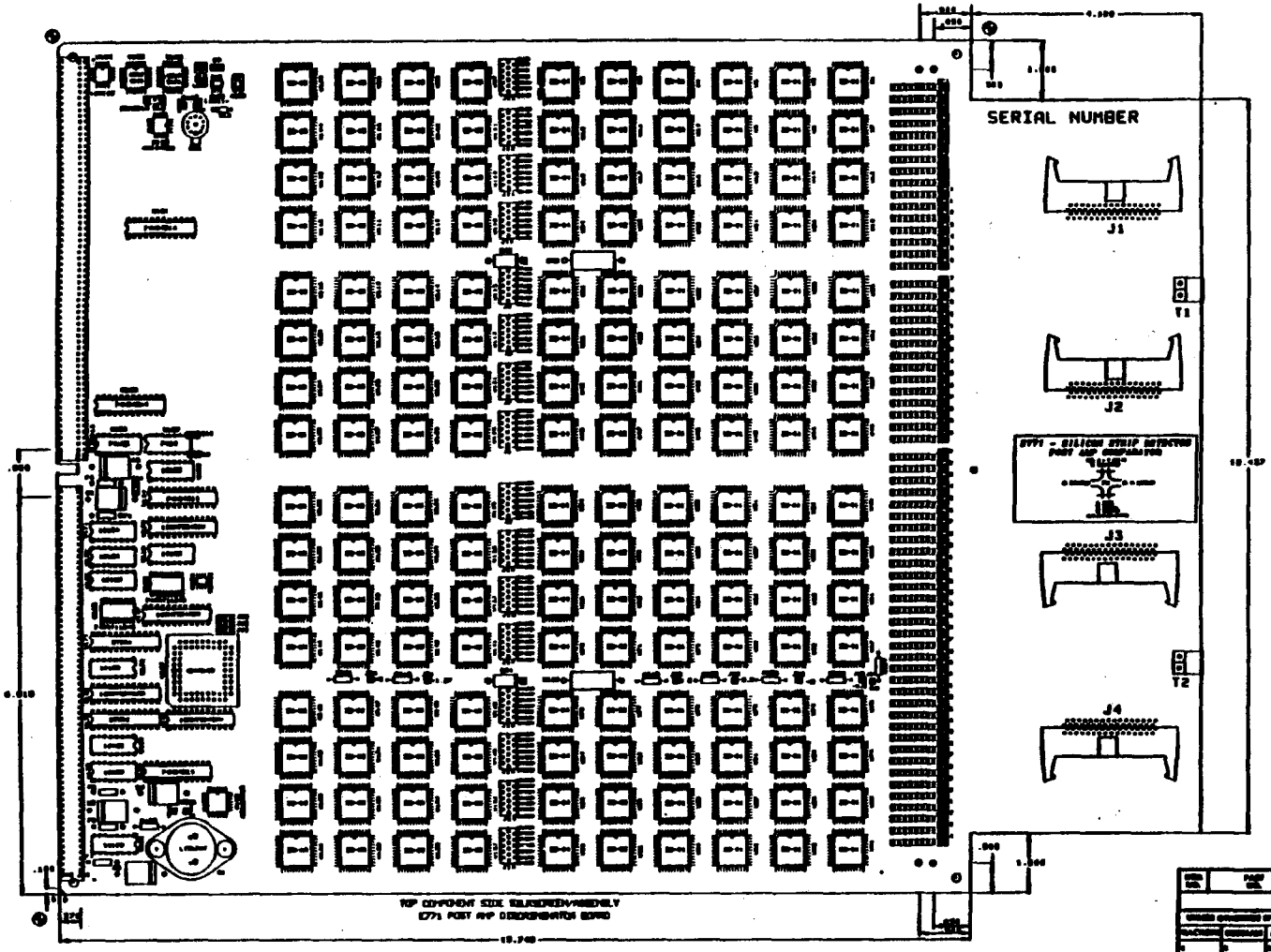
ECD

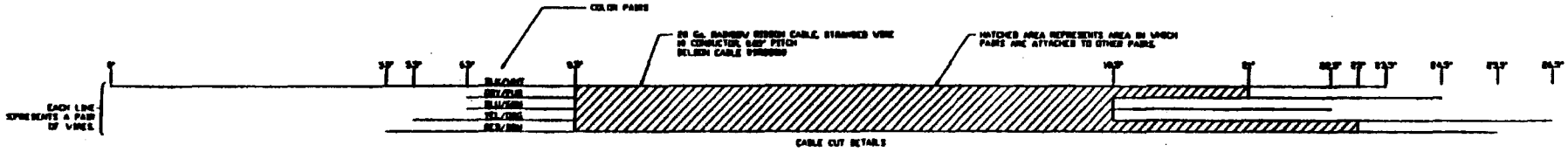
E771 - POST AMP COMPARATOR VER III

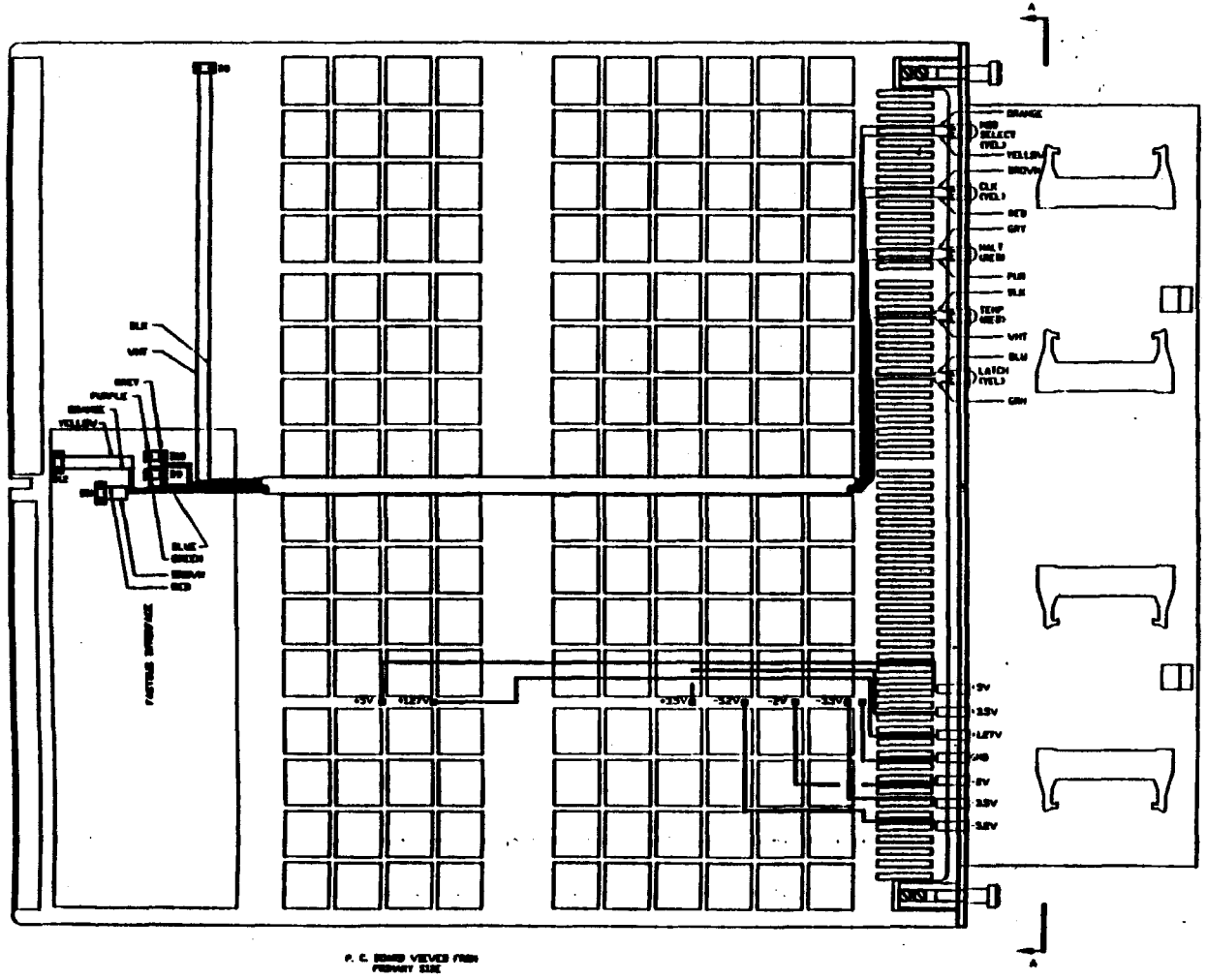
P. C. BOARD ASSEMBLY

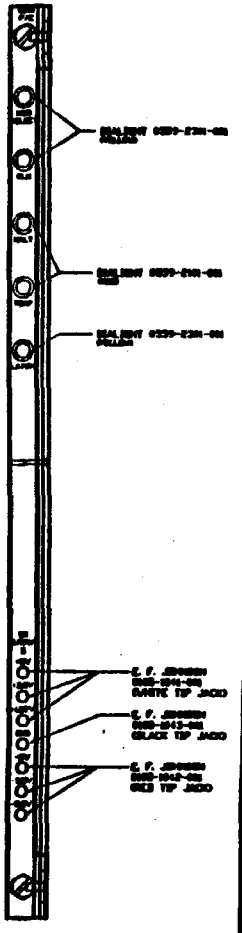
2563A99-MD-015003


FERMI NATIONAL ACCELERATOR LABORATORY
UNITED STATES DEPARTMENT OF ENERGY


SERIAL NUMBER
J1
T1
J2
J3
T2
J4







NOTE:

1- BNC JUMPER WIRING - 2363400-40-21000

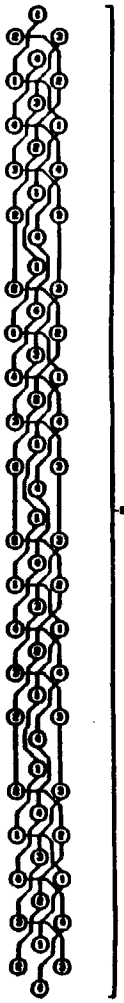
PARTS LIST

FERMI NATIONAL ACCELERATOR LABORATORY
UNITED STATES DEPARTMENT OF ENERGY

ECD

E771 - [illegible] COMPARATOR VERSION II

LED [illegible] POINT WIRING DETAILS

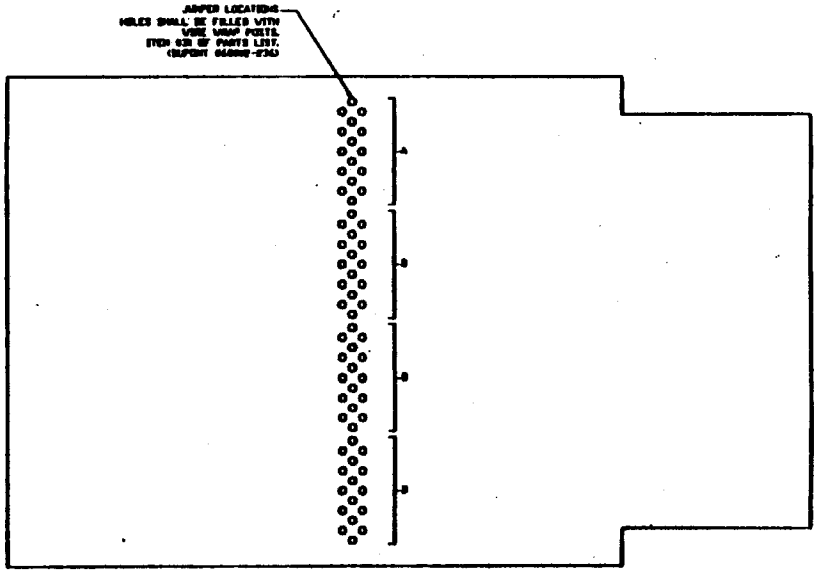


SECTION "A"

SAC JUMPER DETAILS

SAC DIAGRAM TO SAC LOFT OR TO SC SAC USED WHEN A SC-100, 40-PIN PLCC ECP ARE EMPLOYED. THE 4 PLCC'S SHOULD OCCUPY LOCATIONS U17, U22, U27 AND U32. USE 30 AWG WIRE WRAP WIRE WITH CLOSED INSULATION. REPEAT THE DIAGRAM FOR SECTIONS B, C AND D (SEE DRAWING OUTLINE AT RIGHT). COLOR CODE WIRES AS FOLLOWS:

PIN 01 VERIFY - BLUE WIRE
PIN 02 VERIFY - RED WIRE
PIN 03 VERIFY - WHITE WIRE
PIN 04 VERIFY - YELLOW WIRE



P. C. BOARD OUTLINE

REV.	DESCRIPTION	DRAWN	DATE
		APPR.	

REVISED SYMBOLS STANDARD			ORIGINATED	E. VALENTINE	08/23/
DIMENSIONS	TOLERANCES	LETTERS	DRAWN	E. A. BOWMAN	08/23/
X	±		CHECKED		
X.X	±		APPROVED		
X.XX	±		USED ON		
ANGLES ±					
DO NOT SCALE DRAWING					
MATERIAL AND FINISH AS SPECIFIED					
INTERPRET DRAWING PER ANSI Y14.5M					

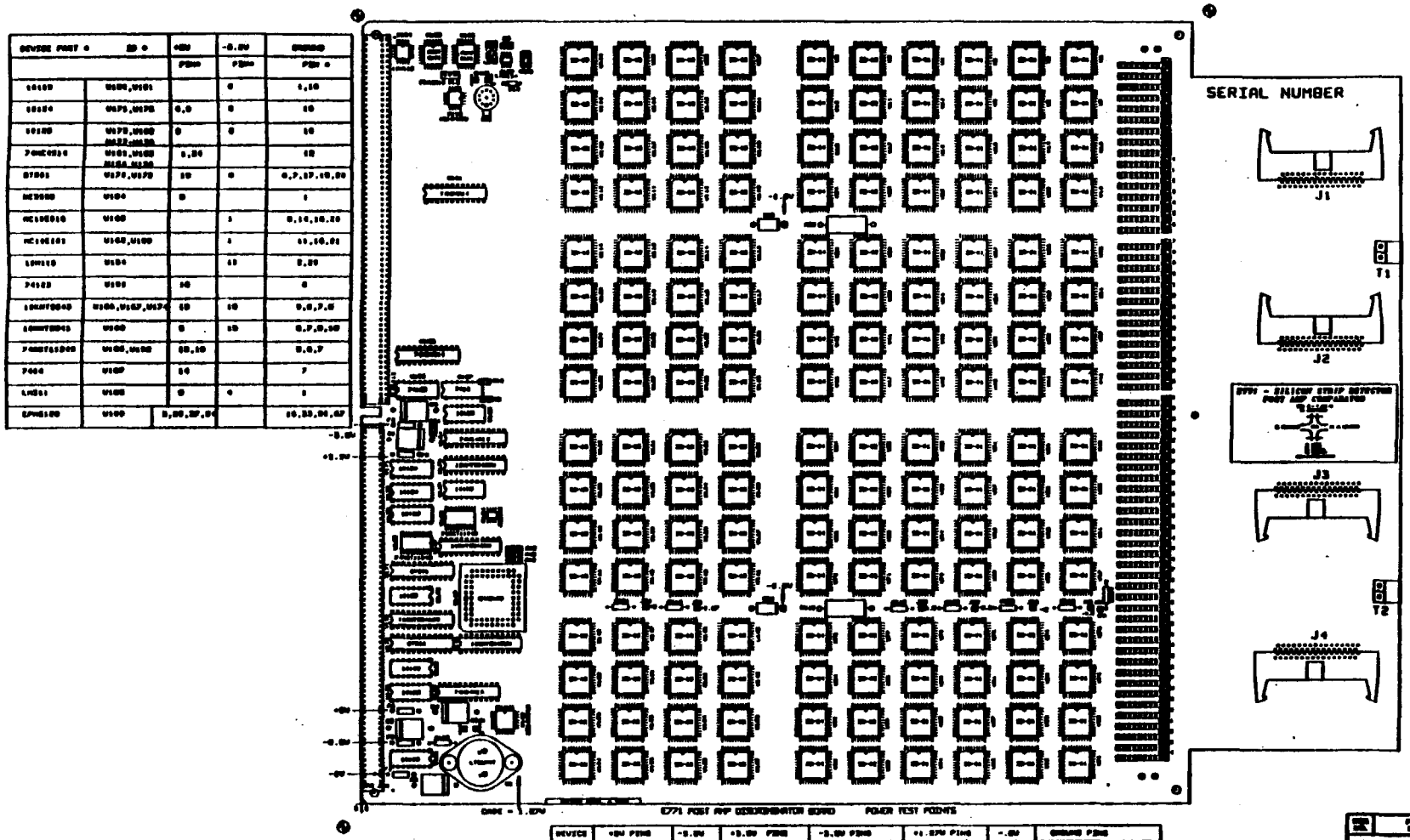
FERMI NATIONAL ACCELERATOR LABORATORY
UNITED STATES DEPARTMENT OF ENERGY

ECD

E771 - POST AMP COMPARATOR VER III

SAC JUMPER DIAGRAM

SCALE NONE	SHEET	DRAWING NUMBER



DEVICE POST #	ID #	+5V PINS	-5.2V PINS	GROUND PINS #
10109	U101,U101		9	1,10
10154	U175,U175	6,9	9	10
10180	U170,U180 U182,U185	9	9	10
74HC0214	U101,U100 U104,U105	1,20		10
87301	U170,U170	10	9	6,7,17,19,20
MC9500	U104	9		1
MC100E016	U100		1	9,14,16,20
MC10E101	U100,U100		1	11,16,01
LM110	U104		11	2,20
74123	U101	16		8
10KHT0040	U100,U107,U174	10	10	9,8,7,6
10KHT0045	U100	9	10	8,7,6,10
74HCT11000	U100,U100	20,10		9,8,7
7404	U107	14		7
LM311	U100	9	4	1
EPH5100	U100	2,20,17,04		10,11,04,07

E771 POST AMP DISCRIMINATOR BOARD POWER TEST POINTS

DEVICE	+5V PINS	-5.2V	+3.3V PINS	-3.3V PINS	+1.67V PINS	-.8V	GROUND PINS
IC-01			3,10,20,30,40	8,17,20,22,33,30			2,7,10,12,21,23,24,28,40
IC-02		40,41,0	20,40				8,24,30
IC-03	8,10,20,30,44				10,17,21,30,41	0,03,40	11,10,01,00,20,27,40
IC-04		20	01				00,00,20

REV	DESCRIPTION AS BUILT
	PARTS LIST

FERMI NATIONAL ACCELERATOR LABORATORY
UNITED STATES DEPARTMENT OF ENERGY

E771- POST AMP DISCRIMINATOR WITH 32-CHANNEL LATCHED PRINT DETAILS

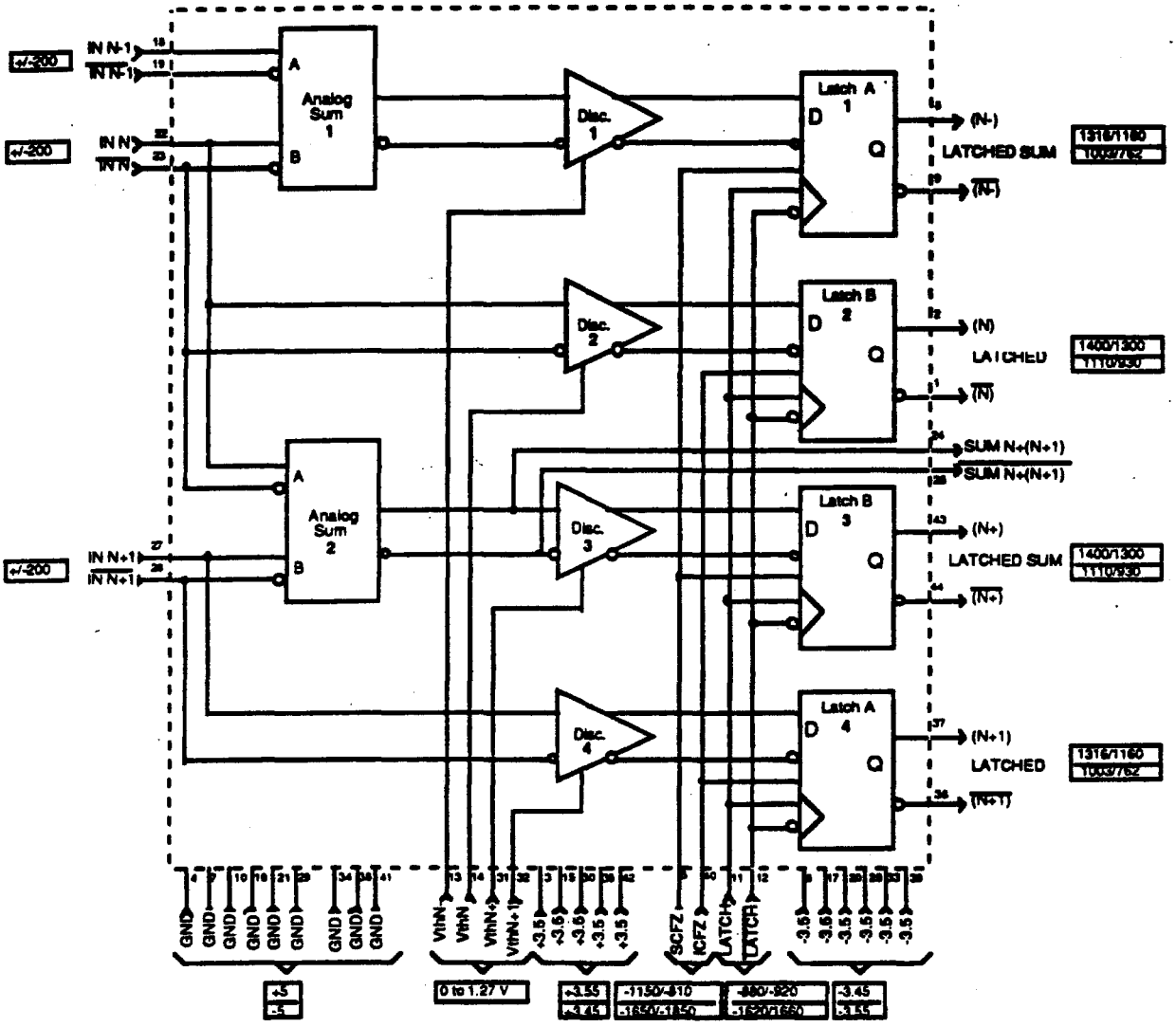
HD-215609

APPENDIX E - DATA SHEETS

This appendix contains data sheets for the ASICs IC-01, IC-02 and IC-04 as well as data sheets for all parts on the P/C moduele.

IC-01: SSD TWO CHANNEL SUM, DISCRIMINATOR AND LATCH

Cavity Down



Part Number IC-01

NOTES:

1. All input and output signals are full differential except power, threshold (Vthi#)
2. The discriminator compares the input amplitude with the threshold setting. If the input exceeds the setting, an output exists for the time over
3. The "LATCH" signal must be at least 1 Nanosecond wide.
4. Power dissipation is typically 406 mW ; 504 mW max.
5. Power supply currents: +3.5 V @typically 48 mA ; 56 mA
   -3.5 V @typically 66 mA ; 86 mA
6. Voltages in boxes are given in millivolts. Max / Min

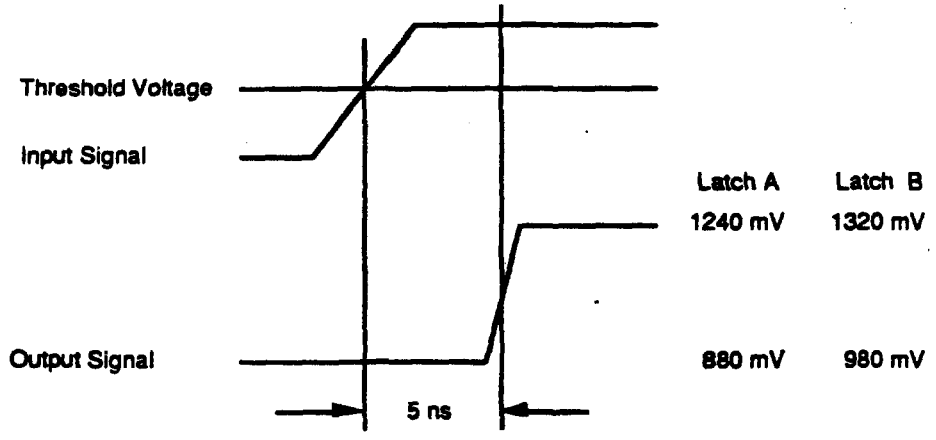
Created 1-24-89
Revised 2-14-91
Marie Haldeman / Scott Heim

IC-01 Typical Characteristics

Package 44 Pin PLCC

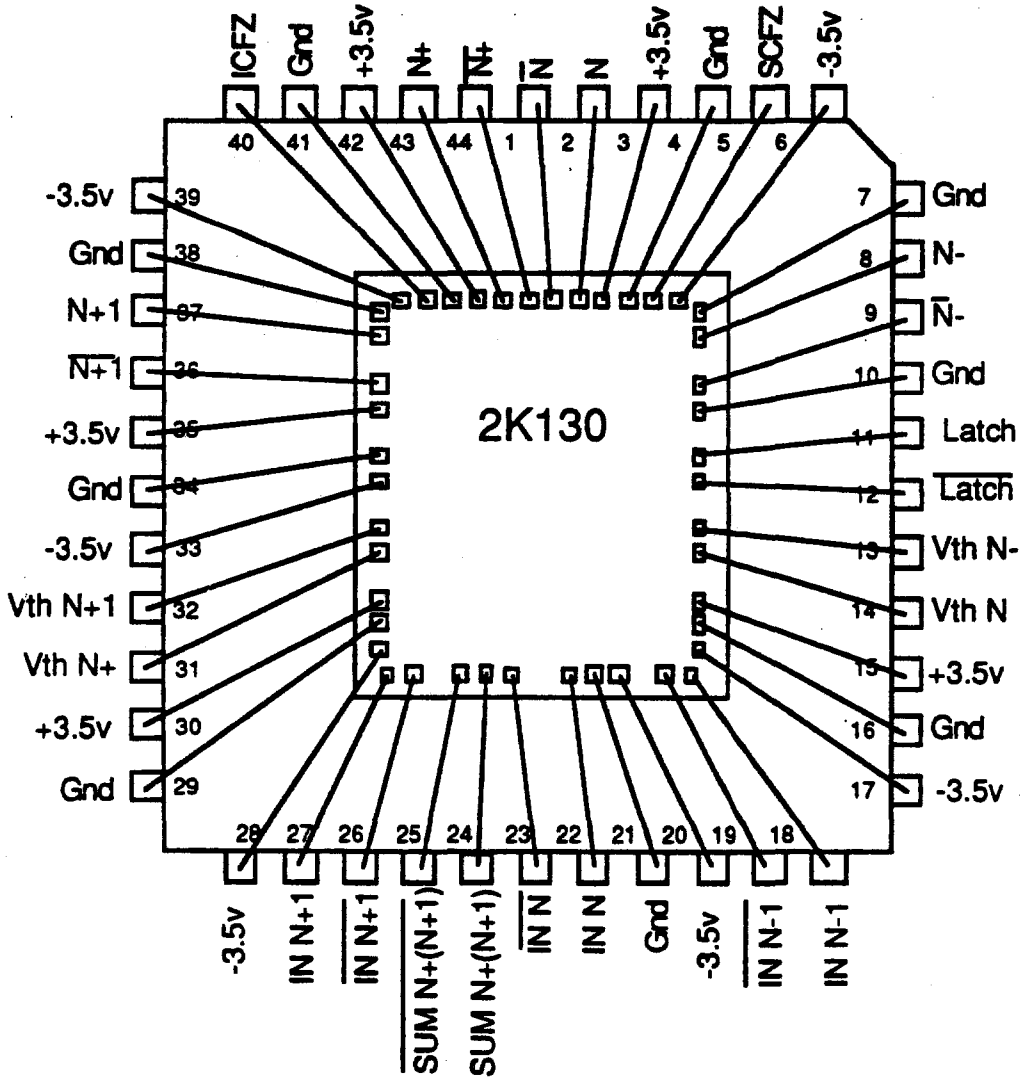
	Tektronix Package Code= 44PLCC4
Power Supply	**+3.5 Volts @ 48 mA** **-3.5 Volts @ 68 mA**
Power Dissipation **Discriminators**	**406 mW** **≈180 mW**
Thermal Resistance	**43°C/W$_{JA}$ 6°C/W$_{JC}$**
Differential Input Signal Range	**90 mV**
Common Mode Signal Range	**+/- 1 Volt**
Input Resistance	**4 kΩ**
Input Bias Current	**3 µA**
Differential Output Voltage	**710 mV**
Output Current	**Source 5 mA** **Sink 1.25 mA**
Propagation Delay	**5 ns**
Adjacent Channel Crosstalk	**40 db**
Hysteresis	**10 mV**
Triggering Uncertainty	**0.5 mV**
Ambient Temperature Range	**0 to 70°C**

IC-01 TIMING



IC-01: SSD TWO CHANNEL SUM, DISCRIMINATOR and LATCH
Cavity Down
Thermally Enhanced, 44 Lead Plastic Chip Carrier

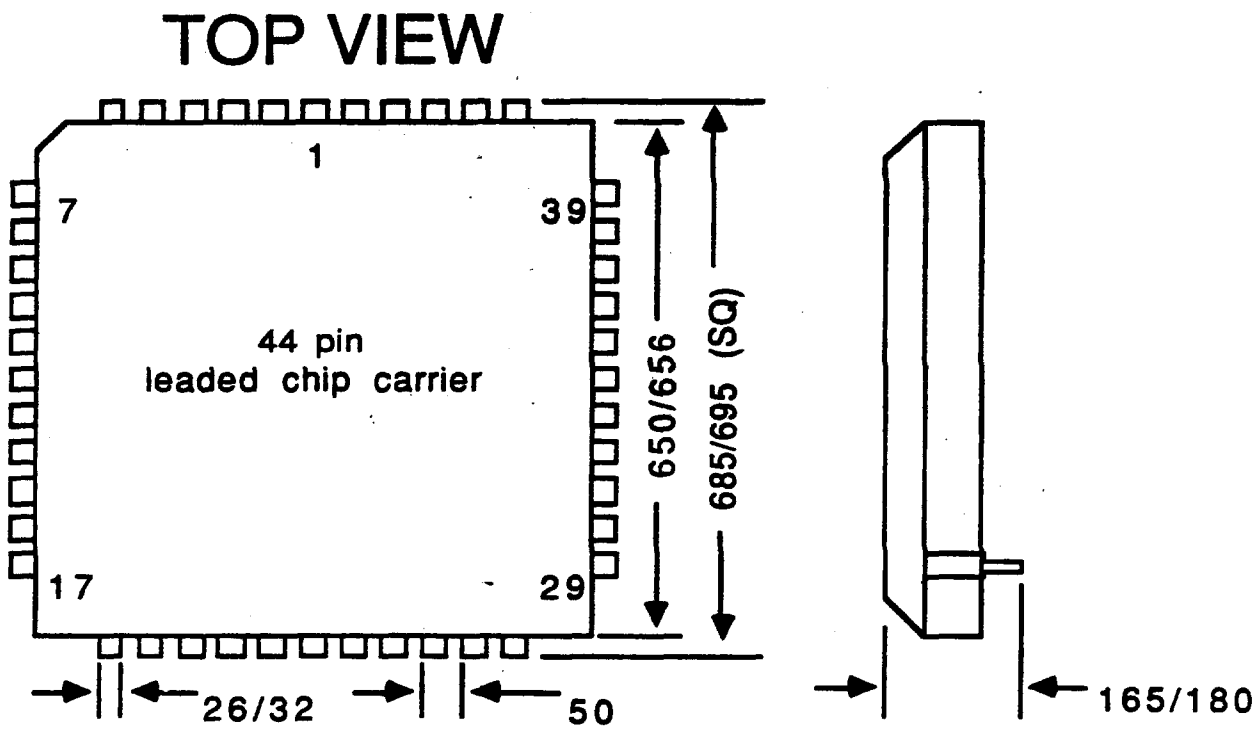
Part Number IC-01



BOTTOM VIEW

Created 8-28-89
Revised
Merle Haldeman / Scott Holm

IC-01: SSD TWO CHANNEL SUM, DISCRIMINATOR and LATCH
Cavity Down



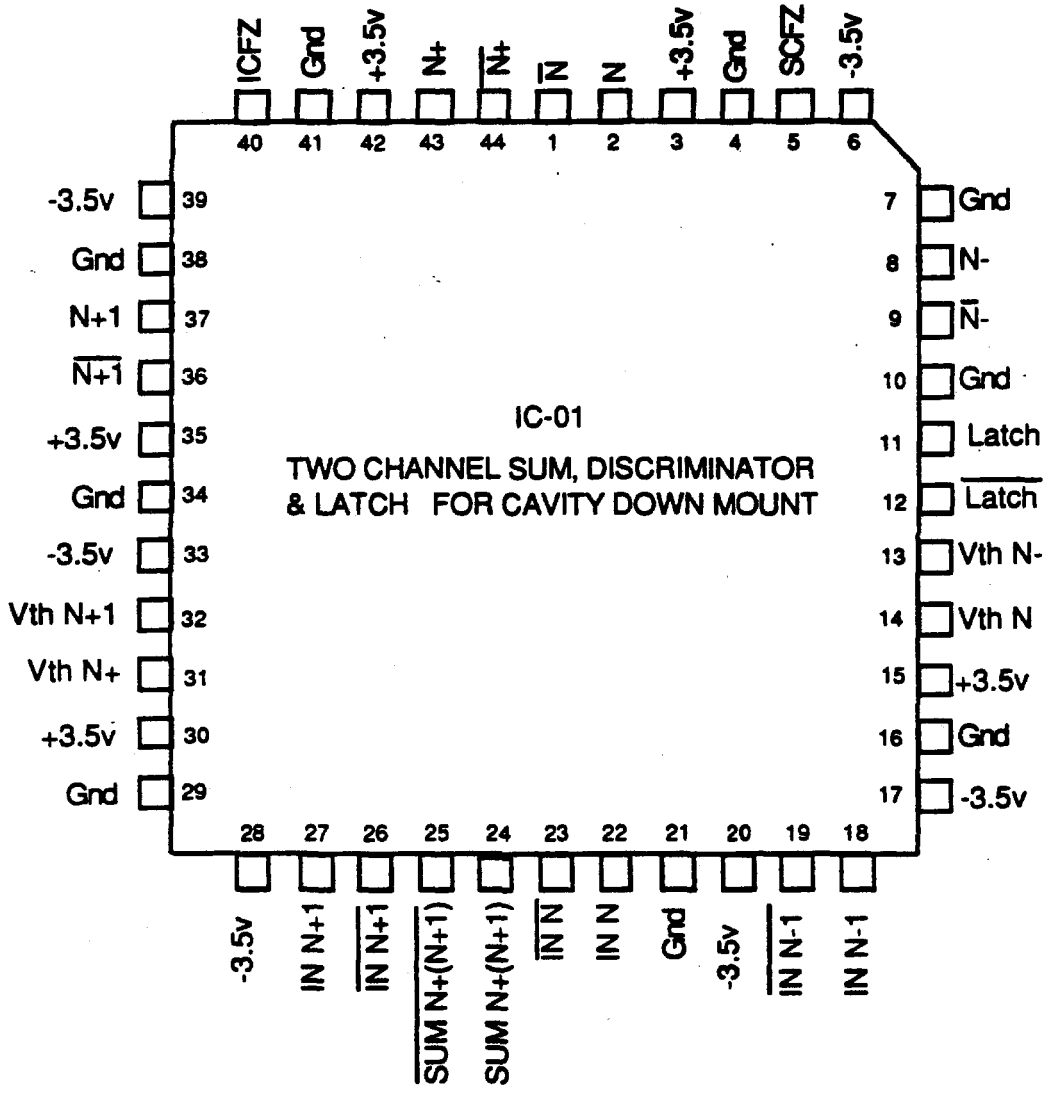
Part Number IC-01

IC-01 Package Dimension Diagram Notes.

1. All dimensions are in mils. (Min/Max)

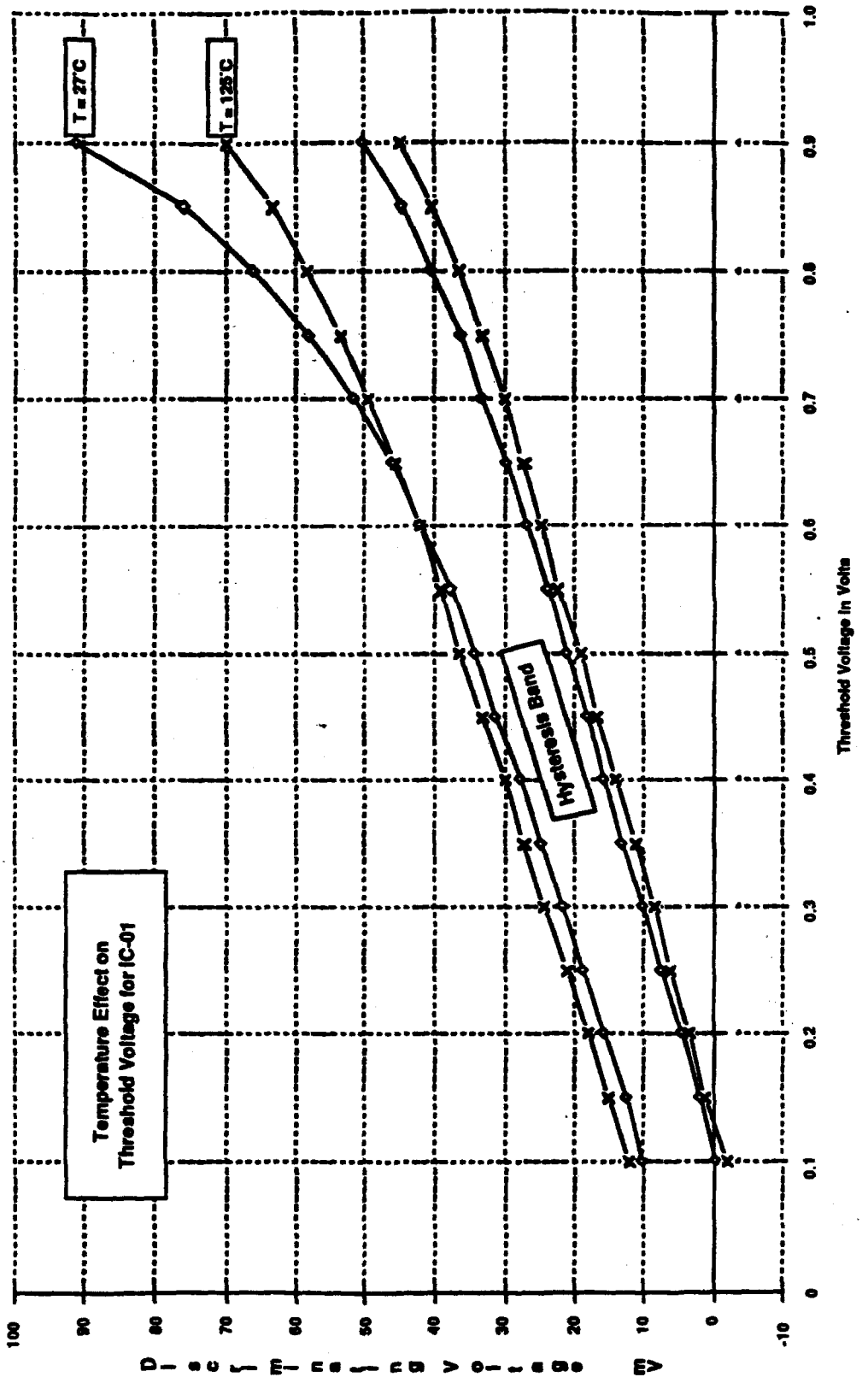
IC-01: SSD TWO CHANNEL SUM, DISCRIMINATOR and LATCH
Cavity Down

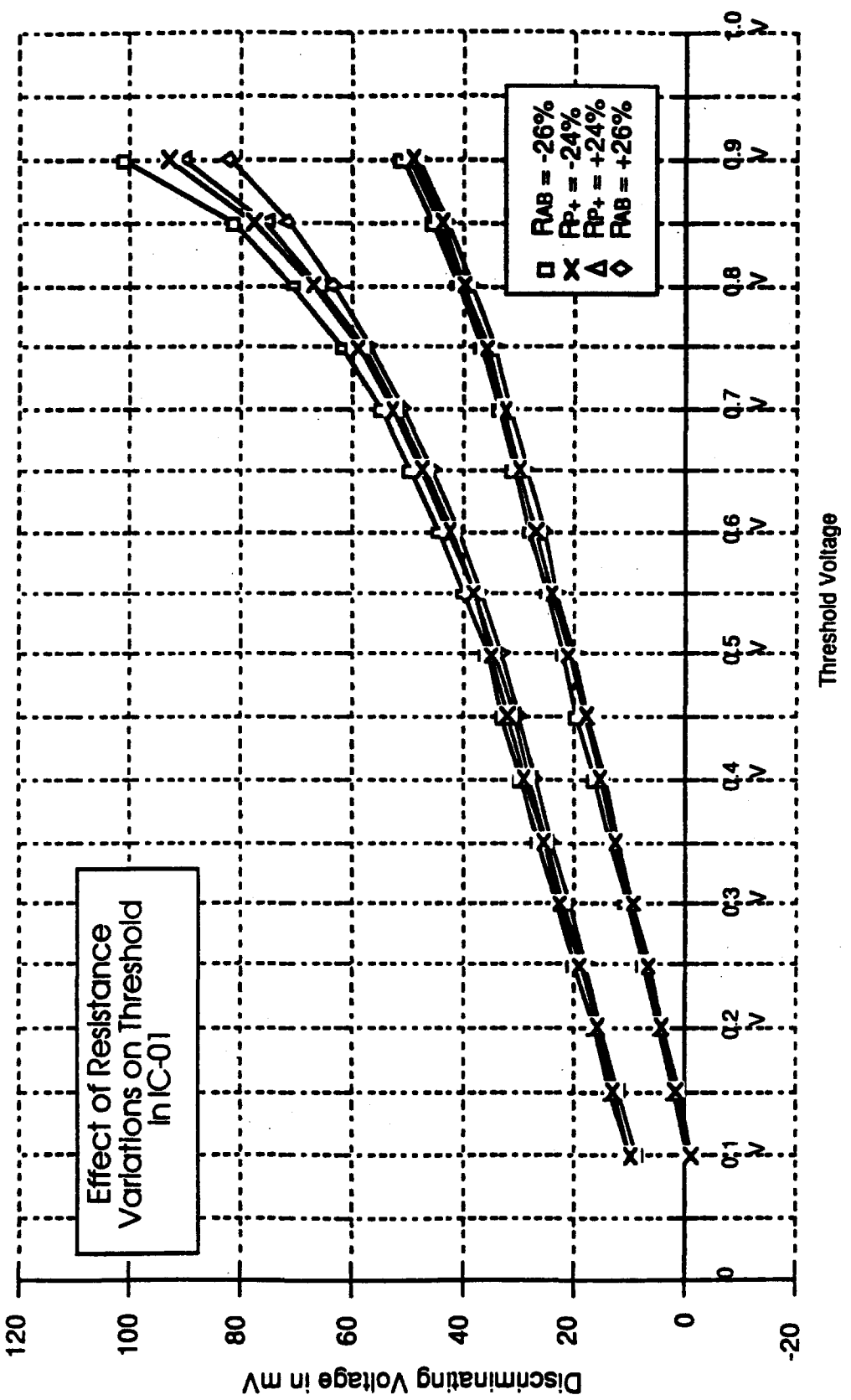
Part Number IC-01

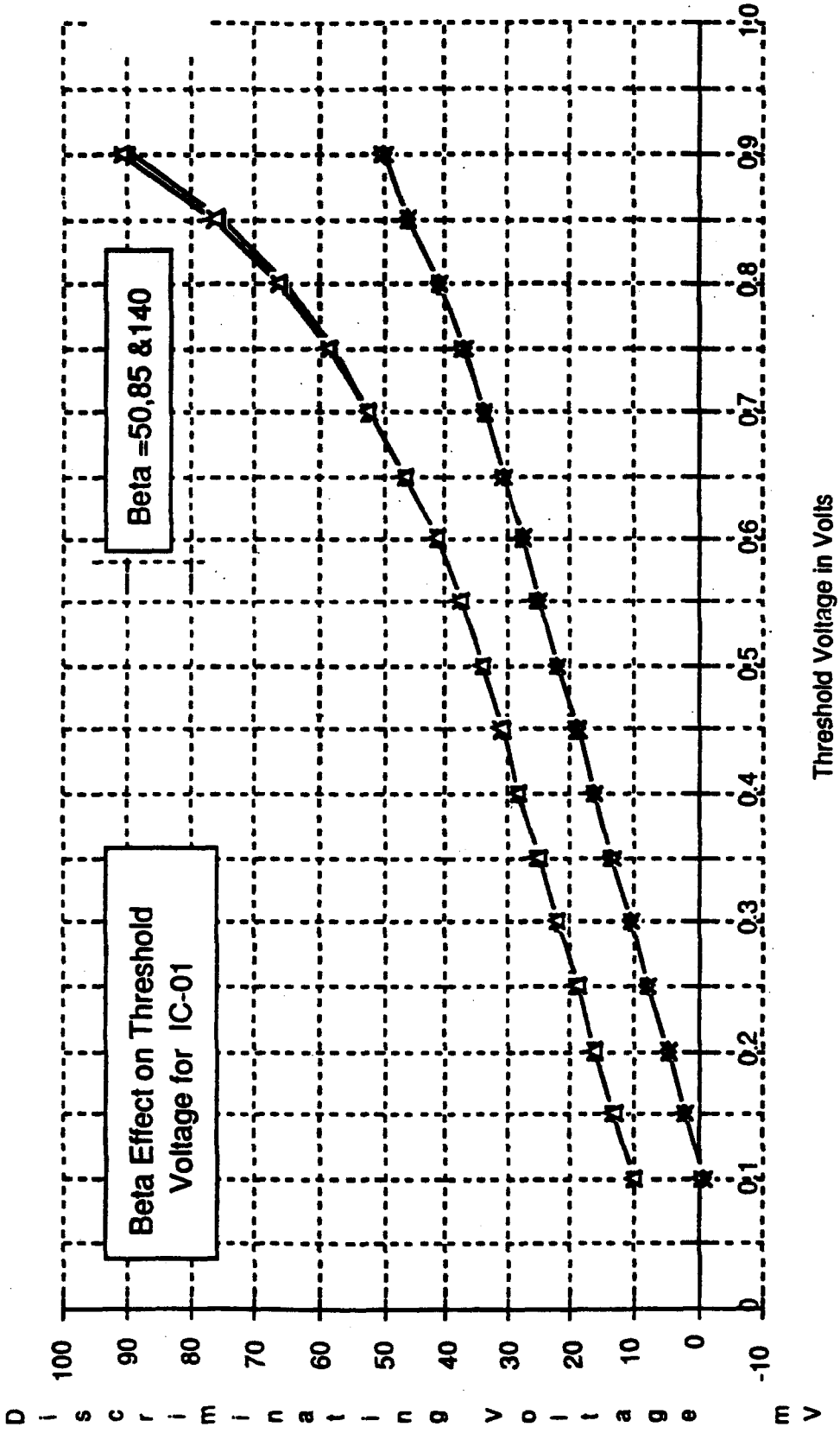


BOTTOM VIEW

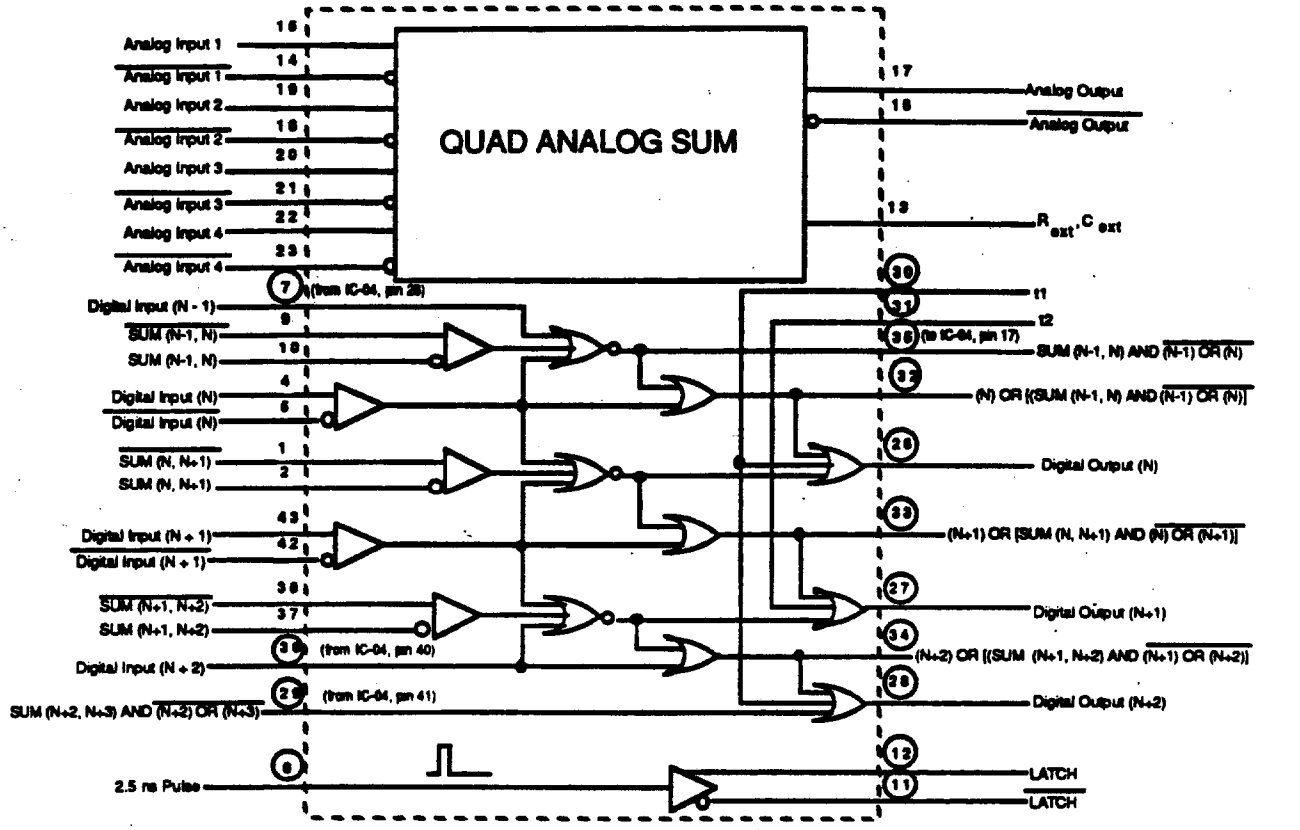
Created 1-24-89
Revised 7-14-89
Merle Haldeman / Scott Holm


T = 27°C
T = 125°C
Hysteresis Band
Temperature Effect on
Threshold Voltage for IC-01
Threshold Voltage in Volts
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-10
0
10
20
30
40
50
60
70
80
90
100


Effect of Resistance Variations on Threshold In IC-01
RAB = -26%
RP+ = -24%
RP+ = +24%
RAB = +26%
Discriminating Voltage in mV
120
100
80
60
40
20
0
-20
Threshold Voltage
0
0.1 V
0.2 V
0.3 V
0.4 V
0.5 V
0.6 V
0.7 V
0.8 V
0.9 V
1.0 V


Beta Effect on Threshold Voltage for IC-01
Beta =50,85 &140
Discriminating Voltage mV
100
90
80
70
60
50
40
30
20
10
0
-10
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Threshold Voltage in Volts

IC-02: 3-CHANNEL LOGIC,QUAD ANALOG SUM & LATCH DRIVER



IMPLICIT PINS

GROUND pins: 3 (GND1 - Logic Circuitry)
39 (GND2 - Logic Output)
24 (GND3 - Analog)

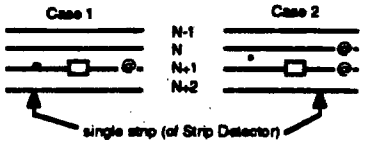
+ 3.5 V pins: 25 - 44

- 5.2 V pins: 6 - 40 - 41

NOTES

1. Digital outputs (pins # 26, 27, 28) go to the Auxiliary Backplane
2. Some Output Sums (pins # 32, 33, 34) go to the NHit section of IC-04
3. Lines entering circled pins follow ECL levels.
4. Power dissipation is 410 mW typical, 559 mW max.
5. Power supply currents: + 3.5 V: @ 19 mA Typical, 26 mA max.
   - 5.2 V: @ 66 mA Typical, 80 mA max.

RELATIONSHIP BETWEEN PART ICLE LOCATION IN THE DETECTOR AND PIC MOD ULE OUTPUT



- = Particle location
- = Signal sent to the NHit section of IC-04
- = Signal sent to the Auxiliary Backplane

TRUE TABLE

INPUTS					OUTPUTS	
N	SUM (N, N+1)	N+1	SUM (N+1, N+2)	N+2	N+1	N+1 OR [(SUM (N, N+1) AND (N) OR (N+1)]
X	X	1	X	X	1	1
0	1	0	X	X	1	1
X	X	0	1	0	1	

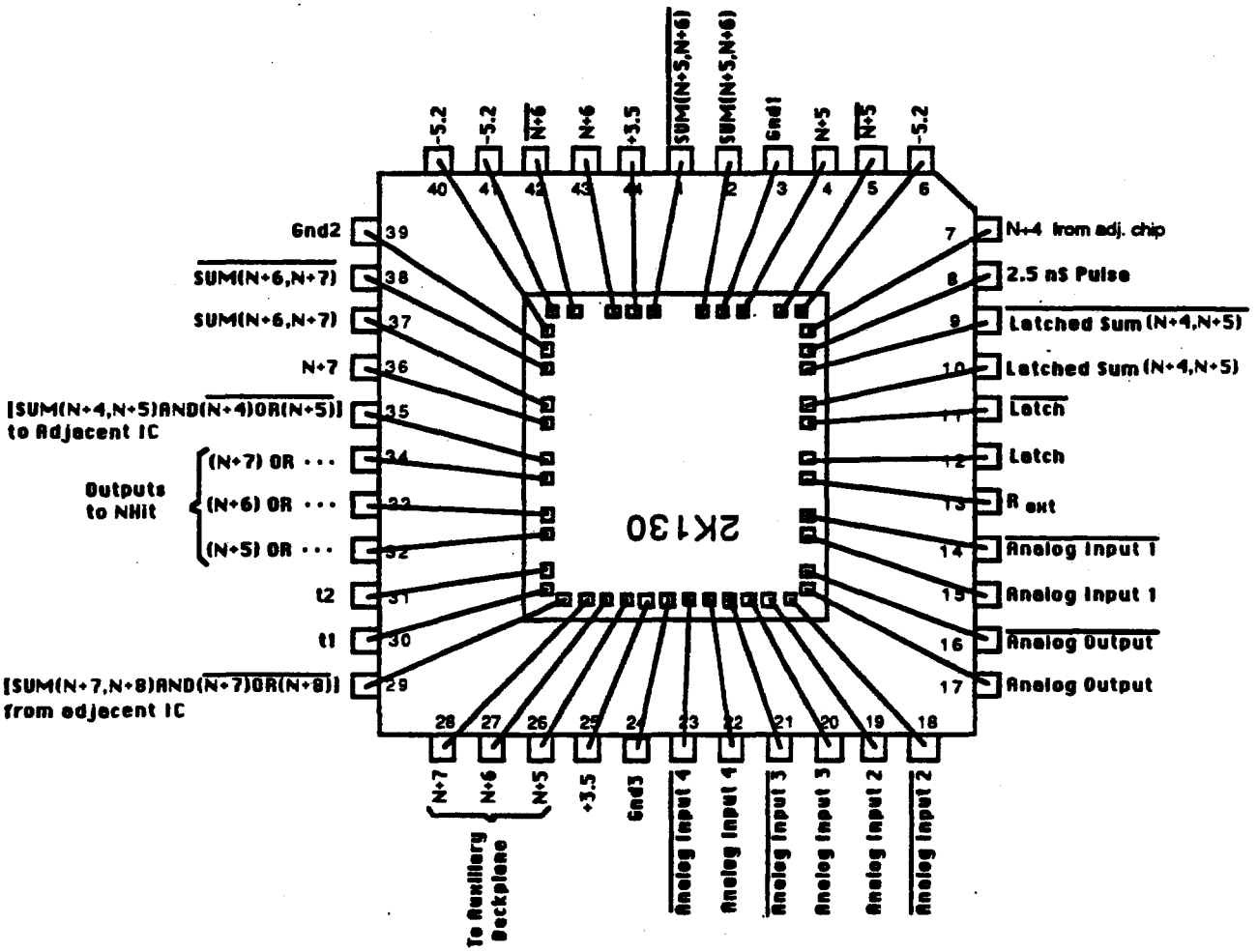
All other input conditions produce zero outputs. X = Don't Care

IC-02: SSD 3-CHANNEL LOGIC, QUAD ANALOG SUM and LATCH DRIVER

Cavity Down

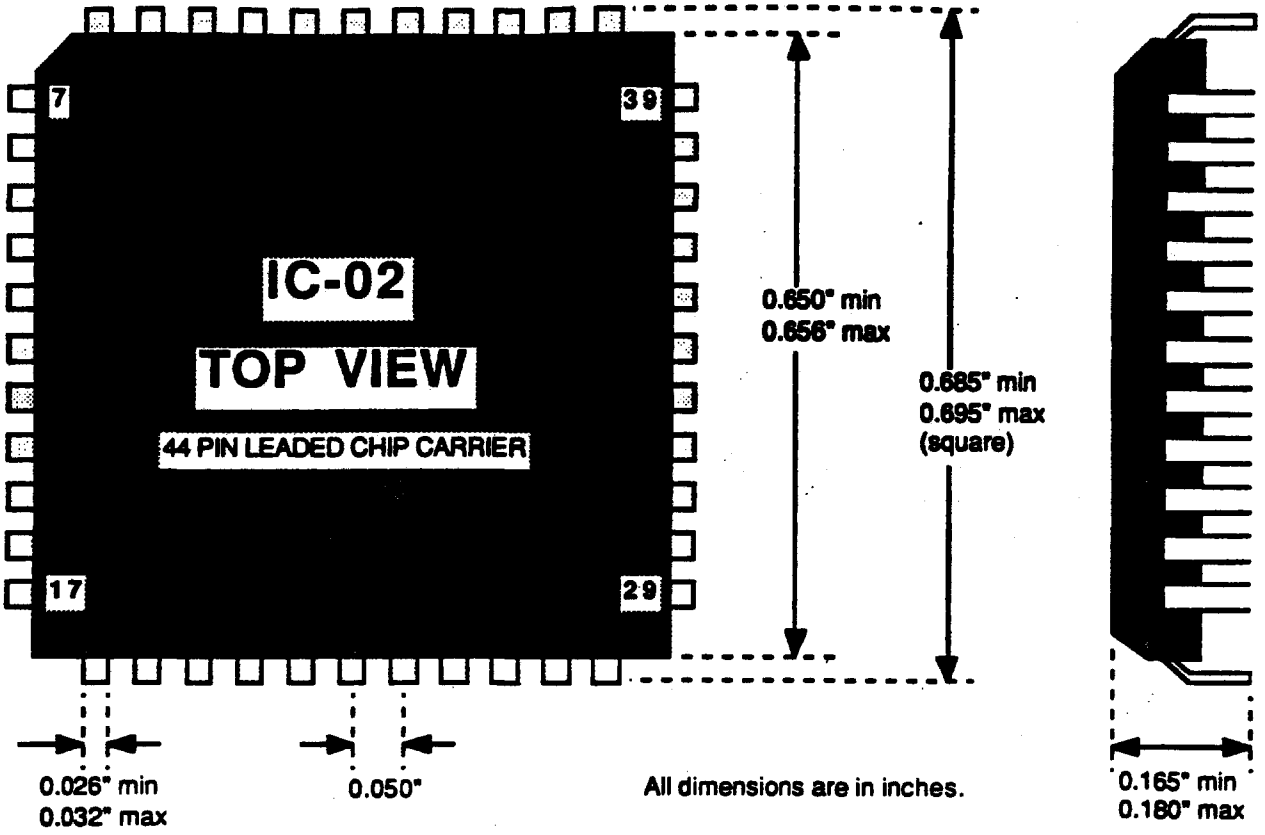
Thermally Enhanced, 44 Lead Plastic Chip Carrier

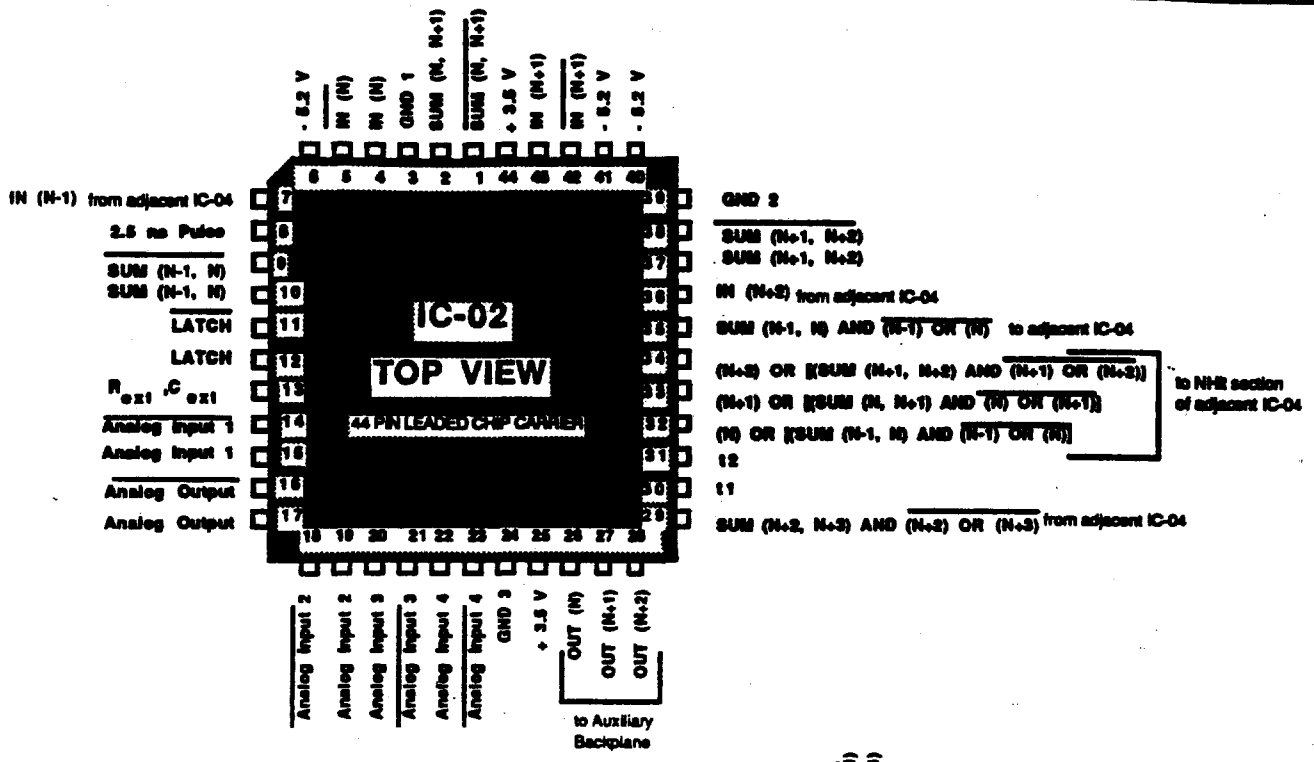
Part Number IC-02

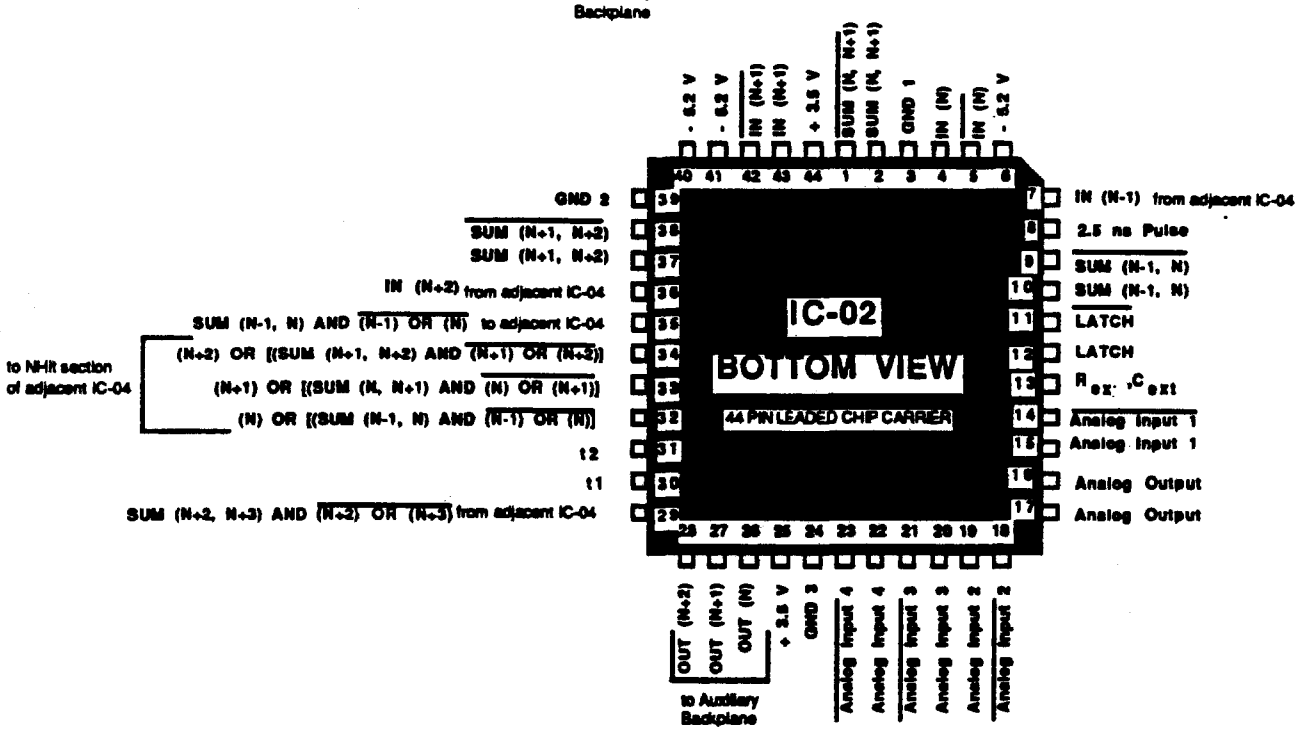


BOTTOM VIEW

Created 8-28-89
Revised
Merle Haldeman / Bruce Merkel

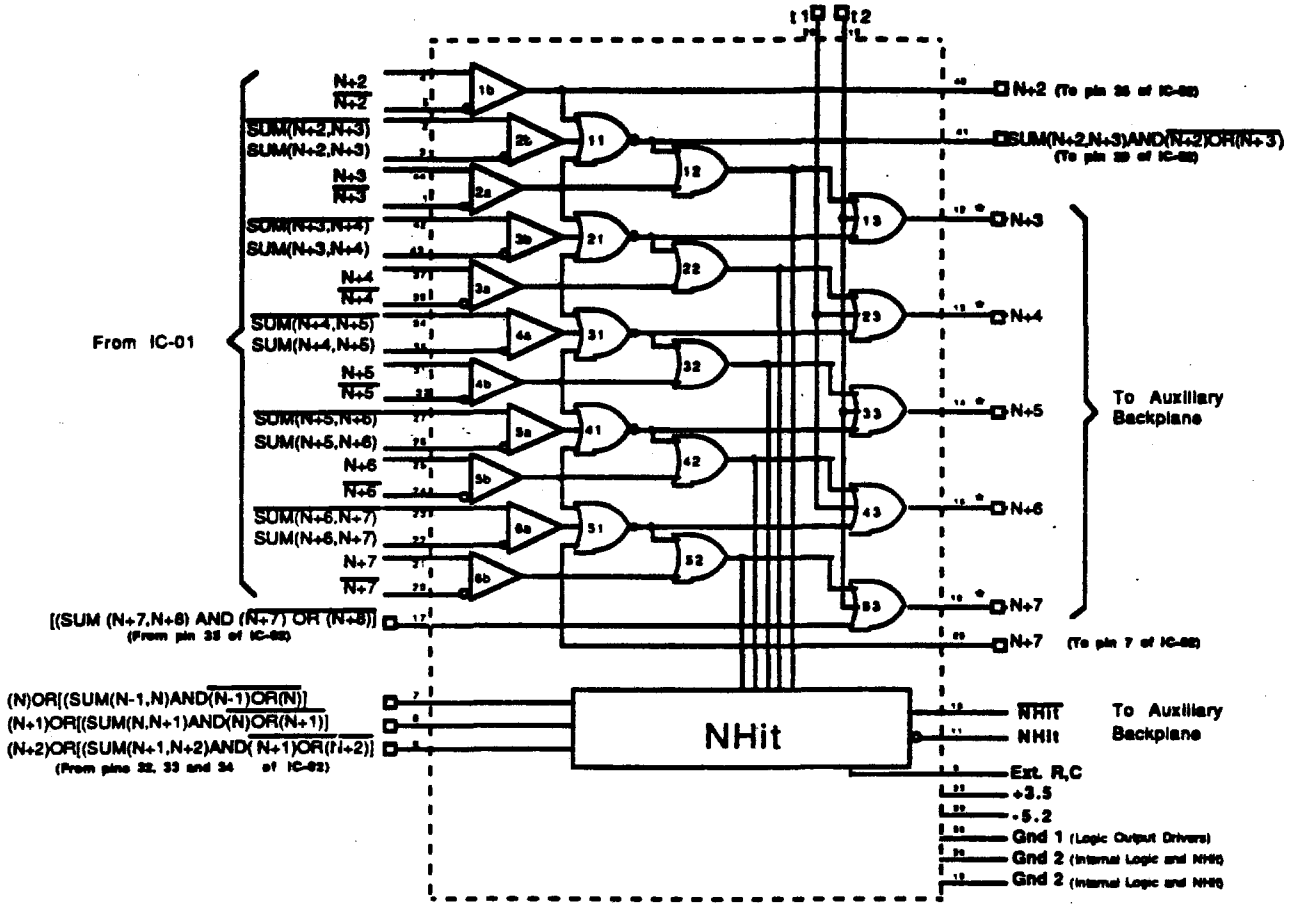

7
39
IC-02
TOP VIEW
44 PIN LEADED CHIP CARRIER
17
29
0.650" min
0.656" max
0.685" min
0.695" max
(square)
0.026" min
0.032" max
0.050"
All dimensions are in inches.
0.165" min
0.180" max


- 5.2 V
IN (N)
IN (N)
GND 1
SUM (N, N+1)
SUM (N, N+1)
+ 3.5 V
IN (N+1)
IN (N+1)
- 5.2 V
- 5.2 V
IN (N-1) from adjacent IC-04
2.5 ns Pulse
SUM (N-1, N)
SUM (N-1, N)
LATCH
LATCH
R_{ext} ,C_{ext}
Analog Input 1
Analog Input 1
Analog Output
Analog Output
IC-02
TOP VIEW
44 PIN LEADED CHIP CARRIER
GND 2
SUM (N+1, N+2)
SUM (N+1, N+2)
IN (N+2) from adjacent IC-04
SUM (N-1, N) AND (N-1) OR (N) to adjacent IC-04
(N+2) OR [(SUM (N+1, N+2) AND (N+1) OR (N+2)]
(N+1) OR [(SUM (N, N+1) AND (N) OR (N+1)]
(N) OR [(SUM (N-1, N) AND (N-1) OR (N)]
to NHit section of adjacent IC-04
t2
t1
SUM (N+2, N+3) AND (N+2) OR (N+3) from adjacent IC-04
Analog Input 2
Analog Input 2
Analog Input 3
Analog Input 3
Analog Input 4
Analog Input 4
GND 3
+ 3.5 V
OUT (N)
OUT (N+1)
OUT (N+2)
to Auxiliary Backplane


- 5.2 V
- 5.2 V
IN (N+1)
IN (N+1)
+ 3.5 V
SUM (N, N+1)
SUM (N, N+1)
GND 1
IN (N)
IN (N)
- 5.2 V
GND 2
SUM (N+1, N+2)
SUM (N+1, N+2)
IN (N+2) from adjacent IC-04
SUM (N-1, N) AND (N-1) OR (N) to adjacent IC-04
(N+2) OR [(SUM (N+1, N+2) AND (N+1) OR (N+2)]
(N+1) OR [(SUM (N, N+1) AND (N) OR (N+1)]
(N) OR [(SUM (N-1, N) AND (N-1) OR (N)]
to NHit section of adjacent IC-04
t2
t1
SUM (N+2, N+3) AND (N+2) OR (N+3) from adjacent IC-04
IC-02
BOTTOM VIEW
44 PIN LEADED CHIP CARRIER
IN (N-1) from adjacent IC-04
2.5 ns Pulse
SUM (N-1, N)
SUM (N-1, N)
LATCH
LATCH
R_{ext} ,C_{ext}
Analog Input 1
Analog Input 1
Analog Output
Analog Output
OUT (N+2)
OUT (N+1)
OUT (N)
+ 3.5 V
GND 3
Analog Input 4
Analog Input 4
Analog Input 3
Analog Input 3
Analog Input 2
Analog Input 2
to Auxiliary Backplane

IC-04: SSD 5-Channel Log ic and Octal NHit

Part Number IC-04 V2.0

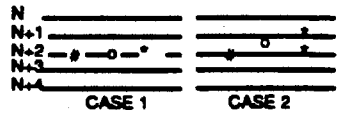


Power Dissipation: 514 mW typ.; 700 mW max. (Outputs: 100 ohms to -2.0 V)
Power supply currents: +3.5 @ 28 mA typ. ; 38 mA max.
-5.2 @ 80 mA typ. ; 109 mA max.

□ ECL Levels

INPUTS					OUTPUTS	
N	SUM(N,N+1)	N+1	SUM(N+1,N+2)	N+2	N+1	N+1 OR [""]
X	X	1	X	X	1	1
0	1	0	X	X	1	1
X	X	0	1	0	1	

All other input conditions produce zero outputs
X = Don't Care



o Location of particle in SSD
Signals to NHit
* Signals to Auxiliary Backplane

Created 4-23-90
Revised 4-2-90
Merle Haldeman / Jim Hoff

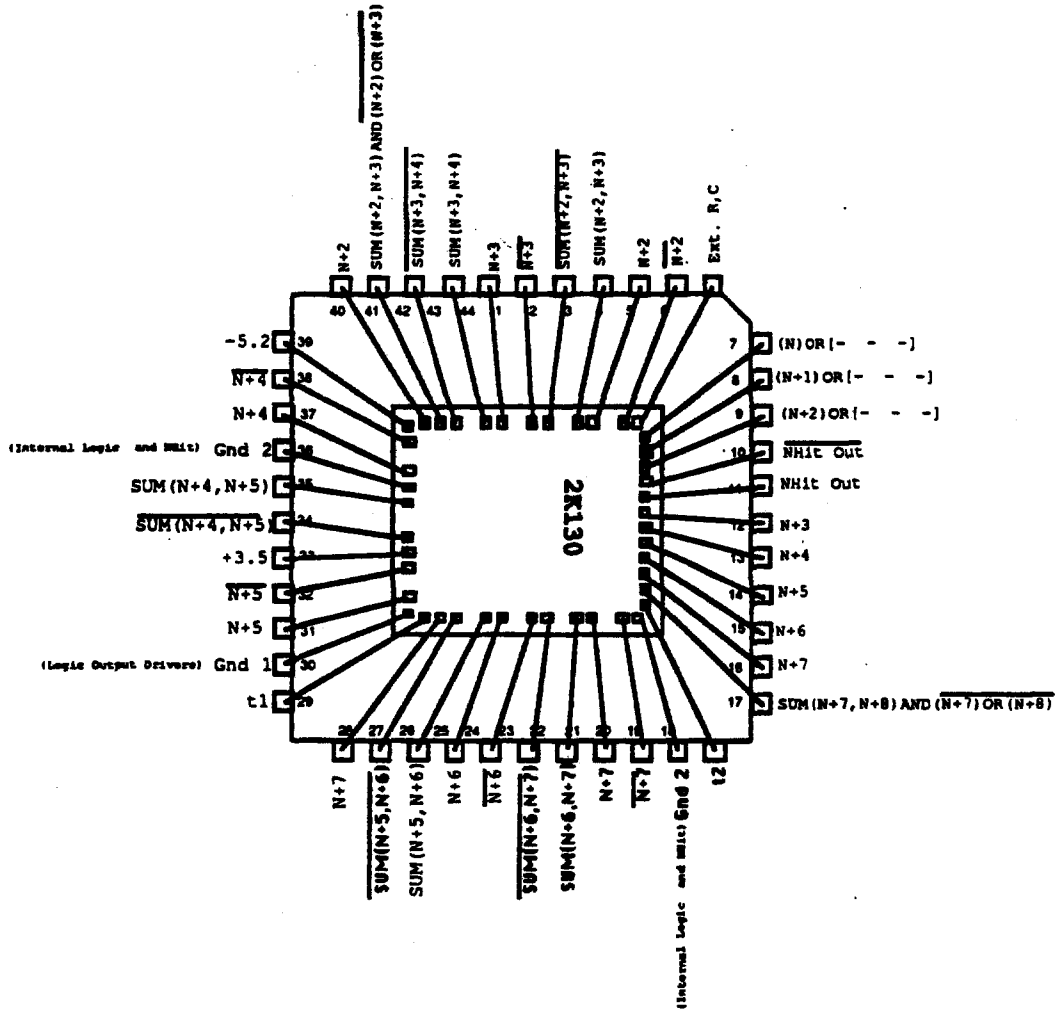
43 Pads Used

IC-04: SSD 5-CHANNEL LOGIC & OCTAL NHit

Cavity Down

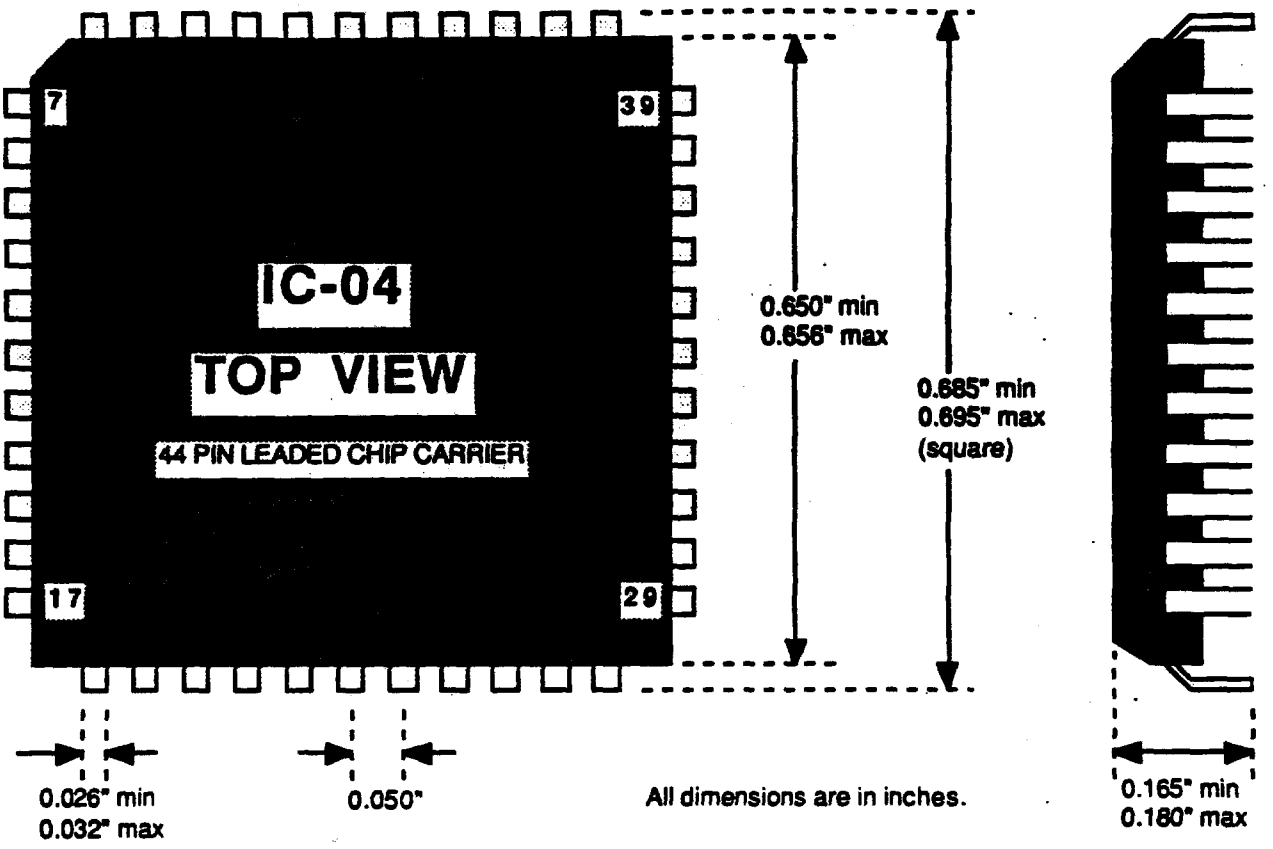
Thermally Enhanced, 44 Lead Plastic Chip Carrier

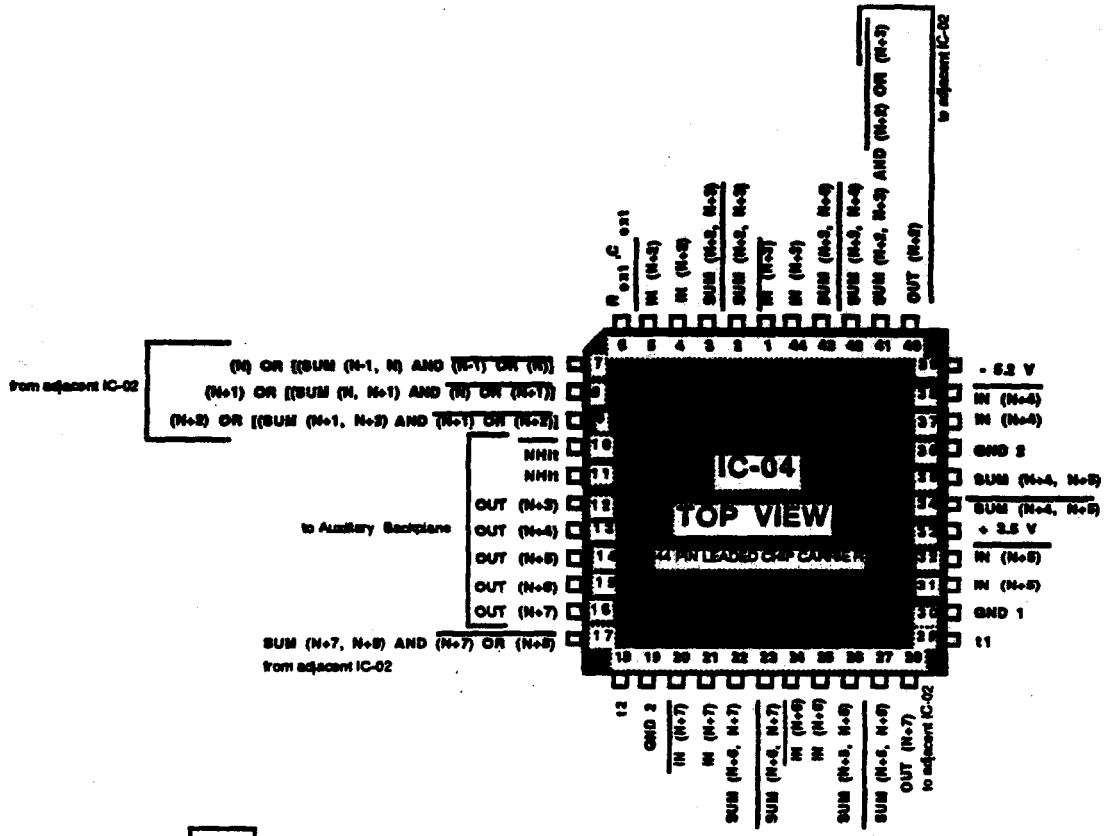
Part Number IC-04, V2.0

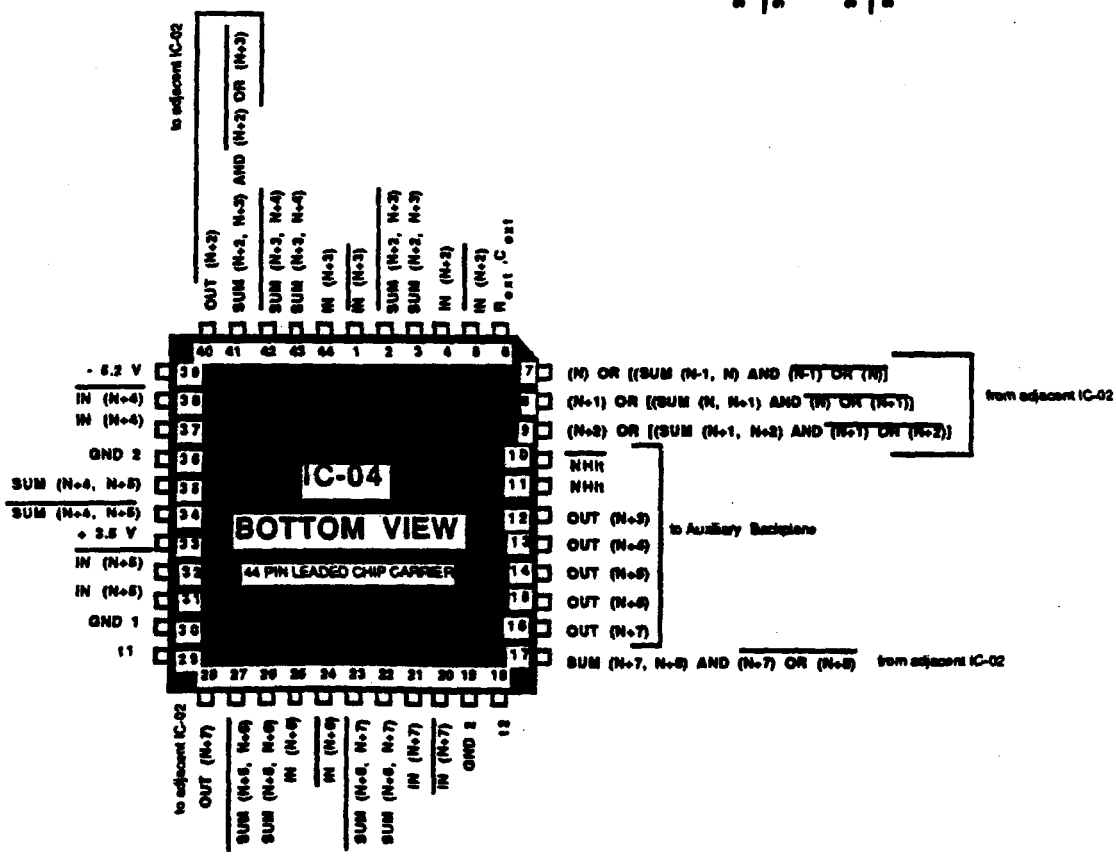


BOTTOM VIEW

Created 8-28-89
Revised 10-13-89
Merle Haldeman / Jim Hoff


7
39
17
29
IC-04
TOP VIEW
44 PIN LEADED CHIP CARRIER
0.650" min
0.656" max
0.685" min
0.695" max
(square)
0.026" min
0.032" max
0.050"
All dimensions are in inches.
0.165" min
0.180" max


IC-04
TOP VIEW
44 PIN LEADED CHIP CARRIER
from adjacent IC-02
to Auxiliary Backplane
to adjacent IC-02
- 5.2 V
+ 3.5 V
GND 1
GND 2
t1


IC-04
BOTTOM VIEW
44 PIN LEADED CHIP CARRIER
from adjacent IC-02
to Auxiliary Backplane
to adjacent IC-02
- 5.2 V
+ 3.5 V
GND 1
GND 2
t1

MOTOROLA

MC10109

DUAL 4-5-INPUT "OR/NOR" GATE

The MC10109 is a dual 4-5 input OR/NOR gate.

P_D = 30 mW typ/gate (No Load)
t_{pd} = 2.0 ns typ
t_r, t_f = 2.0 ns typ (20%–80%)

MECL 10K

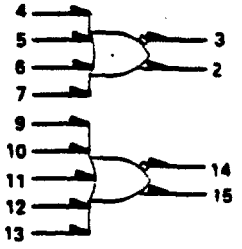
DUAL 4-5-INPUT "OR/NOR" GATE

L SUFFIX
CERAMIC PACKAGE
CASE 620

P SUFFIX
PLASTIC PACKAGE
CASE 648

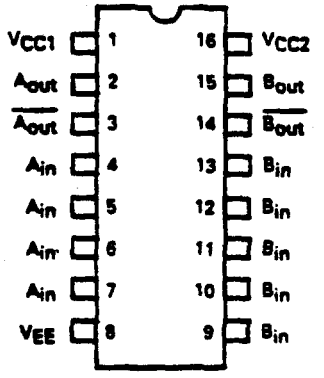
FN SUFFIX
PLCC
CASE 775

LOGIC DIAGRAM



V_{CC1} = Pin 1
V_{CC2} = Pin 16
V_{EE} = Pin 8

DIP PIN ASSIGNMENT



Pin assignment is for Dual-in-line Package.
For PLCC pin assignment, see tables on page I-31.

3

ELECTRICAL CHARACTERISTICS

Each MECL 10,000 series circuit has been designed to meet the dc specifications shown in the test table, after thermal equilibrium has been established. The circuit is in a test socket or mounted on a printed circuit board and transverse air flow greater than 500 linear fpm is maintained. Outputs are terminated through a 50-ohm resistor to −2.0 volts. Test procedures are shown for only one input and one output. The other inputs and outputs are tested in the same manner.

@ Test Temperature	TEST VOLTAGE VALUES (Volts)				
	$V_{IH\ max}$	$V_{IL\ min}$	$V_{IHA\ min}$	$V_{ILA\ max}$	V_{EE}
-30°C	-0.890	-1.890	-1.205	-1.500	-5.2
+25°C	-0.810	-1.850	-1.105	-1.475	-5.2
+85°C	-0.700	-1.825	-1.035	-1.440	-5.2

Characteristic	Symbol	Pin Under Test	MC10103 Test Limits -30°C Min	-30°C Max	+25°C Min	+25°C Typ	+25°C Max	+85°C Min	+85°C Max	Unit	TEST VOLTAGE APPLIED TO PINS BELOW: $V_{IH\ max}$	$V_{IL\ min}$	$V_{IHA\ min}$	$V_{ILA\ max}$	V_{EE}	(V_{CC}) Gnd
Power Supply Drain Current	I_E	8	-	15	-	11	14	-	15	mAdc	-		-		8	1,16
Input Current	I_{inH}	4		425	-	-	265	-	265	μAdc	4	-			8	1,16
	I_{inL}	4	0.5	-	0.5	-	-	0.3		μAdc	-	4			8	1,16
High Output Voltage	V_{OH}	2	-1.050	-0.890	-0.980		-0.810	-0.890	-0.700	Vdc	4	-			8	1,16
		3	-1.050	-0.890	-0.980	-	-0.810	-0.890	-0.700	Vdc	-	-			8	1,16
Low Output Voltage	V_{OL}	2	-1.890	-1.675	-1.850	-	-1.650	-1.826	-1.615	Vdc	-	-			8	1,16
		3	-1.890	-1.675	-1.850	-	-1.650	-1.826	-1.615	Vdc	4				8	1,16
High Threshold Voltage	V_{OHA}	2	-1.080	-	-0.980			-0.910		Vdc	-	-	4	-	8	1,16
		3	-1.080		-0.980	-	-	-0.910		Vdc	-	-	-	4	8	1,16
Low Threshold Voltage	V_{OLA}	2	-	-1.655	-	-	-1.630	-	-1.595	Vdc		-	-	4	8	1,16
		3		-1.655	-	-	-1.630		-1.595	Vdc		-	4	-	8	1,16
Switching Times (50-ohm load)													Pulse In	Pulse Out	-3.2 V	+2.0 V
Propagation Delay	t_{4+2+}	2	1.0	3.7	1.0	2.0	2.9	1.0	3.7	ns			4	2	8	1,16
	t_{4-2-}	2										-		2		
	t_{4+3-}	3									-	-		3		
	t_{4-3+}	3										-		3		
Rise Time (20 to 80%)	t_{2+}	2	1.1	4.0	1.1		3.3	1.1	4.0		-	-		2		
	t_{3+}	3									-	-		3		
Fall Time (20 to 80%)	t_{2-}	2									-	-		2		
	t_{3-}	3									-	-		3		

MOTOROLA

MC10124

QUAD TTL TO MECL TRANSLATOR

The MC10124 is a quad translator for interfacing data and control signals between a saturated logic section and the MECL section of digital systems. The MC10124 has TTL compatible inputs, and MECL complementary open-emitter outputs that allow use as an inverting/non-inverting translator or as a differential line driver. When the common strobe input is at the low logic level, it forces all true outputs to a MECL low logic state and all inverting outputs to a MECL high logic state.

Power supply requirements are ground, +5.0 Volts, and −5.2 Volts. Propagation delay of the MC10124 is typically 3.5 ns. The dc levels are standard or Schottky TTL in, MECL 10,000 out.

An advantage of this device is that TTL level information can be transmitted differentially, via balanced twisted pair lines, to the MECL equipment, where the signal can be received by the MC10115 or MC10116 differential line receivers. The MC10124 is useful in computers, instrumentation, peripheral controllers, test equipment, and digital communications systems.

P_D	= 380 mW typ/pkg (No Load)
t_{pd}	= 3.5 ns typ (+ 1.5 Vdc in to 50% out)
t_r, t_f	= 2.5 ns typ (20%–80%)

MECL 10K

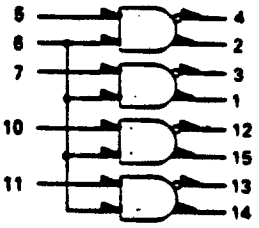
QUAD TTL TO MECL TRANSLATOR

L SUFFIX
CERAMIC PACKAGE
CASE 620

P SUFFIX
PLASTIC PACKAGE
CASE 648

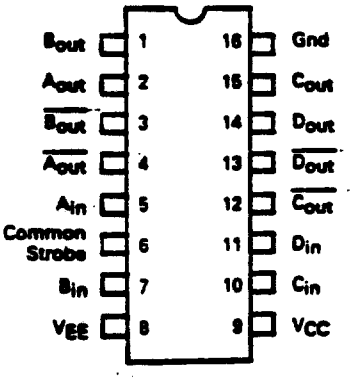
FN SUFFIX
PLCC
CASE 775

LOGIC DIAGRAM



Gnd = Pin 16
V_{CC} (+5.0 Vdc) = Pin 9
V_{EE} (−5.2 Vdc) = Pin 8

DIP PIN ASSIGNMENT



Pin assignment is for Dual-in-line Package.
For PLCC pin assignment, see tables on page 1-31.

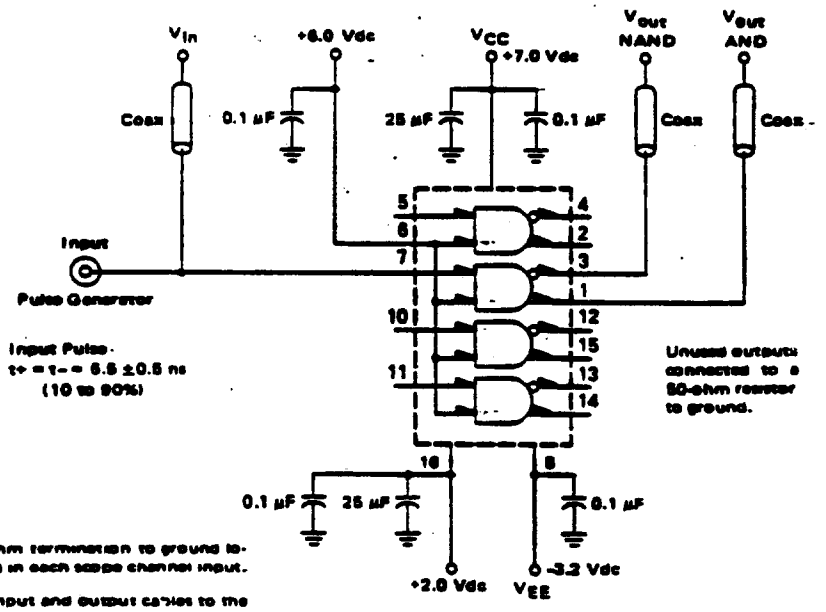
ELECTRICAL CHARACTERISTICS

Each MECL 10,000 series circuit has been designed to meet the dc specifications shown in the test table, after thermal equilibrium has been established. The circuit is in a test socket or mounted on a printed circuit board and transverse air flow greater than 500 linear fpm is maintained. Outputs are terminated through a 50-ohm resistor to −2.0 volts. Test procedures are shown for only one translator. The other translators are tested in the same manner.

			MC10124 Test Limits								TEST VOLTAGE/CURRENT VALUES										
											Volts								mA		
@ Test Temperature											V_{IH}	$V_{IL\ max}$	V_{IHA}	V_{ILA}	V_F	V_R	V_{CC}	V_{EE}	I_J	I_m	
-30°C											+4.0	+0.48	+2.06	+1.16	+0.48	+2.46	+5.00	-5.2	30	+1.0	
+25°C											+4.0	+0.48	+1.80	+1.16	+0.48	+2.46	+5.00	-5.2	-10	+1.0	
+85°C											+4.0	+0.48	+1.80	+0.50	+0.48	+2.46	+5.00	-5.2	10	+1.0	
			-30°C		+25°C			+85°C			TEST VOLTAGE/CURRENT APPLIED TO PINS LISTED BELOW:										
Characteristic	Symbol	Pin Under Test	Min	Max	Min	Typ	Max	Min	Max	Unit	V_{IH}	$V_{IL\ max}$	V_{IHA}	V_{ILA}	V_F	V_R	V_{CC}	V_{EE}	I_J	I_m	Gnd
Negative Power Supply Drain Current	I_E	8	-	72		-	-60	-	72	mAdc	-	-	-	-	-	-	5	8	-		16
Positive Power Supply Drain Current	I_{CCH}	9		18		-	16	-	18	mAdc	5,6,7,10,11	-	-		-		5	8			16
	I_{CCL}	9	-	26	-	-	26	-	26	mAdc		-	-		-		5	8			5,6,7,10,11,16
Reverse Current	I_R	6	-	200	-	-	200	-	200	μAdc	-	-	-		5,7,10,11	6	5	8			16
		7	-	50	-	-	50	-	50	μAdc	-	-	-		6	7	5	8			16
Forward Current	I_F	6	-	-12.0	-	-	-12.0	-	-12.0	mAdc	5,7,10,11	-	-		6		5	8			16
		7	-	-3.2	-	-	-3.2		-3.2	mAdc	6	-	-		7		5	8			16
Input Breakdown Voltage	BV_{in}	6	5.5		5.5	-	-	5.5	-	Vdc	-	-	-		-		5	8		6	5,7,10,11,16
		7	5.5	-	5.5	-	-	5.5	-	Vdc	-	-	-		-		5	8		7	6,16
Clamp Input Voltage	V_I	6	-	-1.5	-	-	-1.5	-	-1.5	Vdc	-	-	-				5	8	6		16
		7		-1.5		-	-1.5		-1.5	Vdc	-	-	-				5	8	7		16
High Output Voltage	V_{OH}	1	-1.060	-0.890	-0.960	-	-0.810	-0.890	-0.700	Vdc	6,7		-				5	8			16
		3	-1.060	-0.890	-0.960	-	-0.810	-0.890	-0.700	Vdc		6,7					5	8			16
Low Output Voltage	V_{OL}	1	-1.960	-1.675	-1.950	-	-1.650	-1.825	-1.615	Vdc	-	6,7	-				5	8			16
		3	-1.960	-1.675	-1.950	-	-1.650	-1.825	-1.615	Vdc	6,7		-				5	8			16
High Threshold Voltage	V_{OHA}	1	-1.060	-	-0.960	-	-	-0.810		Vdc	6		7		-		5	8			16
		3	-1.060	-	-0.960	-	-	-0.810		Vdc	6		-	7			5	8			16
Low Threshold Voltage	V_{OLA}	1	-	-1.665		-	-1.630		-1.595	Vdc	6	-	-	7			5	8			16
		3		-1.666	-	-	-1.630		-1.596	Vdc	6	-	7	-	-		5	8			16
Switching Time (50 Ω load)											+0.8 Vdc	Pulse In	Pulse Out				+7.0 Vdc	-3.2 Vdc			+2.0 Vdc
Propagation Delay (+3.5 Vdc to 50%)①	t_{P+1+}	1	1.6	6.0	1.0	3.6	6.0	1.0	6.0	ns	7	6	1				5	8			16
	t_{P-1-}		1.0	6.0				1.0	6.0		7	6									
	t_{P1+}		1.6	6.0				1.0	6.0		6	7									
	t_{P1-}		1.0	6.0				1.0	6.0												
	t_{P3+}		1.6	6.0				1.0	6.0												
	t_{P3-}	3	1.0	6.0				1.0	6.0				3								
Rise Time (20% to 80%)	t_{1+}	1	1.0	4.2	1.1	2.5	3.9	1.1	4.3				1								
Fall Time (80% to 20%)	t_{1-}	1			1.1	2.5	3.9						1								

① See switching time test circuit. Propagation delay for this circuit is specified from +1.5 Vdc on to the 50% point on the output waveform. The +3.5 Vdc is shown here because all logic and supply levels are shifted 2 volts positive.)

SWITCHING TIME TEST CIRCUIT



NOTE: All power supply and logic levels are shown shifted 2 volts positive.

MOTOROLA

MC10125

QUAD MECL TO TTL TRANSLATOR

The MC10125 is a quad translator for interfacing data and control signals between the MECL section and saturated logic sections of digital systems. The MC10125 incorporates differential inputs and Schottky TTL "totem pole" outputs. Differential inputs allow for use as an inverting/non-inverting translator or as a differential line receiver. The V_{BB} reference voltage is available on pin 1 for use in single-ended input biasing. The outputs of the MC10125 go to a low logic level whenever the inputs are left floating.

Power supply requirements are ground, +5.0 Volts and −5.2 Volts. Propagation delay of the MC10125 is typically 4.5 ns. The MC10125 has fanout of 10 TTL loads. The dc levels are MECL 10,000 in and Schottky TTL, or TTL out. This device has an input common mode noise rejection of ± 1.0 Volt.

An advantage of this device is that MECL level information can be received, via balanced twisted pair lines, in the TTL equipment. This isolates the MECL logic from the noisy TTL environment. This device is useful in computers, instrumentation, peripheral controllers, test equipment and digital communications systems.

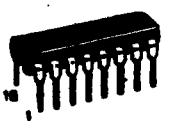
P_D	= 380 mW typ/pkg (No Load)
t_{pd}	= 4.5 ns typ (50% to + 1.5 Vdc out)
t_r, t_f	= 2.5 ns typ (1.0 V to 2.0 V)

MECL 10K

QUAD MECL TO TTL TRANSLATOR

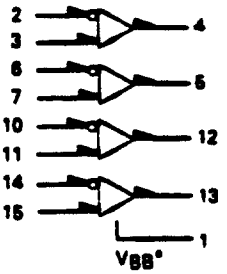
L SUFFIX
CERAMIC PACKAGE
CASE 620

P SUFFIX
PLASTIC PACKAGE
CASE 648



FN SUFFIX
PLCC
CASE 775

LOGIC DIAGRAM

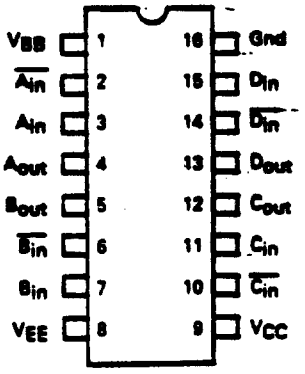


Gnd = Pin 16
V_{CC} (+5.0 Vdc) = Pin 9
V_{EE} (−5.2 Vdc) = Pin 8

*V_{BB} to be used to supply bias to the MC10125 only and bypassed (when used) with 0.01 μF to 0.1 μF capacitor.

When the input pin with the bubble goes positive the output goes negative.

DIP PIN ASSIGNMENT



Pin assignment is for Dual-in-line Package.
For PLCC pin assignment, see tables on page 1-31.

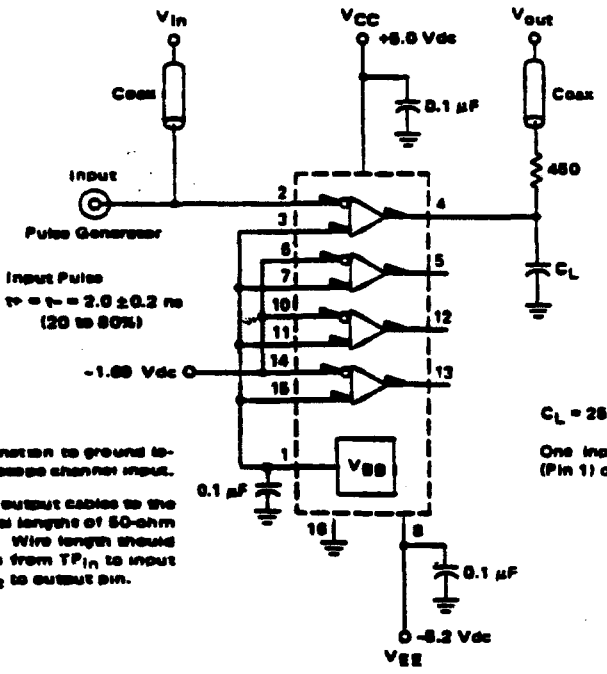
ELECTRICAL CHARACTERISTICS

Each MECL 10,000 series circuit has been designed to meet the dc specifications shown in the test table, after thermal equilibrium has been established. The circuit is in a test socket or mounted on a printed circuit board and transverse air flow greater than 500 linear fpm is maintained. Test procedures are shown for only one translator. The other translators are tested in the same manner.

Characteristic	Symbol	Pin Under Test	MC10125 Test Limits -30°C Min	-30°C Max	+25°C Min	+25°C Typ	+25°C Max	+85°C Min	+85°C Max	Unit	TEST VOLTAGE VALUES (Volts) @ Test Temperature -30°C: $V_{IH\,max}$ -0.890, $V_{IL\,min}$ -1.890, $V_{IHA\,min}$ -1.205, $V_{ILA\,max}$ -1.840, V_{IHH} +0.110, V_{ILH} -0.890, V_{IHL} -1.890, V_{ILL} -2.880; +25°C: -0.810, -1.850, -1.165, -1.475, +0.190, -0.950, -1.810, -2.850; +85°C: -0.700, -1.825, -1.035, -1.440, +0.300, -0.975, -1.700, -2.825; V_{BB} From Pin 1; V_{CC} +5.0; V_{EE} -5.2	Gnd	Output Conditions
											TEST VOLTAGE APPLIED TO PINS LISTED BELOW		
Negative Power Supply Drain Current	I_E	8	-	-44	-	-	-60	-	-64	mAdc	-	3,7,11,15	-
Positive Power Supply Drain Current	I_{CCH}	9	-	52	-	-	52	-	52	mAdc	2,6,10,14	3,7,11,15	-
	I_{CCL}	9	-	38	-	-	38	-	38	mAdc	-	3,7,11,15	-
Input Current	I_{inH} ①	2	-	180	-	-	115	-	115	µAdc	2,6,10,14	3,7,11,15	-
Input Leakage Current	I_{CBO}	2	-	1.5	-	-	10	-	1.0	µAdc	-	3,7,11,15	2,6,8,10,14
High Output Voltage	V_{OH}	4	3.5	-	3.5	-	-	3.5	-	Vdc	-	3,7,11,15	-2.0 mA
Low Output Voltage	V_{OL}	4	-	0.5	-	-	0.5	-	0.5	Vdc	2,6,10,14	3,7,11,15	20 mA
High Threshold Voltage	V_{OHA}	4	3.5	-	3.5	-	-	3.5	-	Vdc	-	3,7,11,15	-2.0 mA
Low Threshold Voltage	V_{OLA}	4	-	0.5	-	-	0.5	-	0.5	Vdc	6,10,14	3,7,11,15	20 mA
Input Current Test	V_{OLS1}	4	-	0.5	-	-	0.5	-	0.5	Vdc	-	3,7,11,15	2,3,6,7,8,10,11,14,15
	V_{OLS2}	4	-	0.5	-	-	0.5	-	0.5	Vdc	-	3,7,11,15	20 mA
Short-Circuit Current	I_{OS}	4	40	100	40	-	100	40	100	mA	-	3,7,11,15	4,16
Reference Voltage	V_{BB}	1	-1.420	-1.38	-1.350	-	-1.230	-1.295	-1.150	Vdc	-	3,7,11,15	-
Common Mode Rejection Tests	V_{OH}	4	3.5	-	3.5	-	-	3.5	-	Vdc	-	3	7
		5	3.5	-	3.5	-	-	3.5	-		-	-	-
	V_{OL}	4	-	0.5	-	-	0.5	-	0.5	Vdc	-	2	3
		5	-	0.5	-	-	0.5	-	0.5		-	-	-
Switching Times											Pulse In	Pulse Out	C_L (pF)
Propagation Delay (50% to +1.5 Vdc)	t_{D+}	8	1.0	6.0	1.0	4.5	6.0	1.0	6.0	ns	6	5	20
	t_{D-}	8									6	5	
	t_{D+}	4									3	4	
	t_{D-}	4											
Rise Time (+1.5 Vdc to 2.5 Vdc)	t_{r+}		-	3.3	+	-	3.3	-	3.3				
Fall Time (+1.5 Vdc to 2.5 Vdc)	t_{f-}		-	3.3	-	-	3.3	-	3.3				

① Individually test each input, apply $V_{IH\,max}$ to pin under test

SWITCHING TIME TEST CIRCUIT



50-ohm termination to ground located in each scope channel input.

All input and output cables to the scope are equal lengths of 50-ohm coaxial cable. Wire length should be < 1/4 inch from TP_{in} to input pin and TP_{out} to output pin.

C_L = 28 pF, including test fixture

One input from each gate must be tied to V_{BB} (Pin 1) during testing.

MC54/74HC4514

1-of-16 Decoder/Demultiplexer With Address Latch

High-Performance Silicon-Gate CMOS

The MC54/74HC4514 is identical in pinout to the MC14514B metal-gate CMOS device. The device inputs are compatible with standard CMOS outputs; with pullup resistors, they are compatible with LSTTL outputs.

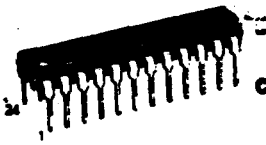
This device consists of a 4-bit storage latch with a Latch Enable and Chip Select input. When a low signal is applied to the Latch Enable input, the Address is stored, and decoded. When the Chip Select input is high, all sixteen outputs are forced to a low level.

The Chip Select input is provided to facilitate the chip-select, demultiplexing, and cascading functions.

The demultiplexing function is accomplished by using the Address inputs to select the desired device output, and then by using the Chip Select as a data input.

- Output Drive Capability: 10 LSTTL Loads
- Outputs Directly Interface to CMOS, NMOS, and TTL
- Operating Voltage Range: 2 to 6 V
- Low Input Current: 1 μA
- High Noise Immunity Characteristic of CMOS Devices
- In Compliance with the Requirements Defined by JEDEC Standard No. 7A
- Chip Complexity: 268 FETs or 67 Equivalent Gates

J SUFFIX
CERAMIC
CASE 709-01



N SUFFIX
PLASTIC
CASE 724-03

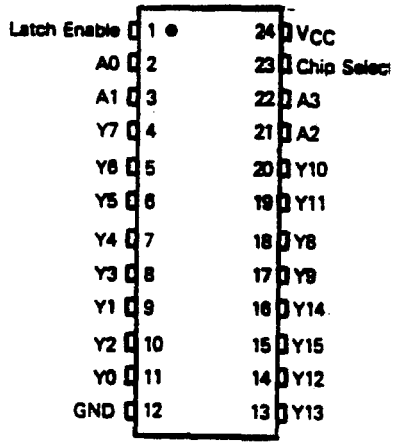
DW SUFFIX
SOIC
CASE 751E-02

ORDERING INFORMATION

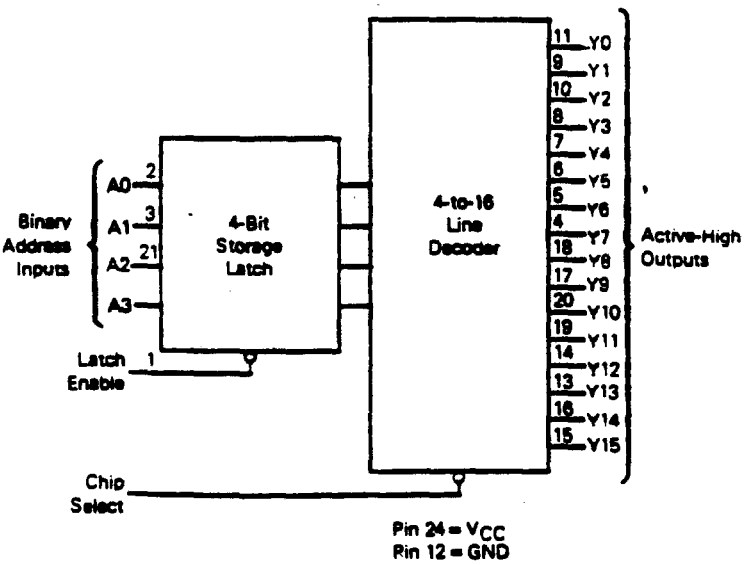
MC74HCXXXXN	Plastic
MC54HCXXXXJ	Ceramic
MC74HCXXXXDW	SOIC

$T_A = -55°$ to 125°C for all packages.
Dimensions in Chapter 7.

PIN ASSIGNMENT



BLOCK DIAGRAM



Pin 24 = V_{CC}
Pin 12 = GND

MAXIMUM RATINGS*

Symbol	Parameter	Value	Unit
V_{CC}	DC Supply Voltage (Referenced to GND)	−0.5 to +7.0	V
V_{in}	DC Input Voltage (Referenced to GND)	−1.5 to V_{CC}+1.5	V
V_{out}	DC Output Voltage (Referenced to GND)	−0.5 to V_{CC}+0.5	V
I_{in}	DC Input Current, per Pin	±20	mA
I_{out}	DC Output Current, per Pin	±25	mA
I_{CC}	DC Supply Current, V_{CC} and GND Pins	±50	mA
P_D	Power Dissipation in Still Air, Plastic or Ceramic DIP†	750	mW
	SOIC Package†	500	
T_{stg}	Storage Temperature	−65 to +150	°C
T_L	Lead Temperature, 1 mm from Case for 10 Seconds		°C
	(Plastic DIP or SOIC Package)	260	
	(Ceramic DIP)	300	

*Maximum Ratings are those values beyond which damage to the device may occur. Functional operation should be restricted to the Recommended Operating Conditions.

†Derating — Plastic DIP: −10 mW/°C from 65° to 125°C
Ceramic DIP: −10 mW/°C from 100° to 125°C
SOIC Package: −7 mW/°C from 65° to 125°C
For high frequency or heavy load considerations, see Chapter 4.

This device contains protection circuitry to guard against damage due to high static voltages or electric fields. However, precautions must be taken to avoid applications of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range GND ≤ (V_{in} or V_{out}) ≤ V_{CC}.

Unused inputs must always be tied to an appropriate logic voltage level (e.g., either GND or V_{CC}). Unused outputs must be left open.

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter		Min	Max	Unit
V_{CC}	DC Supply Voltage (Referenced to GND)		2.0	6.0	V
V_{in}, V_{out}	DC Input Voltage, Output Voltage (Referenced to GND)		0	V_{CC}	V
T_A	Operating Temperature, All Package Types		−55	+125	°C
t_r, t_f	Input Rise and Fall Time	V_{CC} = 2.0 V	0	1000	ns
	(Figure 1)	V_{CC} = 4.5 V	0	500	
		V_{CC} = 6.0 V	0	400	

DC ELECTRICAL CHARACTERISTICS (Voltages Referenced to GND)

Symbol	Parameter	Test Conditions		V_{CC} V	Guaranteed Limit			Unit
					25°C to −55°C	≤85°C	≤125°C	
V_{IH}	Minimum High-Level Input Voltage	V_{out} = 0.1 V or V_{CC} − 0.1 V $\|I_{out}\|$ ≤ 20 μA		2.0	1.5	1.5	1.5	V
				4.5	3.15	3.15	3.15	
				6.0	4.2	4.2	4.2	
V_{IL}	Maximum Low-Level Input Voltage	V_{out} = 0.1 V or V_{CC} − 0.1 V $\|I_{out}\|$ ≤ 20 μA		2.0	0.3	0.3	0.3	V
				4.5	0.9	0.9	0.9	
				6.0	1.2	1.2	1.2	
V_{OH}	Minimum High-Level Output Voltage	V_{in} = V_{IH} or V_{IL} $\|I_{out}\|$ ≤ 20 μA		2.0	1.9	1.9	1.9	V
				4.5	4.4	4.4	4.4	
				6.0	5.9	5.9	5.9	
		V_{in} = V_{IH} or V_{IL}	$\|I_{out}\|$ ≤ 4.0 mA	4.5	3.98	3.84	3.70	
			$\|I_{out}\|$ ≤ 5.2 mA	6.0	5.48	5.34	5.20	
V_{OL}	Maximum Low-Level Output Voltage	V_{in} = V_{IH} or V_{IL} $\|I_{out}\|$ ≤ 20 μA		2.0	0.1	0.1	0.1	V
				4.5	0.1	0.1	0.1	
				6.0	0.1	0.1	0.1	
		V_{in} = V_{IH} or V_{IL}	$\|I_{out}\|$ ≤ 4.0 mA	4.5	0.26	0.33	0.40	
			$\|I_{out}\|$ ≤ 5.2 mA	6.0	0.26	0.33	0.40	
I_{in}	Maximum Input Leakage Current	V_{in} = V_{CC} or GND		6.0	±0.1	±1.0	±1.0	μA
I_{CC}	Maximum Quiescent Supply Current (per Package)	V_{in} = V_{CC} or GND I_{out} = 0 μA		6.0	8	80	160	μA

NOTE: Information on typical parametric values can be found in Chapter 4.

AC ELECTRICAL CHARACTERISTICS (C_L = 50 pF, input $t_r = t_f$ = 6 ns)

Symbol	Parameter	V_{CC} V	Guaranteed Limit			Unit
			25°C to −55°C	≤85°C	≤125°C	
t_{PLH}, t_{PHL}	Maximum Propagation Delay, Chip Select to Output Y (Figures 1 and 5)	2.0	175	220	265	ns
		4.5	35	44	53	
		6.0	30	37	45	
t_{PLH}	Maximum Propagation Delay, Input A to Output Y (Figures 2 and 5)	2.0	230	290	345	ns
		4.5	46	58	69	
		6.0	39	49	59	
t_{PHL}		2.0	175	220	265	
		4.5	35	44	53	
		6.0	30	37	45	
t_{PLH}	Maximum Propagation Delay, Latch Enable to Output Y (Figures 3 and 5)	2.0	230	290	345	ns
		4.5	46	58	69	
		6.0	39	49	59	
t_{PHL}		2.0	175	220	265	
		4.5	35	44	53	
		6.0	30	37	45	
t_{TLH}, t_{THL}	Maximum Output Transition Time, Any Output (Figures 1 and 5)	2.0	75	95	110	ns
		4.5	15	19	22	
		6.0	13	16	19	
C_{in}	Maximum Input Capacitance	—	10	10	10	pF

NOTES:
1. For propagation delays with loads other than 50 pF, see Chapter 4.
2. Information on typical parametric values can be found in Chapter 4.

C_{PD}	Power Dissipation Capacitance (Per Package) Used to determine the no-load dynamic power consumption: $P_D = C_{PD} V_{CC}^2 f + I_{CC} V_{CC}$ For load considerations, see Chapter 4.	Typical @ 25°C, V_{CC} = 5.0 V 70	pF

TIMING REQUIREMENTS (Input $t_r = t_f$ = 6 ns)

Symbol	Parameter	V_{CC} V	Guaranteed Limit			Unit
			25°C to −55°C	≤85°C	≤125°C	
t_{su}	Minimum Setup Time, Input A to Latch Enable (Figure 4)	2.0	100	125	150	ns
		4.5	20	25	30	
		6.0	17	21	26	
t_h	Minimum Hold Time, Latch Enable to Input A (Figure 4)	2.0	5	5	5	ns
		4.5	5	5	5	
		6.0	5	5	5	
t_w	Minimum Pulse Width, Latch Enable (Figure 3)	2.0	80	100	120	ns
		4.5	16	20	24	
		6.0	14	17	20	
t_r, t_f	Maximum Input Rise and Fall Times (Figure 1)	2.0	1000	1000	1000	ns
		4.5	500	500	500	
		6.0	400	400	400	

NOTE: Information on typical parametric values can be found in Chapter 4.

SWITCHING WAVEFORMS

FIGURE 1

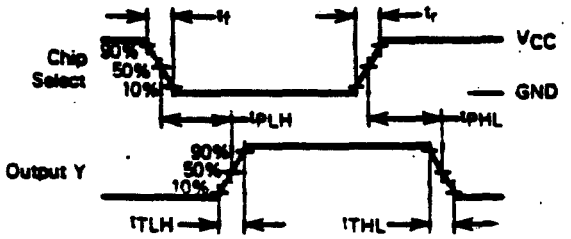


FIGURE 2

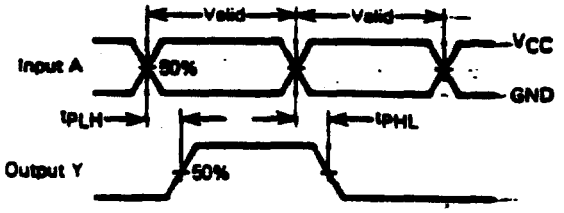


FIGURE 3

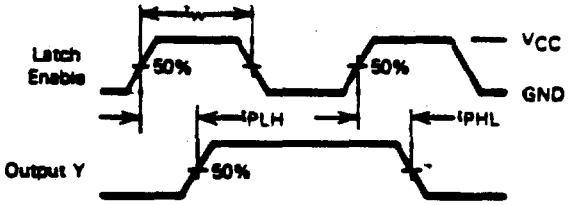


FIGURE 4

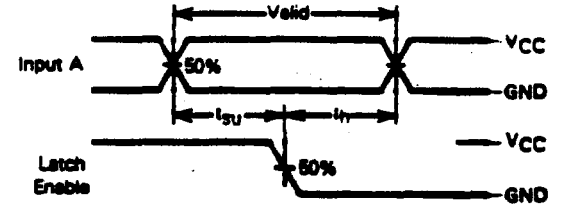
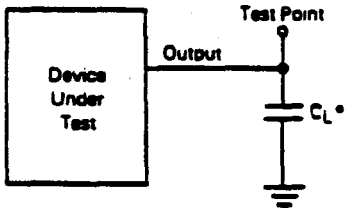


FIGURE 5 — TEST CIRCUIT



* Includes all probe and jig capacitance.

FUNCTION TABLE

Latch Enable	Chip Select	Address Inputs				Selected Output (High)
		A3	A2	A1	A0	
H	L	L	L	L	L	Y0
H	L	L	L	L	H	Y1
H	L	L	L	H	L	Y2
H	L	L	L	H	H	Y3
H	L	L	H	L	L	Y4
H	L	L	H	L	H	Y5
H	L	L	H	H	L	Y6
H	L	L	H	H	H	Y7
H	L	H	L	L	L	Y8
H	L	H	L	L	H	Y9
H	L	H	L	H	L	Y10
H	L	H	L	H	H	Y11
H	L	H	H	L	L	Y12
H	L	H	H	L	H	Y13
H	L	H	H	H	L	Y14
H	L	H	H	H	H	Y15
X	H	X	X	X	X	All Outputs = L
L	L	X	X	X	X	Latched Data

PIN DESCRIPTIONS

ADDRESS INPUTS

A0, A1, A2, A3 (PINS 2, 3, 21, 22) — Address Inputs. These inputs are decoded to produce a high level on one of 16 outputs. The inputs are arranged such that A3 is the most-significant bit and A0 is the least-significant bit. The decimal equivalent of the binary input address indicates which of the 16 data outputs, Y0-Y15, is selected.

OUTPUTS

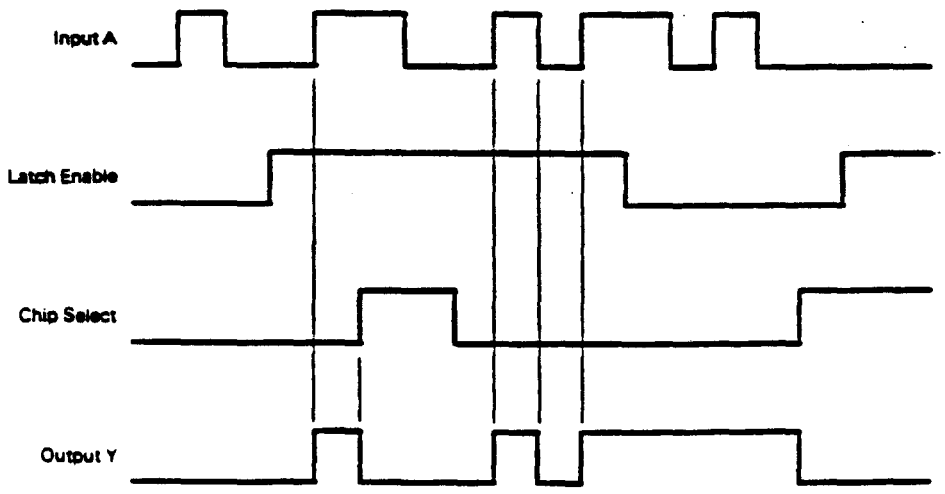
Y0-Y15 (PINS 11, 9, 10, 8, 7, 6, 5, 4, 18, 17, 20, 19, 14, 13, 16, 15) — Active-High Outputs. These outputs produce a high level when selected (Latch Enable = H, Chip Select = L) and are at a low level when not selected.

CONTROL INPUTS

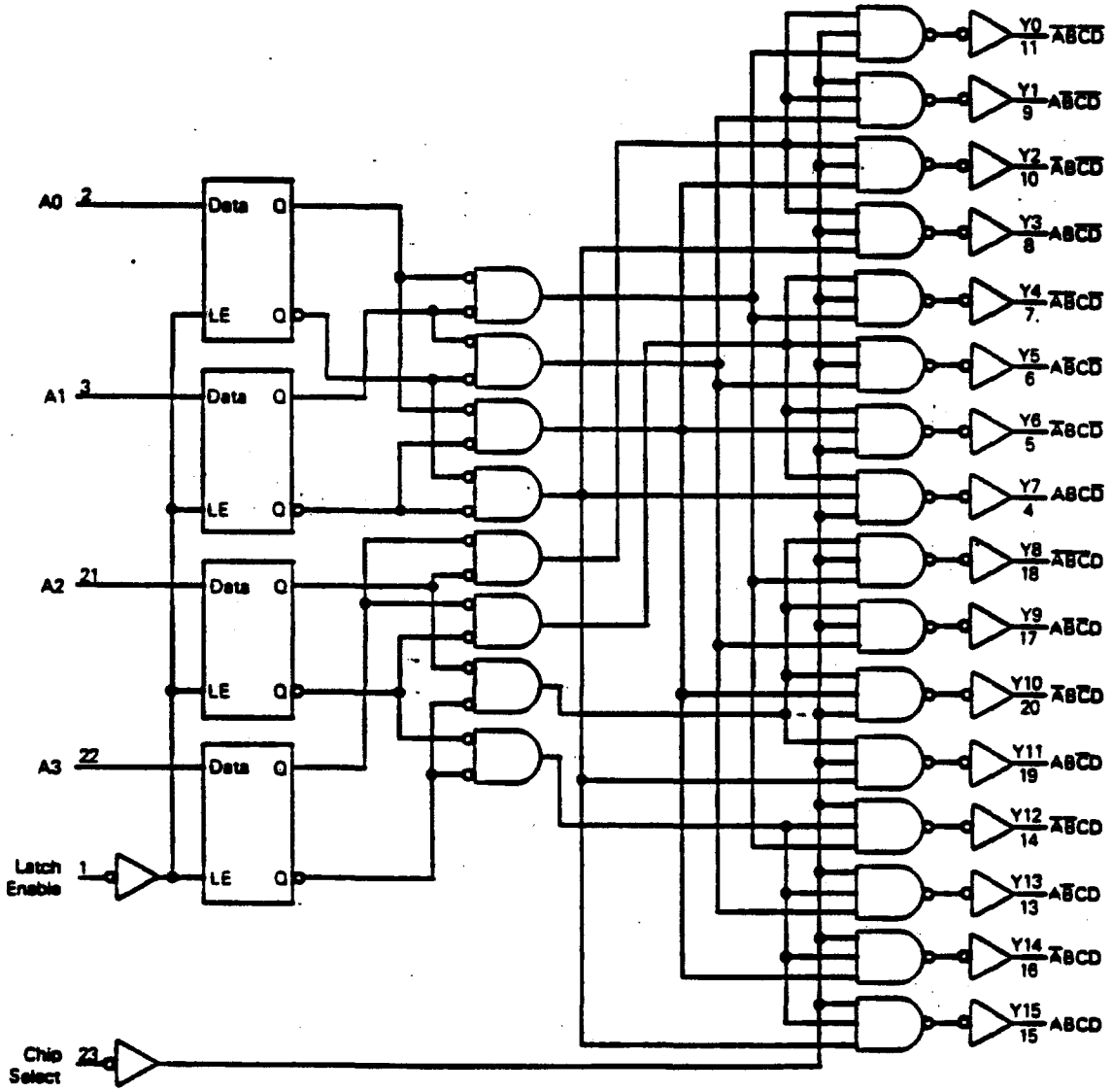
LATCH ENABLE (PIN 1) — Latch Enable Input. A low level on this input stores the data on the Address data inputs in the 4-bit latch. A high level on the Latch Enable input makes the latch transparent and allows the outputs to follow the inputs. Note that the data is latched only while the Latch Enable input is at a low level.

CHIP SELECT (PIN 23) — Chip Select Input. A high on this input produces a low level on all outputs, regardless of what appears at the address or Latch Enable inputs. A low level on the Chip Select input allows the selected output to produce a high level.

TIMING DIAGRAM



EXPANDED LOGIC DIAGRAM



MOTOROLA

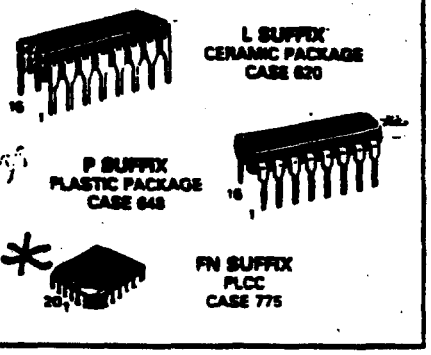
MC10H188

HEX BUFFER WITH ENABLE

The MC10H188 is a high-speed Hex Buffer with a common Enable input. When Enable is in the high-state, all outputs are in the low-state. When Enable is in the low-state, the outputs take the same state as the inputs.

This MECL 10KH part is a functional/pinout duplication of the standard MECL 10K family part, with 100% improvement in propagation delay and no increase in power-supply current.

- Propagation Delay, 1.3 ns Typical Data-to-Output
- Power Dissipation 180 mW Typ/Pkg (No Load)
- Improved Noise Margin 150 mV (Over Operating Voltage and Temperature Range)
- Voltage Compensated
- MECL 10K-Compatible



MAXIMUM RATINGS

Characteristic	Symbol	Rating	Unit
Power Supply (V_{CC} = 0)	V_{EE}	-8.0 to 0	Vdc
Input Voltage (V_{CC} = 0)	V_I	0 to V_{EE}	Vdc
Output Current — Continuous — Surge	I_{out}	50 100	mA
Operating Temperature Range	T_A	0–+75	°C
Storage Temperature Range — Plastic — Ceramic	T_{stg}	-55 to +150 -55 to +165	°C °C

ELECTRICAL CHARACTERISTICS (V_{EE} = -5.2 V ±5%) (See Note)

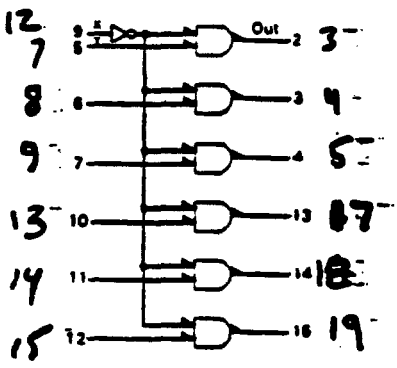
Characteristic	Symbol	0° Min	0° Max	25° Min	25° Max	75° Min	75° Max	Unit
Power Supply Current	I_E	—	46	—	42	—	46	mA
Input Current High	I_{inH}	—	495	—	310	—	310	µA
Input Current Low	I_{inL}	0.5	—	0.5	—	0.3	—	µA
High Output Voltage	V_{OH}	-1.02	-0.84	-0.98	-0.81	-0.92	-0.735	Vdc
Low Output Voltage	V_{OL}	-1.95	-1.63	-1.95	-1.63	-1.95	-1.60	Vdc
High Input Voltage	V_{IH}	-1.17	-0.84	-1.13	-0.81	-1.07	-0.735	Vdc
Low Input Voltage	V_{IL}	-1.95	-1.48	-1.95	-1.48	-1.95	-1.45	Vdc

AC PARAMETERS

Propagation Delay Enable Data	t_{pd}	 0.7 0.7	 2.2 1.9	 0.7 0.7	 2.2 1.9	 0.7 0.7	 2.2 1.9	ns
Rise Time	t_r	0.7	2.4	0.7	2.4	0.7	2.4	ns
Fall Time	t_f	0.7	2.4	0.7	2.4	0.7	2.4	ns

NOTE:
Each MECL 10KH series circuit has been designed to meet the dc specifications shown in the test table, after thermal equilibrium has been established. The circuit is in a test socket or mounted on a printed circuit board and transverse air flow greater than 500 lfpm is maintained. Outputs are terminated through a 50 ohm resistor to -2.0 volts.

LOGIC DIAGRAM

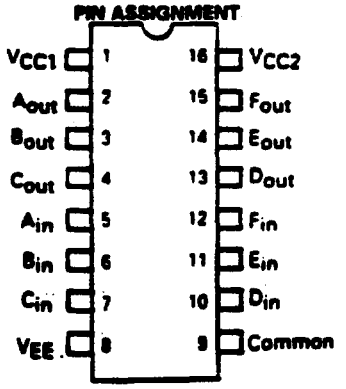


TRUTH TABLE

Inputs		Output
X	Y	OUT
L	L	L
L	H	H
H	L	L
H	H	L

V_{CC1} = Pin 1 2
V_{CC2} = Pin 16 20
V_{EE} = Pin 8 10

DIP PIN ASSIGNMENT



Pin assignment is for Dual-in-line Package.
For PLCC pin assignment, see tables on page 1-31.

(10)

REVISED DECEMBER 1983

- **D-C Triggered from Active-High or Active-Low Gated Logic Inputs**
- **Retriggerable for Very Long Output Pulses, Up to 100% Duty Cycle**
- **Overriding Clear Terminates Output Pulse**
- **'122, 'L122, 'LS122 Have Internal Timing Resistors**

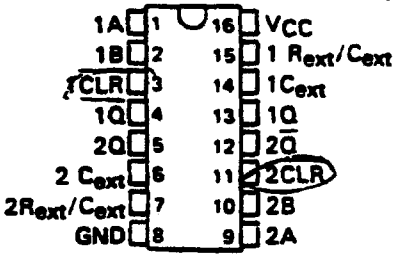
description

These d-c triggered multivibrators feature output pulse width control by three methods. The basic pulse time is programmed by selection of external resistance and capacitance values (see typical application data). The '122, 'L122, and 'LS122 have internal timing resistors that allow the circuits to be used with only an external capacitor, if so desired. Once triggered, the basic pulse width may be extended by retriggering the gated low-level-active (A) or high-level-active (B) inputs, or be reduced by use of the overriding clear. Figure 1 illustrates pulse control by retriggering and early clear.

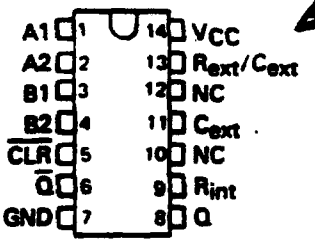
The 'LS122 and 'LS123 are provided enough Schmitt hysteresis to ensure jitter-free triggering from the B input with transition rates as slow as 0.1 millivolt per nanosecond.

The R_{int} is nominally 10 k ohms for '122, 'LS122, and is nominally 20 k ohms for 'L122.

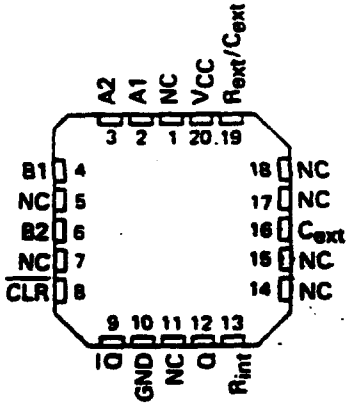
SN54122, SN54LS122 . . . J OR W PACKAGE
SN54L122 . . . J PACKAGE
SN74122 . . . J OR N PACKAGE
SN74LS122 . . . D, J OR N PACKAGE
(TOP VIEW) (SEE NOTES 1 THRU 4)



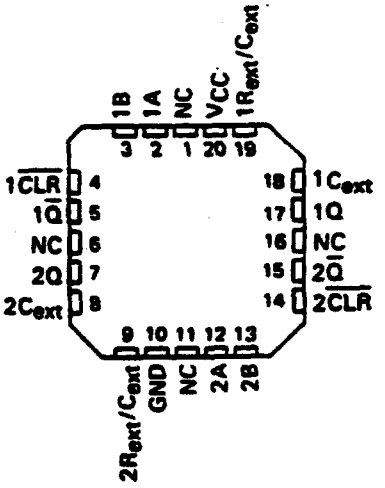
SN54123, SN54130, SN54LS123 . . . J OR W PACKAGE
SN54L123 . . . J PACKAGE
SN74123, SN74130 . . . J OR N PACKAGE
SN74LS123 . . . D, J OR N PACKAGE
(TOP VIEW) (SEE NOTES 1 THRU 4)



SN54LS122 . . . FK PACKAGE
SN74LS122 . . . FN PACKAGE
(TOP VIEW) (SEE NOTES 1 THRU 4)



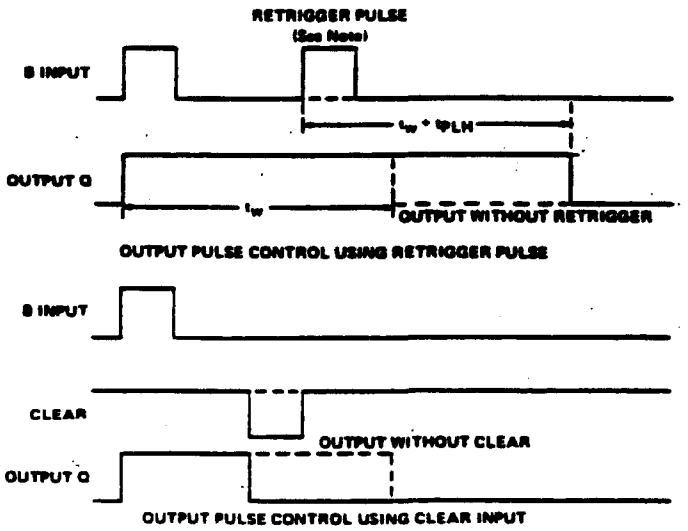
SN54LS123 . . . FK PACKAGE
SN74LS123 . . . FN PACKAGE
(TOP VIEW) (SEE NOTES 1 THRU 4)



NC - No internal connection

NOTES: 1. An external timing capacitor may be connected between C_{ext} and R_{ext}/C_{ext} (positive).
2. To use the internal timing resistor of '122, 'L122, or 'LS122, connect R_{int} to V_{CC}.
3. For improved pulse width accuracy and repeatability, connect an external resistor between R_{ext}/C_{ext} and V_{CC} with R_{int} open-circuited.
4. To obtain variable pulse widths, connect an external variable resistance between R_{int} or R_{ext}/C_{ext} and V_{CC}.

escription (continued)



NOTE: Retrigger pulses starting before 0.22 C_{ext} (in picofrads) nanoseconds after the initial trigger pulse will be ignored and the output pulse will remain unchanged.

FIGURE 1–TYPICAL INPUT/OUTPUT PULSES

'122, 'L122, 'LS122
FUNCTION TABLE

INPUTS					OUTPUTS	
CLEAR	A1	A2	B1	B2	Q	$\bar{Q}$
L	X	X	X	X	L	H
X	H	H	X	X	L†	H†
X	X	X	L	X	L†	H†
X	X	X	X	L	L†	H†
H	L	X	↑	H	⊓	⊔
H	L	X	H	↑	⊓	⊔
H	X	L	↑	H	⊓	⊔
H	X	L	H	↑	⊓	⊔
H	H	↓	H	H	⊓	⊔
H	↓	↓	H	H	⊓	⊔
H	↓	H	H	H	⊓	⊔
↑	L	X	H	H	⊓	⊔
↑	X	L	H	H	⊓	⊔

See explanation of function tables on page

† These lines of the functional tables assume that the indicated steady-state conditions at the A and B inputs have been set up long enough to complete any pulse started before the set up.

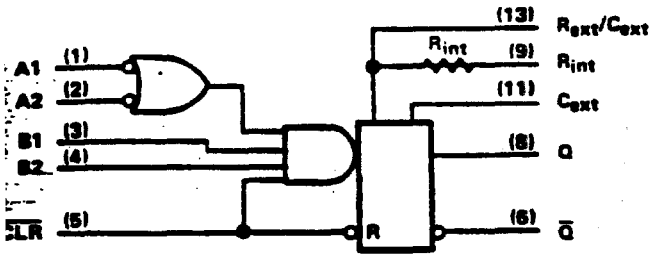
'123, '130, 'L123, 'LS123
FUNCTION TABLE

INPUTS			OUTPUTS	
CLEAR	A	B	Q	$\bar{Q}$
L	X	X	L	H
X	H	X	L†	H†
X	X	L	L†	H†
H	L	↑	⊓	⊔
H	↓	H	⊓	⊔
↑	L	H	⊓	⊔

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

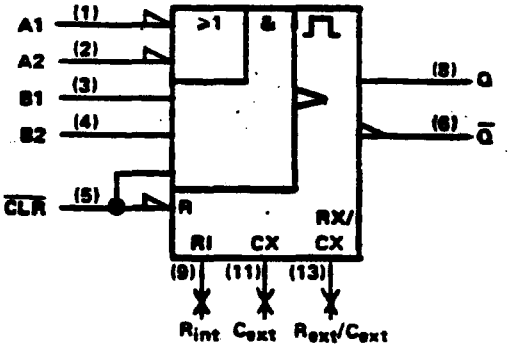
logic diagram

'122, 'L122, 'LS122



logic symbol

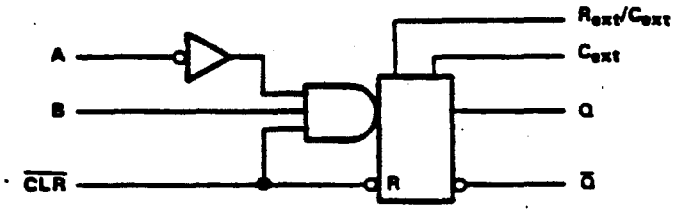
'122, 'L122, 'LS122



R_{int} is nominally 10 k ohms for '122, 'LS122, and 20 k ohms for 'L122.

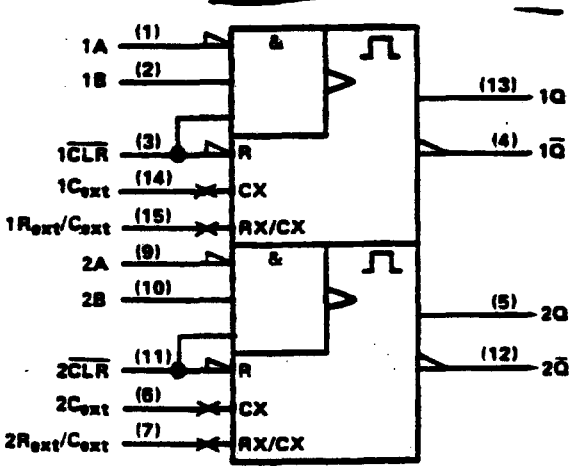
logic diagram (each multivibrator)

'123, '130, 'L123, 'LS123



logic symbol

'123, '130, 'L123, 'LS123

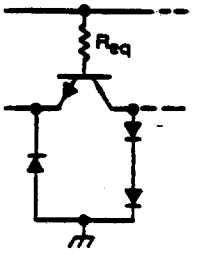


Pin numbers shown on logic notation are for D, J or N packages.

schematics of inputs and outputs

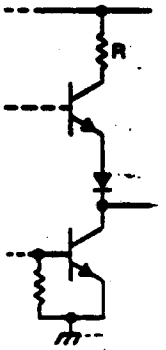
'122, '123, '130, 'L122, 'L123 CIRCUITS

EQUIVALENT OF EACH INPUT



Clear inputs of '122, '123, '130: R_{eq} = 2 kΩ NOM
Other inputs of '122, '123, '130: R_{eq} = 4 kΩ NOM
Clear inputs of 'L122, 'L123: R_{eq} = 4 kΩ NOM
Other inputs of 'L122, 'L123: R_{eq} = 8 kΩ NOM

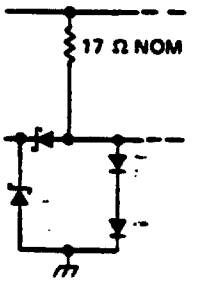
TYPICAL OF ALL OUTPUTS



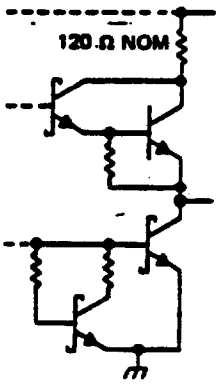
'122, '123, '130: R = 100 Ω NOM
'L122, 'L123: R = 200 Ω NOM

'LS122, 'LS123 CIRCUITS

EQUIVALENT OF EACH INPUT



TYPICAL OF ALL OUTPUTS



TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

recommended operating conditions

	SN54'			SN74'			UNIT
	MIN	NOM	MAX	MIN	NOM	MAX	
Supply voltage, V_{CC}	4.5	5	5.5	4.75	5	5.25	V
High-level output current, I_{OH}			−800			−800	μA
Low-level output current, I_{OL}			16			16	mA
Pulse width, t_w	40			40			ns
External timing resistance, R_{ext}	5		25	5		50	kΩ
External capacitance, C_{ext}	No restriction			No restriction			
Wiring capacitance at R_{ext}/C_{ext} terminal			50			50	pF
Operating free-air temperature, T_A	−55		125	0		70	°C

electrical characteristics over recommended free-air operating temperature range (unless otherwise noted)

PARAMETER			TEST CONDITIONS†	'122			'123, '130			UNIT
				MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IH}	High-level input voltage			2			2			V
V_{IL}	Low-level input voltage					0.8			0.8	V
V_{IK}	Input clamp voltage		V_{CC} = MIN, I_I = −12 mA			−1.5			−1.5	V
V_{OH}	High-level output voltage		V_{CC} = MIN, I_{OH} = −800 μA, See Note 1	2.4	3.4		2.4	3.4		V
V_{OL}	Low-level output voltage		V_{CC} = MIN, I_{OL} = 16 mA, See Note 1		0.2	0.4		0.2	0.4	V
I_I	Input current at maximum input voltage		V_{CC} = MAX, V_I = 5.5 V			1			1	mA
I_{IH}	High-level input current	Data inputs	V_{CC} = MAX, V_I = 2.4 V			40			40	μA
		Clear input				80			80	
I_{IL}	Low-level input current	Data inputs	V_{CC} = MAX, V_I = 0.4 V			−1.6			−1.6	mA
		Clear input				−3.2			−3.2	
I_{OS}	Short-circuit output current♦		V_{CC} = MAX, See Note 5	−10		−40	−10		−40	mA
I_{CC}	Supply current (quiescent or triggered)		V_{CC} = MAX, See Notes 6 and 7		23	36		46	66	mA

† For conditions shown as MIN or MAX, use the value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
♦ Not more than one output should be shorted at a time.

NOTES: 5. Ground C_{ext} to measure V_{OH} at Q, V_{OL} at $\overline{Q}$, or I_{OS} at Q. C_{ext} is open to measure V_{OH} at $\overline{Q}$, V_{OL} at Q, or I_{OS} at $\overline{Q}$.
6. Quiescent I_{CC} is measured (after clearing) with 2.4 V applied to all clear and A inputs, B inputs grounded, all outputs open, C_{ext} = 0.02 μF, and R_{ext} = 25 kΩ. R_{int} of '122 is open.
7. I_{CC} is measured in the triggered state with 2.4 V applied to all clear and B inputs, A inputs grounded, all outputs open, C_{ext} = 0.02 μF, and R_{ext} = 25 kΩ. R_{int} of '122 is open.

switching characteristics, V_{CC} = 5 V, T_A = 25°C, see note 8

PARAMETER¶	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	'122, '130			'123			UNIT
				MIN	TYP	MAX	MIN	TYP	MAX	
t_{PLH}	A	Q	C_{ext} = 0, C_L = 15 pF, R_{ext} = 5 kΩ, R_L = 400 Ω		22	33		22	33	ns
	B				19	28		19	28	
t_{PHL}	A	$\overline{Q}$			30	40		30	40	ns
	B				27	36		27	36	
t_{PHL}	Clear	Q			18	27		18	27	ns
t_{PLH}		$\overline{Q}$			30	40		30	40	
t_{wQ} (min)	A or B	Q			45	65		45	65	ns
t_{wQ}	A or B	Q	C_{ext} = 1000 pF, C_L = 15 pF, R_{ext} = 10 kΩ, R_L = 400 Ω	3.08	3.42	3.76	2.76	3.03	3.37	μs

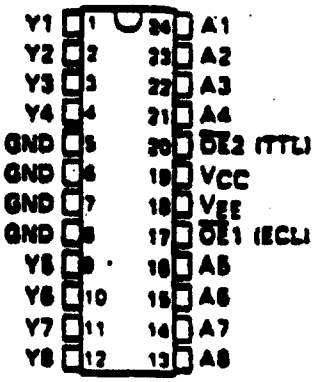
¶ t_{PLH} = propagation delay time, low-to-high-level output
t_{PHL} = propagation delay time, high-to-low-level output
t_{wQ} = width of pulse at output Q

NOTE 8: See General Information Section for load circuits and voltage waveforms.

D3136, AUGUST 1988—REVISED DECEMBER 1991

- 10KH Compatible
- ECL and TTL Control Inputs
- P-N-P Inputs Reduce DC Loading
- New Flow-Through Architecture to Optimize PCB Layout
- Center Pin V_{CC}, V_{EE} and GND Configurations to Minimize High-Speed Switching Noise
- ESD Protection Exceeds 2000 V, MIL-STD-883C Method 3015
- Package Options Include "Small Outline" Packages and Standard Plastic 300-mil DIPs

DW OR NT PACKAGE
(TOP VIEW)



description

These octal TTL-to-ECL translators are designed to provide an efficient translation function between a TTL signal environment and a 10KH ECL signal environment. The designer has a choice of inverting ('5542) or true ('5543) outputs. Two pins, $\overline{OE}1$ and $\overline{OE}2$, are provided for output enable control. These control inputs are negative ANDed together, with $\overline{OE}1$ being ECL compatible and $\overline{OE}2$ being TTL compatible. This offers the choice of controlling the outputs of the device from either a TTL or ECL signal environment. The outputs, when disabled, go to a normal ECL logic low level.

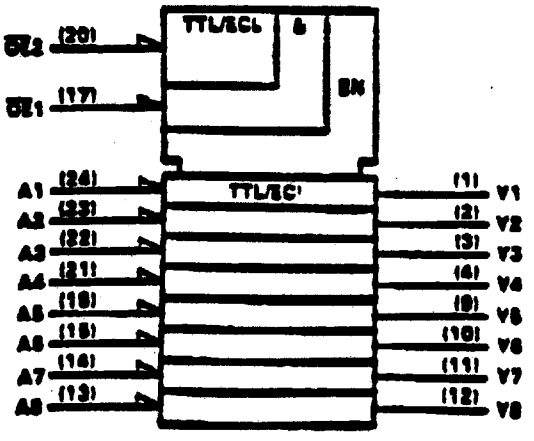
The SN10KHT5542 and SN10KHT5543 are characterized for operation from 0°C to 75°C.

FUNCTION TABLE

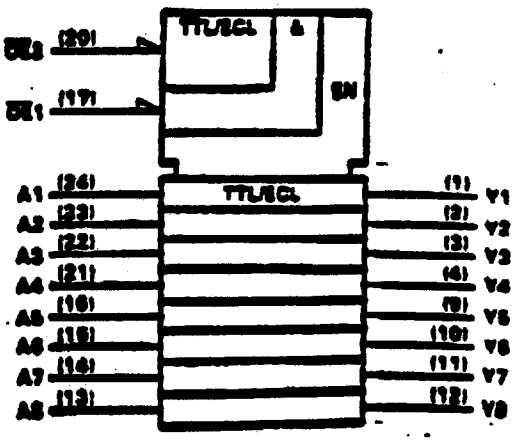
OUTPUT CONTROL		DATA INPUT	OUTPUT	
$\overline{OE}1$	$\overline{OE}2$	A	'5542	'5543
H	X	X	L	L
X	H	X	L	L
L	L	L	H	L
L	L	H	L	H

logic symbols†

SN10KHT5542



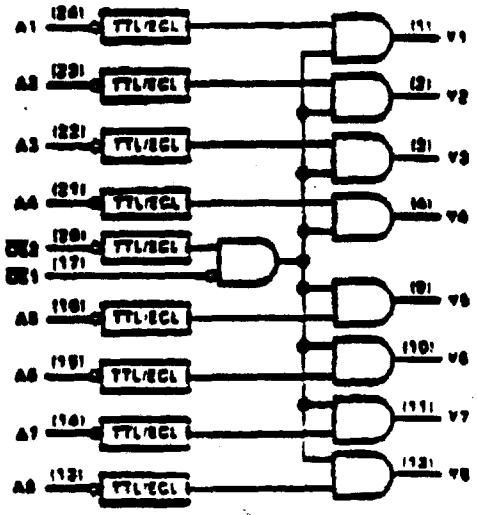
SN10KHT5543



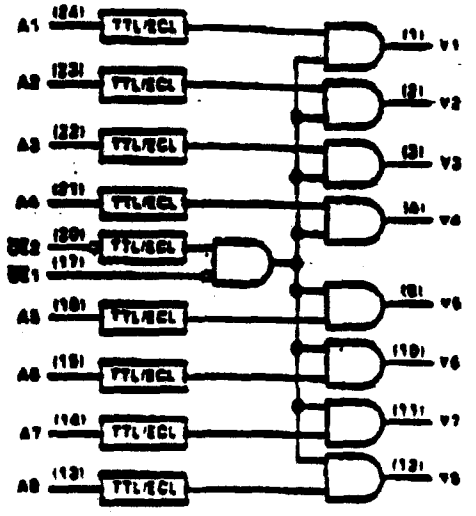
†These symbols are in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.

logic diagrams (positive logic)

SN10KHT5542



SN10KHT5543



absolute maximum ratings over operating ambient temperature range (unless otherwise noted

Supply voltage, V_{CC} . . . . . −0.5 V to
Supply voltage, V_{EE} . . . . . −8 V to
Input voltage (TTL) (See Note 1) . . . . . −1.2 V to
Input voltage (ECL) . . . . . V_{EE} to
Input current (TTL) . . . . . −30 mA to 5
Operating ambient temperature range . . . . . 0°C
Storage temperature range . . . . . −65°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ra only and functional operation of the device at these or any other conditions beyond those indicated under "recommended oper conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 1: The input voltage ratings may be exceeded provided the input current ratings are observed.

recommended operating conditions (see Note 2)

			MIN	NOM	MAX	UI
V_{CC}	TTL supply voltage		4.5	5.0	5.5	V
V_{EE}	ECL supply voltage		−4.94	−5.2	−5.46	V
V_{IH}	TTL high-level input voltage		2			V
V_{IH}	ECL high-level input voltage‡	0°C	−1170		−840	m
		25°C	−1130		−810	
		75°C	−1070		−735	
V_{IL}	TTL low-level input voltage				0.8	V
V_{IL}	ECL low-level input voltage‡	0°C	−1950		−1480	m
		25°C	−1950		−1480	
		75°C	−1950		−1450	
I_{IK}	TTL input clamp current				−18	m
T_A	Operating ambient temperature (see Note 3)		0		75	°C

‡ The algebraic convention, in which the least positive (most negative) value is designated minimum, is used in this data sheet for voltage levels and temperature only.

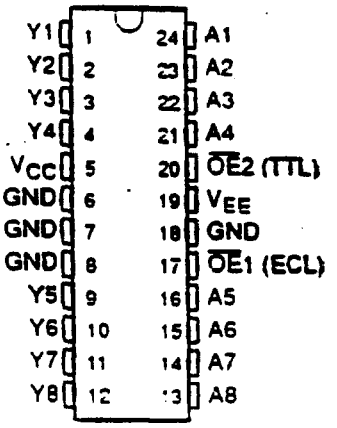
NOTES: 2. If unused, OE1 should be tied directly to −2 V.
3: Each 10KH series circuit has been designed to meet the dc specifications shown in the electrical characteristics table thermal equilibrium has been established. The circuit is in a test socket or mounted on a printed circuit board, and transv air flow greater than 500 linear ft/min is maintained.

TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

DXXXX, JULY 1989 COMPOSED JUL 20, 1989 AT 10:32:14

- 10KH Compatible
- ECL and TTL Control Inputs
- New Flow–Through Architecture to Optimize PCB Layout
- Center Pin V_{CC}, V_{EE} and GND Configurations to Minimize High Speed Switching Noise
- ESD Protection Exceeds 2000 V, Mil Standard 883C Method 3015
- Package Options Include "Small Outline" Packages, Standard Plastic and Ceramic 300–mil DIPs

SN10KHT5540...DW or NT Package
SN10KHT5541...DW or NT Package
(TOP VIEW)



description

These octal ECL–to–TTL translators are designed to provide the efficient translation function between a 10KH ECL signal environment to a TTL signal environment. These devices are designed specifically to improve the performance and density of ECL–to–TTL CPU/Bus oriented functions such as memory address drivers, clock drivers, and bus oriented receivers and transmitters.

The designer has a choice of inverting ('5540) or true ('5541) outputs. Two pins $\overline{OE}1$ and $\overline{OE}2$ are allowed for output enable control. These control inputs are or'ed together with $\overline{OE}1$ being TTL compatible and $\overline{OE}2$ being ECL compatible. This offers the choice of controlling the outputs of the device from either a TTL or ECL signal environment.

The SN10KHT5540 and SN10KHT5541 are characterized for operation from 0°C to 75°C.

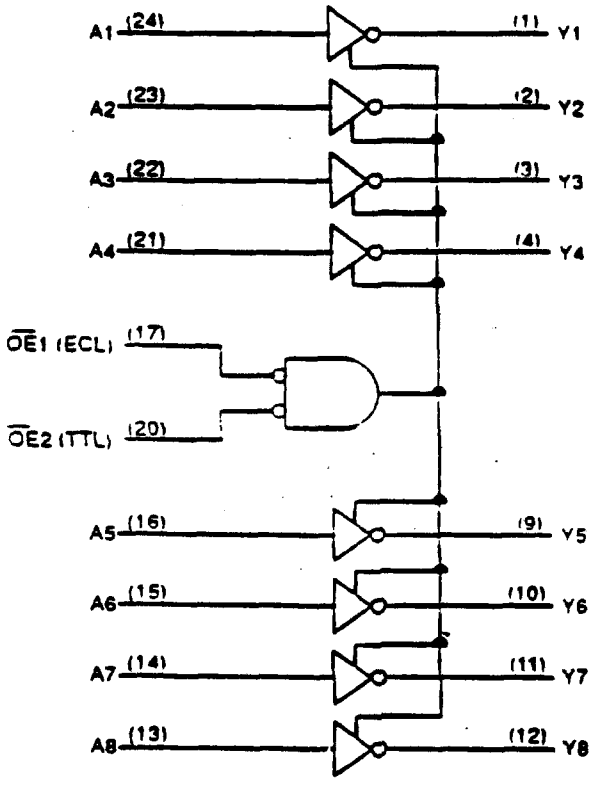
PRELIMINARY

FUNCTION TABLE

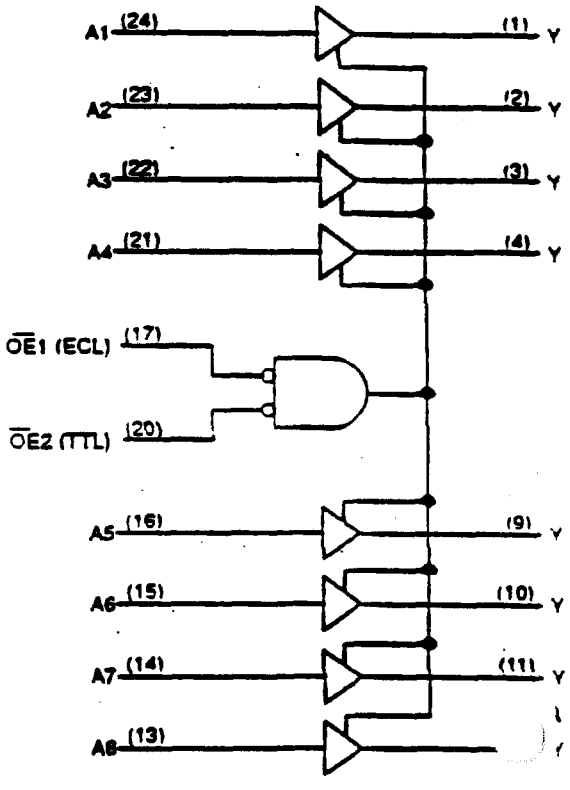
OUTPUT CONTROL		DATA INPUT	OUTPUT	
$\overline{OE}1$	$\overline{OE}2$	A	'5540	'5541
X	H	X	Z	Z
H	X	X	Z	Z
L	L	L	H	L
L	L	H	L	H

logic diagram (positive logic)

SN10KHT5540



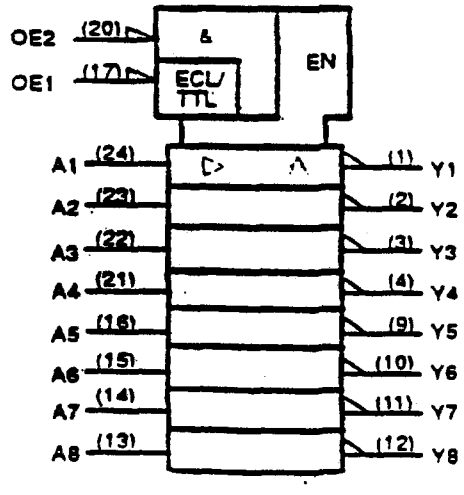
SN10KHT5541



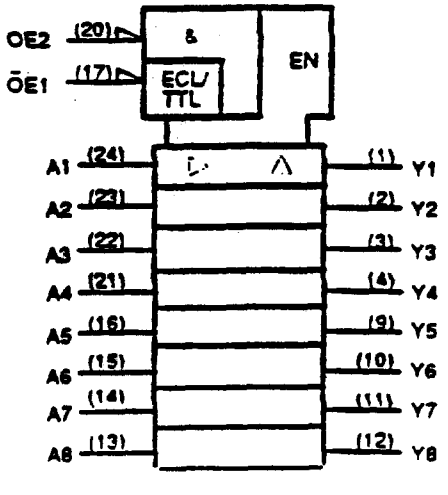
logic symbols*

PRELIMINARY

SN10KHT5540



SN10KHT5541



*These symbols are in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.

TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

EPM5016 to EPM5192

High-Speed, High-Density MAX EPLDs

October 1990, ver. 1 — Data Sheet

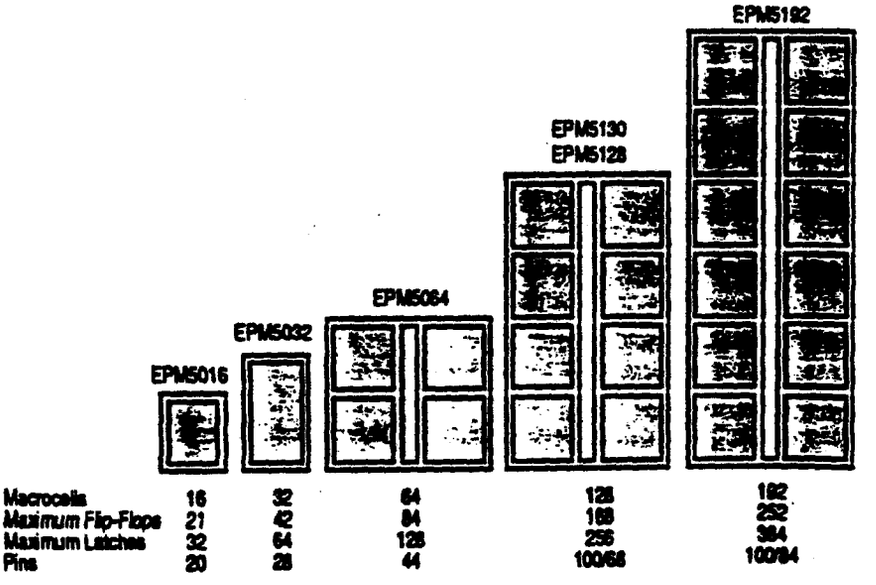
Features

- Complete family of CMOS EPLDs solves design tasks ranging from fast 20-pin address decoders to 100-pin LSI custom peripherals.
- The advanced MAX architecture combines the speed, ease of use, and familiarity of PAL devices with the density of programmable gate arrays.
- EP5000-series EPLDs provide 15-ns combinatorial delays, counter frequencies up to 100 MHz, pipelined data rates of 100 MHz, and high-complexity designs with true system clock rates up to 66 MHz.
- Available in a wide variety of packages, including DIP, SOIC, J-Lead, PGA, and QFP formats in windowed ceramic and plastic one-time-programmable versions.
- MAX+PLUS PC- and workstation-based development tools compile large designs in minutes.
- Industry-standard EDIF interfaces to workstation and third-party CAE tools are available.

3

Figure 1 shows the EPM5000-series modular architecture.

Figure 1. EPM5000-Series Modular Architecture



14

Family Highlights

- **Multiple Array MatriX (MAX) architecture solves speed, density, and design flexibility problems**
  - Advanced macrocell array provides registered, combinatorial, or flow-through latch operation.
  - Expander product-term array automatically provides additional combinatorial or registered logic.
  - Decoupled I/O block with dual feedback on I/O pins allows flexible pin utilization.
  - Programmable Interconnect Array provides automatic 100% routing in devices with multiple LABs.
  - Each macrocell supports synchronous or asynchronous operation of every macrocell, using single or multiple clocks per device.
- **EPM5000-Series Performance**
  - Pipelined data rates up to 100 MHz
  - Counters as fast as 100 MHz
  - t_{PD} performance from 15 ns to 25 ns
  - Advanced 0.8-micron CMOS EPROM technology
- **EPM5000-Series Logic Density**
  - 16- to 192-macrocell devices
  - 20- to 100-pin packages
  - 32 to 384 flip-flops and latches
  - More than 32 product terms on a single macrocell
  - Product-term expansion on any data or control path
- **MAX+PLUS Design Tools**
  - Design entry via unified, hierarchical schematic capture and Altera Hardware Description Language (AHDL)
  - Fast, automatic design processing with logic synthesis
  - Automatic device fitting, no hand editing needed
  - Hardware and software design verification tools
  - Compiles a 16-bit counter in 34 seconds on a 16-MHz 386 computer
- **EDIF interfaces to MAX+PLUS provide paths to Dazix, Valid Logic Systems, Mentor Graphics, and other workstation-based CAE tools.**

General Description

EPM5000-series Erasable Programmable Logic Devices (EPLDs) represent a revolutionary step in programmable logic: they combine innovative architecture and state-of-the-art process to offer optimum performance, logic density, flexibility, and the highest speeds and densities available in general-purpose reprogrammable logic. These EPLDs are high-speed, high-density replacements for SSI and MSI TTL and CMOS packages and conventional PLDs. For example, an EPM5192 replaces over 100 7400-series SSI and MSI TTL and CMOS packages, integrating complete subsystems into a single package, saving board area, and reducing power consumption.

These MAX EPLDs range in density from 16 to 192 macrocells. They are divided into two groups: higher-speed MAX EPLDs (EPM5016 and EPM5032) and higher-density MAX EPLDs (EPM5064, EPM5128, EPM5130 and EPM5192). The higher-speed MAX EPLDs achieve system c[illegible] frequencies of 66 MHz, and are capable of counter frequencies of 100 M[illegible]

Logic Array Blocks The EPM5016 and EPM5032 MAX EPLDs have a single Logic Array Block (LAB). The EPM5064, EPM5128, EPM5130, and EPM5192 MAX EPLDs contain multiple LABs. Each LAB contains a macrocell array, an expander product-term array, and a decoupled I/O block. Expander product terms (expanders) are unallocated, inverted product terms that can be used and shared by all macrocells in the LAB to create combinatorial and registered logic. Thus, expressions requiring up to 66 product terms can be implemented in a single macrocell. Signals in the higher-density devices are routed between multiple LABs by a Programmable Interconnect Array (PIA) that ensures 100% routability. This multiple array architecture enables EPM5000-series EPLDs to offer the speed of smaller arrays with the integration density of larger arrays.

Modular Architecture The modular architecture of MAX EPLDs provides integration solutions over a wide range of logic densities. Migration from one type of device to another is easy. For example, the EPM5128 and EPM5130 EPLDs have the same logic capacity, but have packages optimized to handle different I/O requirements. Over the entire family, a wide range of packaging options for both through-hole and surface-mount applications is available. Plastic one-time-programmable (OTP) packages are available for economical volume production.

3

Logic Design Entry Logic designs are created and programmed into EPM5000-series EPLDs with the MAX+PLUS Development System. MAX+PLUS is a complete CAE system offering hierarchical design entry tools, automatic design compilation and fitting, timing simulation, and device programming. The MAX+PLUS Compiler features advanced logic synthesis algorithms, allowing designs to be entered in a variety of high-level formats while ensuring the most efficient use of EPLD resources. The combination of a flexible architecture and advanced CAE tools ensures rapid design cycles so that a design may go from conception to completion in single day. Interfaces to third-party tools are also available to allow design entry and logic simulation on a variety of workstation platforms.

Functional Description

EPM5000-series EPLDs use CMOS EPROM cells to configure logic functions within the devices. MAX architecture is user-configurable to accommodate a variety of independent logic functions, and the EPLDs can be erased for quick and efficient iterations during design development and debug cycles.

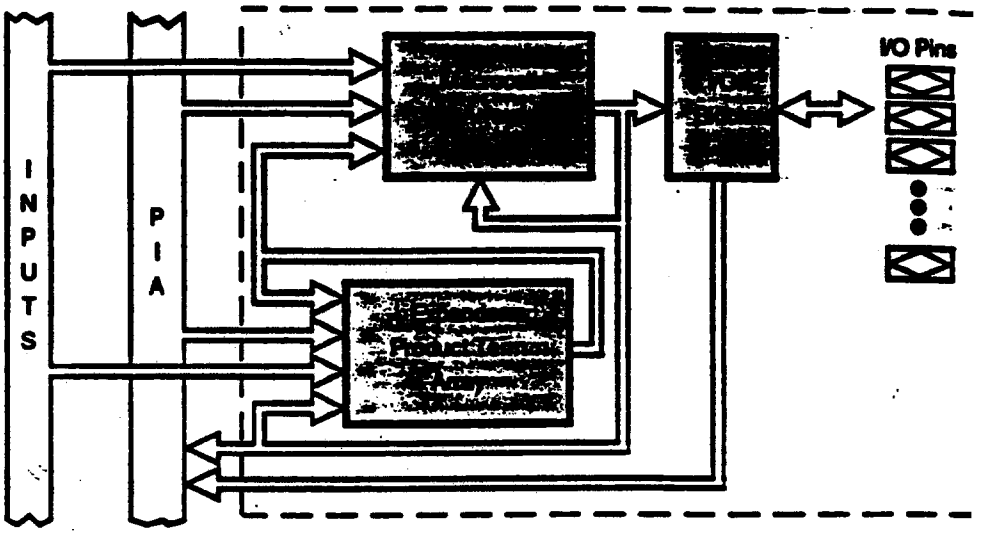
Logic Array Block

EPM5000-series EPLDs contain from 1 to 12 Logic Array Blocks. Each LAB, shown in Figure 2, consists of a macrocell array, an expander product-term array, and an I/O control block. (The number of macrocells and expanders in the arrays varies with each device.) Macrocells are the primary resource for logic implementation, but if needed, expanders can be used to supplement the capabilities of any macrocell. The expander product-term array consists of a group of unallocated, inverted product terms. Flexible macrocells and allocatable expanders facilitate variable product-term designs

(14

Figure 2. Logic Array Block

The LAB consists of a macrocell array, an expander product-term array, and a decoupled I/O block. The flexibility of the LAB ensures high speeds and efficient device utilization.



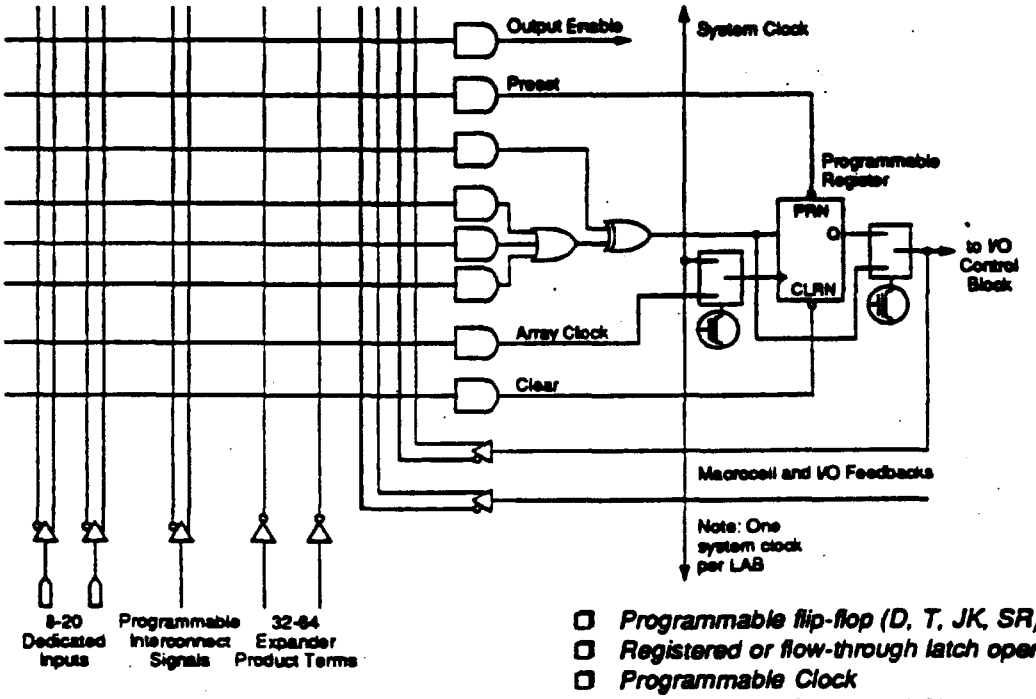
without the waste associated with fixed product-term architectures. Thus PAL or PLA devices are easily integrated into MAX EPLDs. The outputs o the macrocells feed the decoupled I/O block, which consists of a gro programmable tri-state buffers and I/O pins. In the EPM5064, EPM EPM5130, and EPM5192, multiple LABs are connected by a Programmabl Interconnect Array (PIA).

Macrocells

The EPM5000-series macrocell, shown in Figure 3, consists of a programmable logic array and an independently configurable register. This register may be programmed for D, T, JK, or SR operation; or as a flow through latch; or bypassed for purely combinatorial operation Combinatorial logic is implemented in the programmable logic array, whicl consists of three product terms ORed together that feed one input of an XOI gate. The second input to the XOR gate is also controlled by a product term that makes it possible to implement active-high or active-low logic. The XOI gate is also used for complex XOR arithmetic logic functions and fo De Morgan's inversion to reduce the number of product terms. The outpu of the XOR gate feeds the programmable register, or bypasses it for purely combinatorial operation. The logic array ensures high speed whil eliminating inefficient, unused product terms. Also, expanders can b allocated to enhance the capability of the logic array.

Additional product terms, called secondary product terms, are used fo Output Enable, Preset, Clear, and Clock logic. Preset and Clear produc terms drive the active-low asynchronous Preset and asynchronous Clea inputs to the configurable flip-flop. The Clock product term allow register to have an independent Clock and supports positive- and

Figure 3. EPM5000-Series Macrocell



- *Programmable flip-flop (D, T, JK, SR)*
- *Registered or flow-through latch operation*
- *Programmable Clock*
- *Asynchronous Preset and Clear*

3

edge-triggered operation. Macrocells that drive an output pin may use the Output Enable product term to control the active-high tri-state buffer in the I/O control block. These secondary product terms allow 7400-series TTL functions to be emulated exactly.

The EPM5000-series macrocell configurability makes it possible to efficiently integrate complete subsystems into a single device. All macrocell outputs are globally routed within an LAB and also feed the PIA to provide efficient routing of signal-intensive designs.

Clock Options

Each LAB has two clocking modes: asynchronous and synchronous. During asynchronous clocking, each flip-flop is clocked by a product term. Thus, any input or internal logic may be used as a clock. Systems that require multiple clocks are easily integrated into EPM5000-series EPLDs. Asynchronous clocking also allows each flip-flop to be configured for positive- or negative-edge-triggered operation, giving the macrocell a high degree of flexibility.

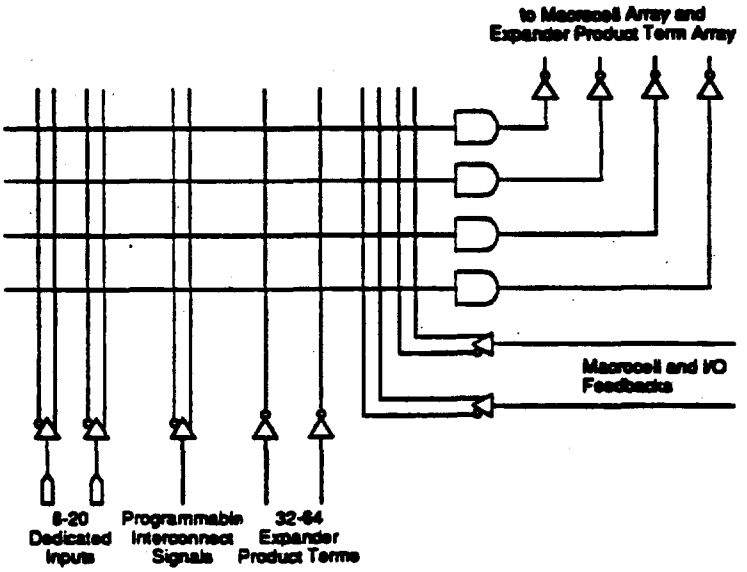
Synchronous clocking is provided by a dedicated system clock (**CLK**). This direct connection provides enhanced clock-to-output delay times. Since each LAB has one synchronous clock, all flip-flop clocks within it are positive-edge-triggered from the **CLK** pin. If the **CLK** pin is not used as a system clock, it may be used as a dedicated input.

Expander Product Terms

The expander product-term array (Figure 4) contains unallocated, invertec product terms that enhance the macrocell array. Expanders can be used anc shared by all product terms in the LAB. Wherever extra logic is needec (including register control functions), expanders can be used to implemen the logic. These expanders provide the flexibility to implement register intensive and product-term-intensive designs for MAX EPLDs.

Figure 4. Expander Product Terms

Expander product terms are unallocated logic that can be used and shared by all macrocells in an LAB. Sharing allows efficient integration of complex combinatorial functions.



Expanders are fed by all signals in the LAB. One expander may feed al macrocells in the LAB or multiple product terms in the same macrocell Since expanders also feed the secondary product terms of each macrocell complex logic functions can be implemented without using anothe macrocell. Expanders can be cross-coupled to build additional flip-flops o latches.

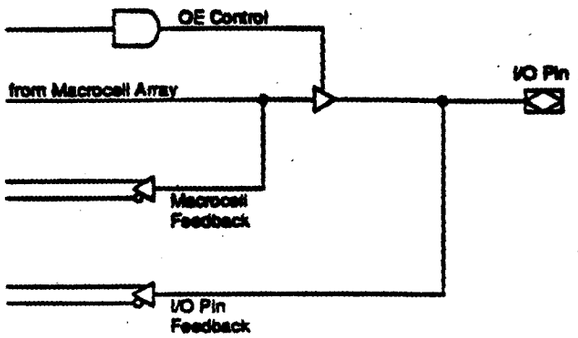
I/O Control Block

Each LAB has an I/O control block (Figure 5) that consists of a user- configurable I/O control function for each I/O pin. The I/O control block i fed by the macrocell array. The tri-state buffer is controlled by a dedicatec macrocell product term, and drives the I/O pad.

Each MAX EPLD has dual feedback—one feedback path before and on after the tri-state buffer— for every I/O pin. The tri-state buffer decouple the I/O pins from the macrocells so that all registers within the LAB can b "buried." Thus, I/O pins can be configured as dedicated input, output, or bidirectional pins. In multiple-LAB MAX devices, I/O pins feed the PIA.

Figure 5. I/O Control Block

The decoupled I/O control block features dual feedback to maximize use of device pins.



Programmable Interconnect Array

The higher-density EPM5000-series devices (EPM5064, EPM5128, EPM5130, and EPM5192) use a Programmable Interconnect Array (PIA) to route signals between the various LABs. The PIA routes only the signals required for implementing logic in an LAB, and is fed by all macrocell feedbacks and all I/O pin feedbacks. Unlike channel routing in masked or programmable gate arrays—where routing delays are variable and path-dependent—the PIA has a fixed delay. Because the PIA eliminates skew between signals, timing performance is easy to predict.

3

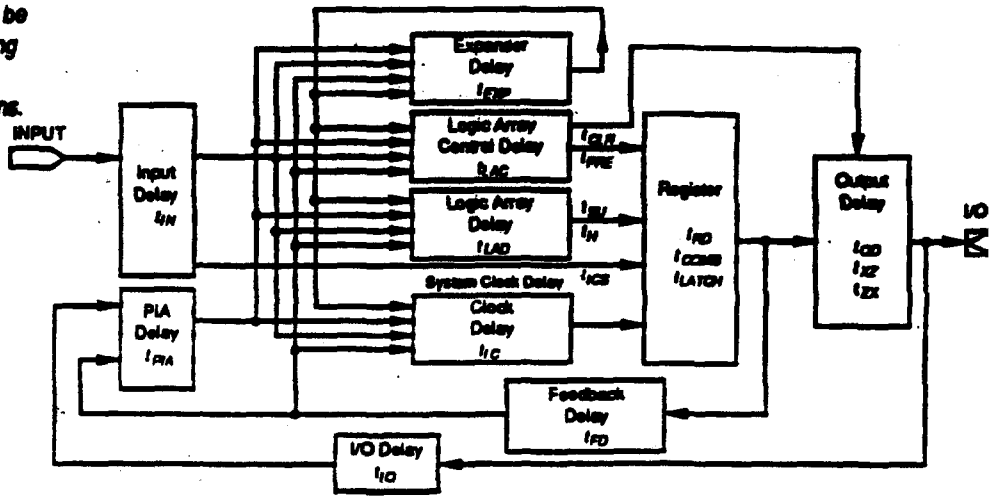
Timing Model

Timing within EPM5000-series EPLDs is easily determined with MAX+PLUS software or with the models shown in Figure 6. EPM5000-series EPLDs have fixed internal delays, that allow the user to determine the worst-case timing delays for any design. For complete timing information, MAX+PLUS software provides a timing simulator, a delay predictor, and a detailed timing analyzer.

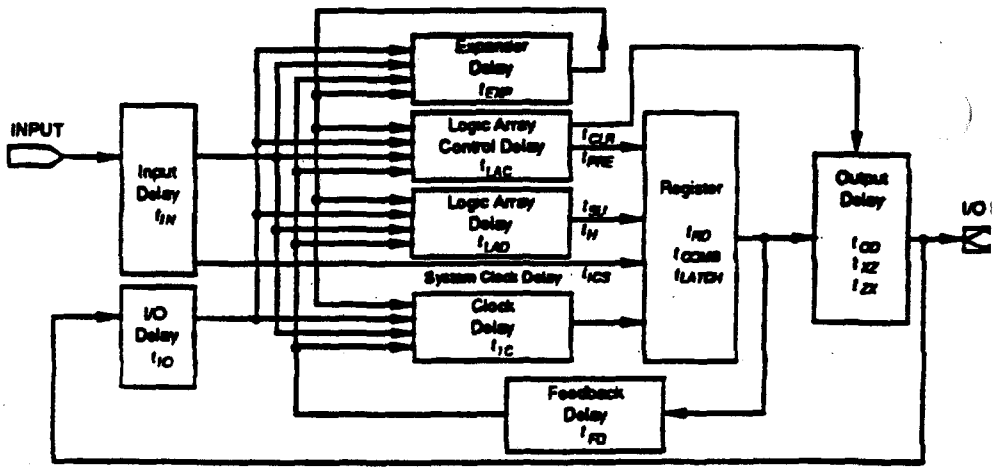
Figure 6. Timing Models

Design performance can be predicted with these timing models and the device performance specifications.

Multiple-LAB EPLDs



Single-LAB EPLDs



The timing models shown in Figure 6 may be used together with the interna timing parameters for a particular EPLD to derive timing informatior External timing parameters are derived from a sum of internal parameter and represent pin-to-pin timing delays. Figure 7 shows the internal timin waveforms for these devices. Refer to *Application Brief 75 (EPM5000-Serie MAX EPLD Timing)* in this data book for further information.

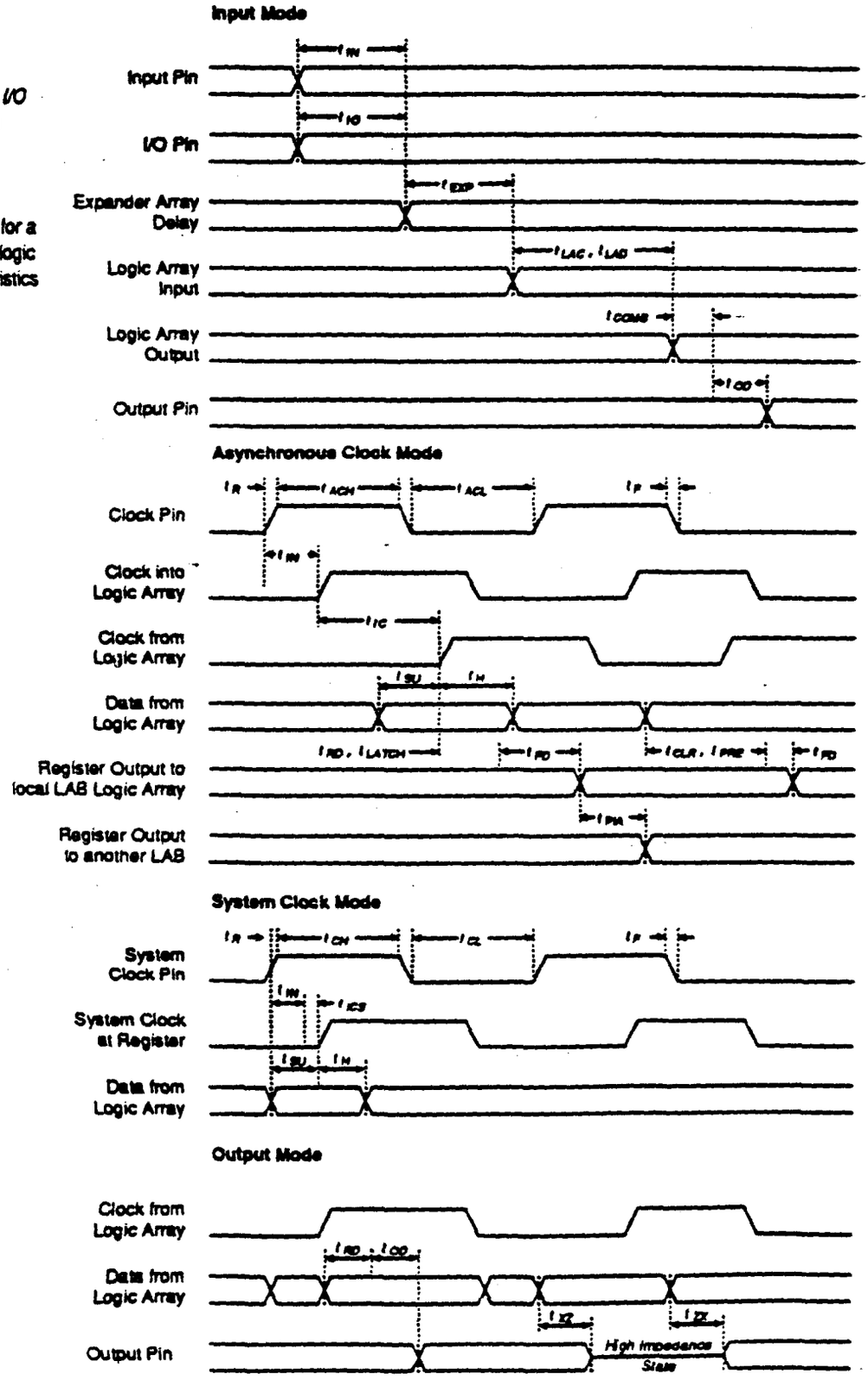
Design Security

MAX EPLDs contain a programmable Security Bit that controls access to th data programmed into the device. If this feature is used, a proprietar design implemented in the device cannot be copied or retrieved. Thi feature provides a high level of design security, since programme within EPROM cells is invisible. The Security Bit that controls this fur as well as all other program data, is reset by erasing the EPLD.

Figure 7. Switching Waveforms

In multiple LAB EPLDs, I/O pins used as inputs can traverse the PIA.

t_R & t_F < 3 ns.
Inputs are driven at 3 V for a logic high and 0 V for a logic low. All timing characteristics are measured at 1.5 V.



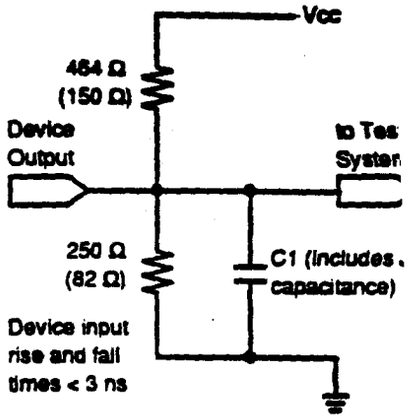
3

MAX EPLDs are fully functionally tested and guaranteed. Complete tes of each programmable EPROM bit and all internal logic elements ensu 100% programming yield. AC test measurements are performed under conditions shown in Figure 8.

Figure 8. AC Test Conditions

Power supply transients can affect AC measurements. Simultaneous transitions of multiple outputs should be avoided for accurate measurement. Threshold tests should not be performed under AC conditions. Large-amplitude, fast-ground current transients normally occur as the device outputs discharge the load capacitances. When these transients flow through the parasitic inductance between the device ground pin and the test system ground, it can create significant reductions in observable input noise immunity.

Note: Numbers in parentheses are for the EPM5016.



Test programs may be used and then erased during early stages of th production flow. This facility to use application-independent, gene purpose tests is called generic testing and is unique among user-config logic devices. EPLDs also contain on-board logic test circuitry to allov verification of function and AC specifications once they are packaged i windowless packages.

MAX+PLUS Development System

The MAX+PLUS Development System is a unified CAE system fo integrating designs into EPM5000-series MAX EPLDs. Designs can be entered as logic schematics with the Graphic Editor or as state machines, truth tables, and Boolean equations with the Altera Hardware Description Language (AHDL). Logic synthesis and minimization optimize the logic of a design. Design verification and timing analysis are performed with the Simulator or the delay prediction feature. Errors in a design are automatically located and highlighted in the schematic or text design file. Hosted on IBM PS/2, PC-AT, or compatible machines, and workstations (e.g., Apollo, Sun, IBM), MAX+PLUS gives the designer the tools to quickly and efficiently create complex logic designs. Further details about the MAX+PLUS Development System are available in the *PLS-MAX Data Sheet*.

Device Programming

EPM5000-series EPLDs may be programmed on an IBM PS/2, PC-AT or compatible computer with an Altera Logic Programmer card, the PLE3-12A Master Programming Unit, and an appropriate device adapter. These are included in the complete PLDS-MAX Development System or purchased separately. EPM5000-series EPLDs may also be pr with third-party hardware (see the *Third-Party Developme Support Data Sheet* in this data book). Contact Altera equipment manufacturer for more information.

Features

- High-density 128-macrocell general-purpose MAX EPLD
- 256 shareable expander product terms that allow over 32 product terms in a single macrocell
- High-speed multiple-LAB architecture
  - t_{PD} as fast as 25 ns
  - Counter frequencies up to 50 MHz
  - Pipelined data rates up to 62.5 MHz
- Programmable I/O architecture allowing up to 60 inputs or 52 outputs
- Available in 68-pin windowed ceramic or plastic one-time-programmable J-lead packages and in 68-pin windowed ceramic PGA packages

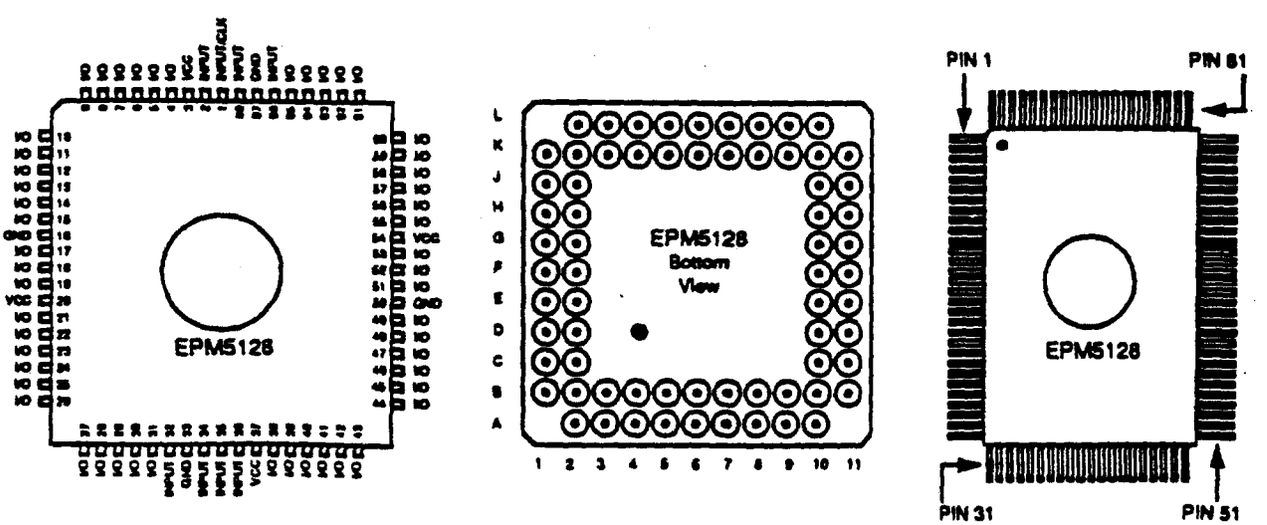
General Description

The Altera EPM5128 is a user-configurable, high-performance MAX EPLD that provides a high-density replacement for 7400-series SSI and MSI TTL and CMOS logic. (For example, a 74161 counter uses only 3% of the EPM5128.) The EPM5128 can replace over 60 TTL MSI and SSI components and integrate multiple 20- and 24-pin low-density PLDs. Figure 18 shows the J-lead and PGA package diagrams for the EPM5128.

3

Figure 18. EPM5128 Pin-Out Diagrams

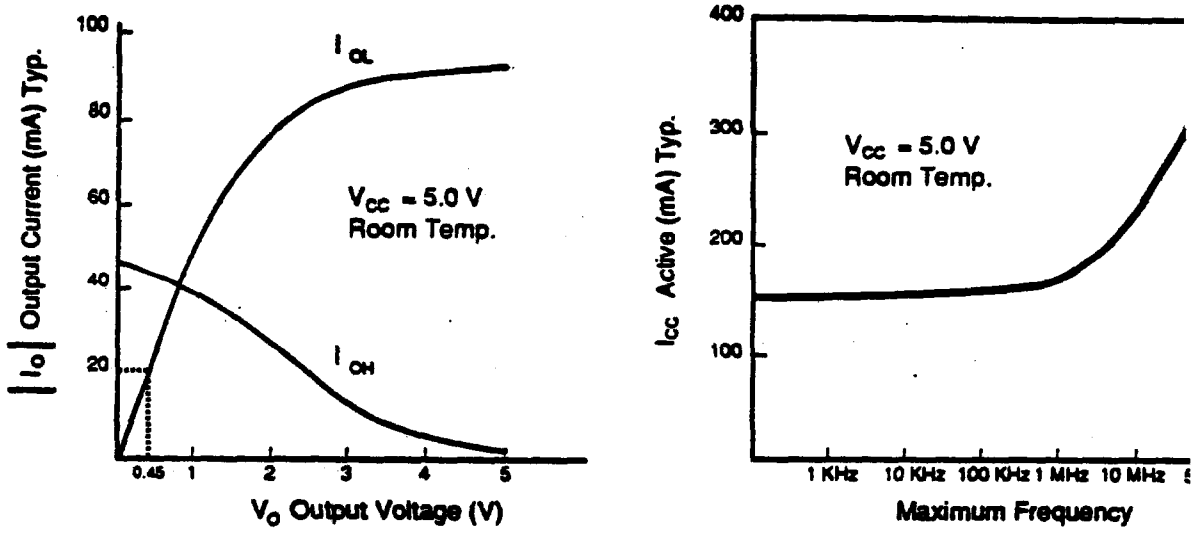
A quad flat pack (QFP) package is under development. Contact Altera Marketing for information. See Table 1 in this data sheet for PGA package pin-outs. Package outlines not drawn to scale.



J-Lead PGA QFP

Figure 19 shows output drive characteristics of EPM5128 I/O pin typical supply current versus frequency for the EPM5128.

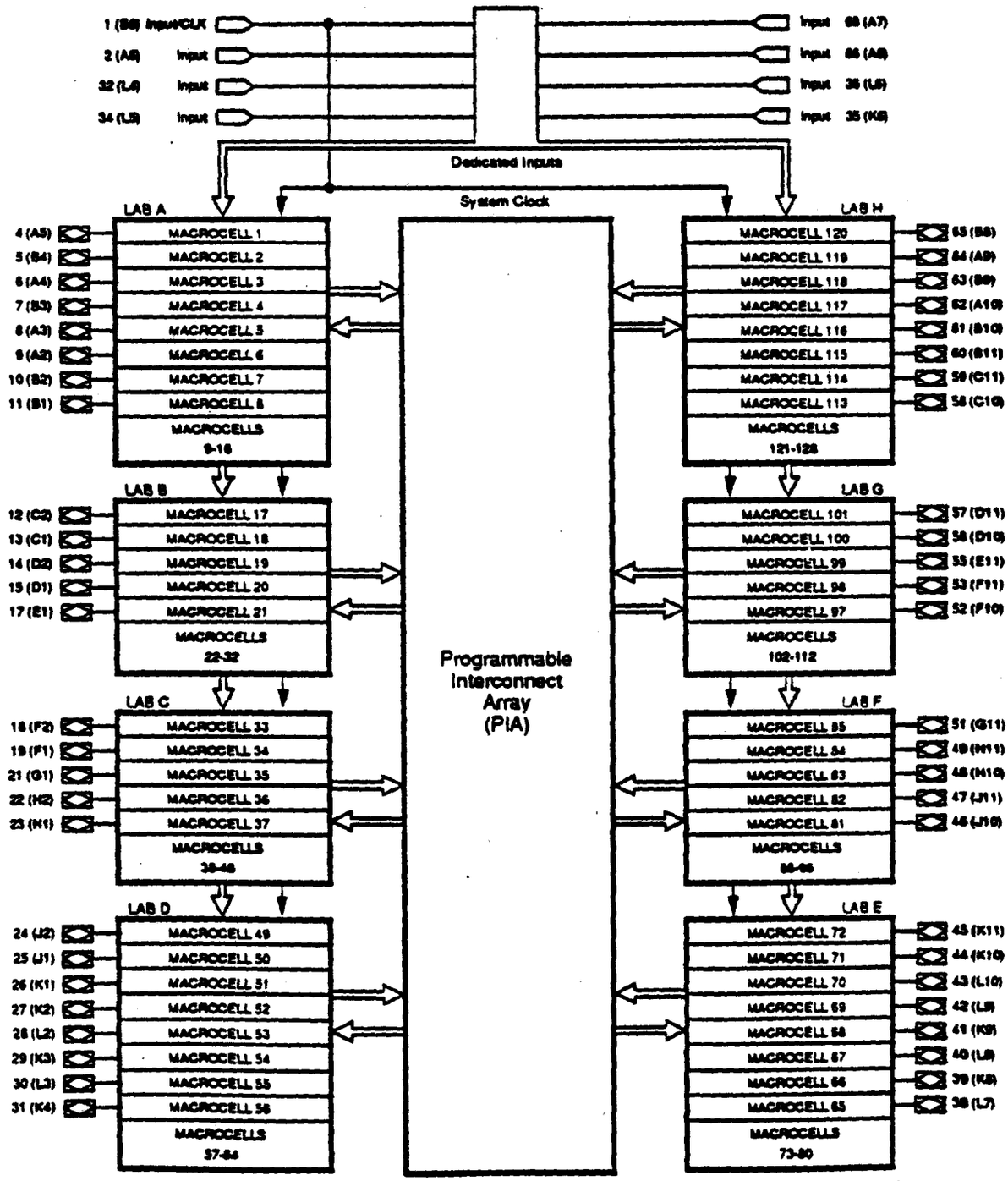
Figure 19. EPM5128 Output Drive Characteristics and I_{CC} vs. Frequency



The EPM5128 consists of 128 macrocells equally divided into 8 Logic A Blocks (LABs) that each contain 16 macrocells (see Figure 20). Each also contains 32 expander product terms. The EPM5128 has 8 dedic input pins, one of which may be used as a synchronous system clock. EPM5128 contains 52 I/O pins that can be configured for input, outpu bidirectional data flow. Four of the LABs have 8 I/O pins, and the oth have 5 I/O pins.

Figure 20. EPM5128 Block Diagram

Numbers in parentheses are for PGA packages.



Absolute Maximum Ratings Note: See *Operating Requirements for EPLDs* in this data book.

Symbol	Parameter	Conditions	Min	Max	Ur
V_{CC}	Supply voltage	With respect to GND	–2.0	7.0	V
V_{PP}	Programming supply voltage	*See Note (1)*	–2.0	13.5	V
V_I	DC input voltage		–2.0	7.0	V
I_{MAX}	DC V_{CC} or GND current			500	mA
I_{OUT}	DC output current, per pin		–25	25	mA
P_D	Power dissipation			2500	mW
T_{STG}	Storage temperature	No bias	–65	+150	°C
T_{AMB}	Ambient temperature	Under bias	–65	+135	°C
T_J	Junction temperature	Under bias		+150	°C

Recommended Operating Conditions *See Note (2)*

Symbol	Parameter	Conditions	Min	Max	Unit
V_{CC}	Supply voltage		4.75 (4.5)	5.25 (5.5)	V
V_I	Input voltage		0	V_{CC}	V
V_O	Output voltage		0	V_{CC}	V
T_A	Operating temperature	For commercial use	0	+70	°C
T_A	Operating temperature	For industrial use	–40	+85	
T_C	Case temperature	For military use	–55	+125	°C
t_R	Input rise time			100	ns
t_F	Input fall time			100	ns

DC Operating Conditions *See Note (2), (3), (4)*

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IH}	High-level input voltage		2.0		V_{CC} + 0.3	V
V_{IL}	Low-level input voltage		–0.3		0.8	V
V_{OH}	High-level TTL output voltage	I_{OH} = –4 mA DC	2.4			V
V_{OL}	Low-level output voltage	I_{OL} = 8 mA DC			0.45	V
I_I	Input leakage current	$V_I = V_{CC}$ or GND	–10		+10	µA
I_{OZ}	Tri-state output off-state current	$V_O = V_{CC}$ or GND	–40		+40	µA
I_{CC1}	V_{CC} supply current (standby)	$V_I = V_{CC}$ or GND		150	225 (300)	mA
I_{CC3}	V_{CC} supply current	$V_I = V_{CC}$ or GND No load, f = 1.0 MHz *See Note (5)*		155	250 (350)	mA

Capacitance

Symbol	Parameter	Conditions	Min	Max
C_{IN}	Input capacitance	V_{IN} = 0 V, f = 1.0 MHz		
C_{OUT}	Output capacitance	V_{OUT} = 0 V, f = 1.0 MHz		

AC Operating Conditions *See Note (4)*

External Timing Parameters			EPM5128-1		EPM5128-2		EPM5128		
Symbol	Parameter	Conditions	Min	Max	Min	Max	Min	Max	Unit
t_{PD1}	Input to non-registered output	C1 = 35 pF		25		30		35	ns
t_{PD2}	I/O input to non-reg. output	C1 = 35 pF		40		45		55	ns
t_{SU}	Setup time		15		20		25		ns
t_{H}	Hold time		0		0		0		ns
t_{CO1}	Clock to output delay	C1 = 35 pF		14		16		20	ns
t_{ASU}	Asynchronous setup time		5		6		8		ns
t_{AH}	Asynchronous hold time		6		8		10		ns
t_{CH}	Clock high time		8		10		12.5		ns
t_{CL}	Clock low time		8		10		12.5		ns
t_{ACH}	Asynchronous clock high time		11		14		16		ns
t_{ACL}	Asynchronous clock low time		9		11		14		ns
t_{ACO1}	Asynch. clock to output delay	C1 = 35 pF		25		30		35	ns
t_{CNT}	Minimum clock period			20		25		30	ns
f_{CNT}	Internal maximum frequency		50		40		33.3		MHz
t_{ACNT}	Minimum asynch. clock period	*See Note (6)*		20		25		30	ns
f_{ACNT}	Max. internal asynch. frequency	*See Note (6)*	50		40		33.3		MHz
f_{MAX}	Max. frequency; pipelined data		62.5		50		40		MHz

For information on internal timing parameters, refer to *App. Brief 75 (EPM5000-Series MAX EPLD Timing)*.

Internal Timing Parameters			EPM5128-1		EPM5128-2		EPM5128		
Symbol	Parameter	Conditions	Min	Max	Min	Max	Min	Max	Unit
t_{IN}	Input pad and buffer delay			5		7		9	ns
t_{IO}	I/O input pad and buffer delay			6		6		9	ns
t_{EXP}	Expander array delay			12		14		20	ns
t_{LAD}	Logic array delay			12		14		16	ns
t_{LAC}	Logic control array delay			10		12		13	ns
t_{OD}	Output buffer and pad delay	C1 = 35 pF		5		5		6	ns
t_{ZX}	Output buffer enable delay			10		11		13	ns
t_{XZ}	Output buffer disable delay	C1 = 5 pF		10		11		13	ns
t_{SU}	Register setup time		6		8		10		ns
t_{LATCH}	Flow-through latch delay			3		4		4	ns
t_{RD}	Register delay			1		2		2	ns
t_{COMB}	Combinatorial delay			3		4		4	ns
t_{H}	Register hold time		6		8		10		ns
t_{IC}	Clock delay			14		16		18	ns
t_{ICS}	System clock delay			2		2		3	ns
t_{FD}	Feedback delay			1		1		2	ns
t_{PRE}	Register preset time			5		6		7	ns
t_{CLR}	Register clear time			5		6		7	ns
t_{PIA}	Progr. Interconn. Array delay			14		16		20	ns

Notes to tables:

(1) Minimum DC input is -0.3 V. During transitions, the inputs may undershoot to -2.0 V or overshoot to 7.0 V for periods less than 20 ns under no-load conditions.
(2) Numbers in parentheses are for military and industrial temperature-range version
(3) Typical values are for T_A = 25° C and V_{CC} = 5 V.
(4) V_{CC} = 5 V ± 5%, T_A = 0° C to 70° C for commercial use.
V_{CC} = 5 V ± 10%, T_A = -40° C to 85° C for industrial use.
V_{CC} = 5 V ± 10%, T_C = -55° C to 125° C for military use.
(5) Measured with device programmed as a 16-bit counter in each LAB.
(6) This parameter is measured with a positive-edge-triggered clock at the register. For negative-edge clocking, the t_{ACH} and t_{ACL} parameters must be swapped.

Product Availability

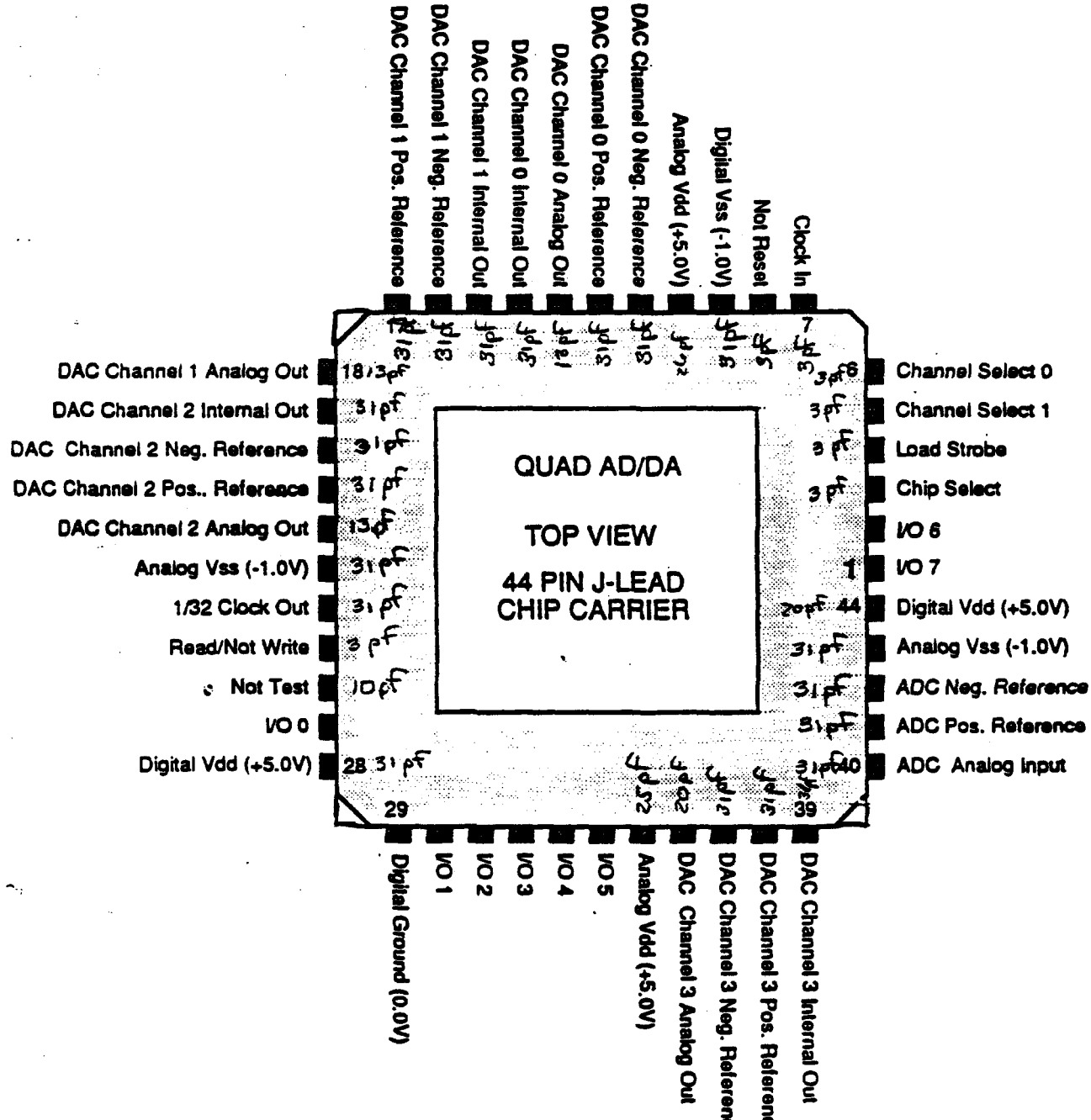
Grade		Availability
Commercial	(0° C to 70° C)	EPM5128-1, EPM5128-2, EPM5128
Industrial	(-40° C to 85° C)	EPM5128
Military	(-55° C to 125° C)	EPM5128

Note: Only military-temperature-range EPLDs are listed above. MIL-STD-883-compliant product specifications are provided in Military Product Drawings (MPDs), available from Altera Marketing by calling 1 (800) SOS-EPLD. These MPDs should be used to prepare Source Control Drawings (SCDs). See *Military Products* in this data book.

Table 1 shows the pin-outs for the EPM5128 PGA package.

Table 1. EPM5128 PGA Pin-Outs

Pin	Function	Pin	Function	Pin	Function	Pin	Function
A2	I/O	B10	I/O	G1	I/O	K7	VCC
A3	I/O	B11	I/O	G2	VCC	K8	I/O
A4	I/O	C1	I/O	G10	GND	K9	I/O
A5	I/O	C2	I/O	G11	I/O	K10	I/O
A6	Input	C10	I/O	H1	I/O	K11	I/O
A7	Input	C11	I/O	H2	I/O	L2	I/O
A8	Input	D1	I/O	H10	I/O	L3	I/O
A9	I/O	D2	I/O	H11	I/O	L4	Input
A10	I/O	D10	I/O	J1	I/O	L5	Input
B1	I/O	D11	I/O	J2	I/O	L6	Input
B2	I/O	E1	I/O	J10	I/O	L7	I/O
B3	I/O	E2	GND	J11	I/O	L8	I/O
B4	I/O	E10	VCC	K1	I/O	L9	I/O
B5	VCC	E11	I/O	K2	I/O	L10	I/O
B6	Input/CLK	F1	I/O	K3	I/O		
B7	GND	F2	I/O	K4	I/O		
B8	I/O	F10	I/O	K5	GND		
B9	I/O	F11	I/O	K6	Input		


QUAD AD/DA
TOP VIEW
44 PIN J-LEAD
CHIP CARRIER
DAC Channel 1 Analog Out
DAC Channel 2 Internal Out
DAC Channel 2 Neg. Reference
DAC Channel 2 Pos.. Reference
DAC Channel 2 Analog Out
Analog Vss (-1.0V)
1/32 Clock Out
Read/Not Write
Not Test
I/O 0
Digital Vdd (+5.0V)
DAC Channel 1 Pos. Reference
DAC Channel 1 Neg. Reference
DAC Channel 1 Internal Out
DAC Channel 0 Internal Out
DAC Channel 0 Analog Out
DAC Channel 0 Pos. Reference
DAC Channel 0 Neg. Reference
Analog Vdd (+5.0V)
Digital Vss (-1.0V)
Not Reset
Clock In
Channel Select 0
Channel Select 1
Load Strobe
Chip Select
I/O 6
I/O 7
Digital Vdd (+5.0V)
Analog Vss (-1.0V)
ADC Neg. Reference
ADC Pos. Reference
ADC Analog Input
Digital Ground (0.0V)
I/O 1
I/O 2
I/O 3
I/O 4
I/O 5
Analog Vdd (+5.0V)
DAC Channel 3 Analog Out
DAC Channel 3 Neg. Reference
DAC Channel 3 Pos. Reference
DAC Channel 3 Internal Out

15 devices 9/21/89

TESTED ASIC FOR MIN/MAX Power Supply Currents. The 4 DACs were SET to Full scale (2.56V out) and were then loaded for a 1 LSB Change (2.55V @ 17.5mA) and the power supply currents were then recorded.

	MIN (NO LOAD)	MAX	
+5V	27mA	77mA + 5mA	(SOURCING)
-1.0V	2.5 mA	26mA	(SINKING)
+2.56V	3.5mA	3.5mA	(SOURCING)

The Power Supplies were monitored for noise and Glitches — none were observed.

The above currents are the avg MIN/MAX currents of the 1552 D/A A/D chips available.

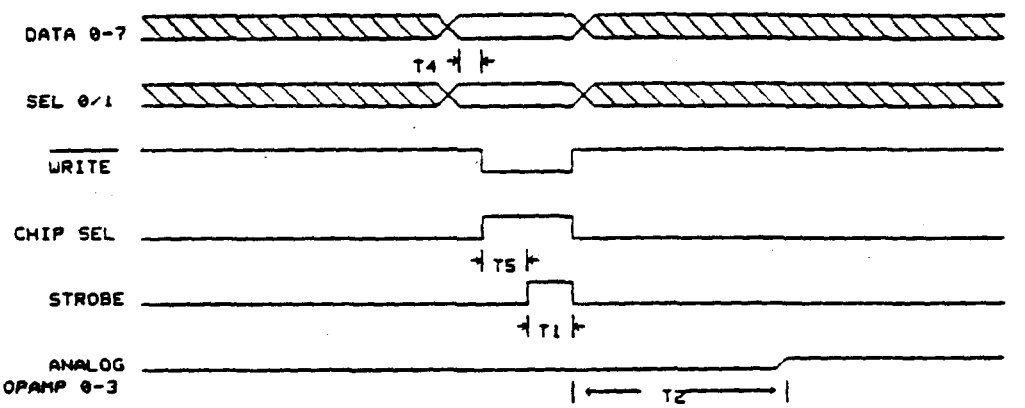
Z_{OUT} 3.0 - 14 Ω / OUTPUT

P.S. currents		USE
64 x 3.5 mA	224 mA	250
64 x 2.5 mA	160 mA	200
64 x 27	1,728 mA	2A

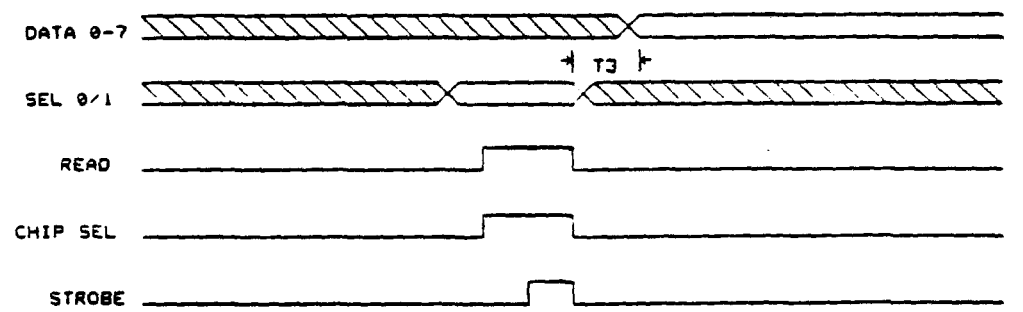
CUE CAN DRIVE ENTIRE BOARD WITH 2 PACKAGE

13

TYPICAL WRITE CYCLE



TYPICAL READ CYCLE



TIMING*

T1 = 100nS MIN T4 = 4nS MIN

T2 = 3.25uS MAX T5 = 100nS MIN

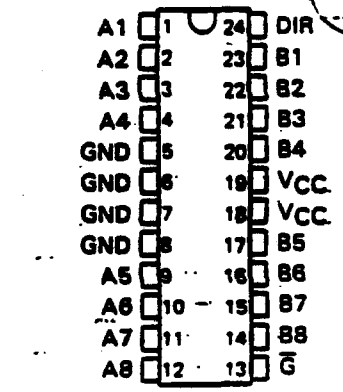
T3 = 40nS MAX

* APPROXIMATE VALUES FROM SIMULATION/DATA SHEETS

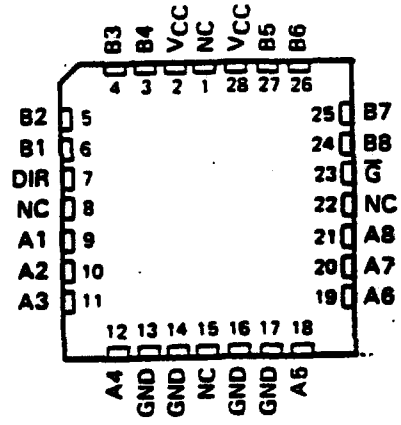
D2967, JULY 1987—REVISED AUGUST 1988

- 3-State Outputs Drive Bus Lines Directly
- Inputs are TTL-Voltage Compatible
- New Flow-Through Architecture to Optimize PCB Layout
- Center-Pin V_{CC} and GND Configurations to Minimize High-Speed Switching Noise
- EPIC™ (Enhanced-Performance Implanted CMOS) 1-μm Process
- 500-mA Typical Latch-Up Immunity at 125°C
- Package Options Include Plastic "Small Outline" Packages, Ceramic Chip Carriers, and Standard Plastic and Ceramic 300-mil DIPs

54ACT11245 . . . JT PACKAGE
74ACT11245 . . . DW OR NT PACKAGE
(TOP VIEW)



54ACT11245 . . . FK PACKAGE
(TOP VIEW)



NC—No internal connection

description

These octal bus transceivers are designed for asynchronous two-way communication between data buses. The control function implementation minimizes external timing requirements.

The device allows data transmission from the A bus to the B bus, or from the B bus to the A bus, depending upon the logic level at the direction control (DIR) input. The enable input ($\overline{G}$) can be used to disable the device so that the buses are effectively isolated.

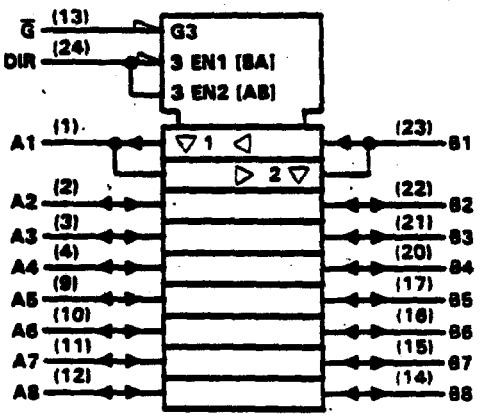
The 54ACT11245 is characterized for operation over the full military temperature range of −55°C to 125°C. The 74ACT11245 is characterized for operation from −40°C to 85°C.

FUNCTION TABLE

ENABLE $\overline{G}$	DIRECTION CONTROL DIR	OPERATION
L	L	B data to A bus
L	H	A data to B bus
H	X	Isolation

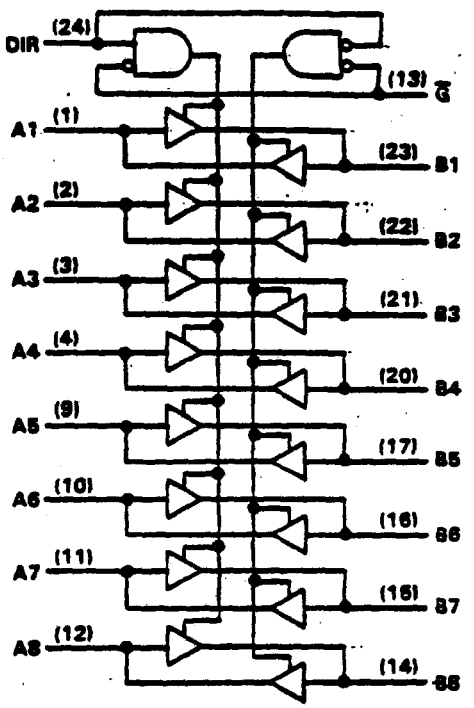
EPIC is a trademark of Texas Instruments Incorporated.

logic symbol†



† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.
Pin numbers shown are for DW, JT, and NT packages.

logic diagram (positive logic)



Pin numbers shown are for DW, JT, and NT packages.

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)‡

Supply voltage, V_{CC}	−0.5 V to 7 V
Input voltage, V_I (see Note 1)	−0.5 V to V_{CC}+0.5 V
Output voltage, V_O (see Note 1)	−0.5 V to V_{CC}+0.5 V
Input clamp current, I_{IK} (V_I < 0 or V_I > V_{CC})	±20 mA
Output clamp current, I_{OK} (V_O < 0 or V_O > V_{CC})	±50 mA
Continuous output current, I_O (V_O = 0 to V_{CC})	±50 mA
Continuous current through V_{CC} or GND pins	±200 mA
Storage temperature range	−65°C to 150°C

‡ Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 1: The input and output voltage ratings may be exceeded if the input and output current ratings are observed.

recommended operating conditions

		54ACT11245		74ACT11245		UNIT
		MIN	MAX	MIN	MAX	
V_{CC}	Supply voltage	4.5	5.5	4.5	5.5	V
V_{IH}	High-level input voltage	2		2		V
V_{IL}	Low-level input voltage		0.8		0.8	V
I_{OH}	High-level output current		−24		−24	mA
I_{OL}	Low-level output current		24		24	mA
V_I	Input voltage	0	V_{CC}	0	V_{CC}	V
V_O	Output voltage	0	V_{CC}	0	V_{CC}	V
$\Delta t/\Delta v$	Input transition rise or fall rate	0	10	0	10	ns/V
T_A	Operating free-air temperature	−55	125	−40	85	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS	V_{CC}	T_A = 25°C			54ACT11245		74ACT11245		UNIT
			MIN	TYP	MAX	MIN	MAX	MIN	MAX	
V_{OH}	I_{OH} = −50 μA	4.5 V	4.4			4.4		4.4		V
		5.5 V	5.4			5.4		5.4		
	I_{OH} = −24 mA	4.5 V	3.94			3.7		3.8		
		5.5 V	4.94			4.7		4.8		
	I_{OH} = −50 mA†	5.5 V				3.85				
	I_{OH} = −75 mA†	5.5 V						3.85		
V_{OL}	I_{OL} = 50 μA	4.5 V			0.1		0.1		0.1	V
		5.5 V			0.1		0.1		0.1	
	I_{OL} = 24 mA	4.5 V			0.36		0.5		0.44	
		5.5 V			0.36		0.5		0.44	
	I_{OL} = 50 mA†	5.5 V					1.65			
	I_{OL} = 75 mA†	5.5 V							1.65	
I_{OZ}	V_O = V_{CC} or GND	5.5 V			±0.5		±10		±5	μA
I_I	V_I = V_{CC} or GND	5.5 V			±0.1		±1		±1	μA
I_{CC}	V_I = V_{CC} or GND, I_O = 0	5.5 V			8		160		80	μA
ΔI_{CC}‡	One input at 3.4 V, Other inputs at GND or V_{CC}	5.5 V			0.9		1		1	mA
C_i	V_I = V_{CC} or GND	5 V		4						pF
C_{io}	V_{IO} = V_{CC} or GND	5 V		12						pF

† Not more than one output should be tested at a time, and the duration of the test should not exceed 10 ms.
‡ This is the increase in supply current for each input that is at one of the specified TTL voltage levels rather than 0 V or V_{CC}.

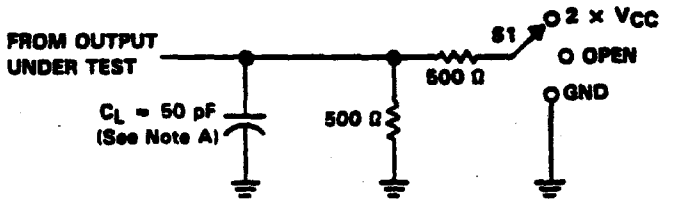
switching characteristics, V_{CC} = 5 V ±0.5 V (see Figure 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	T_A = 25°C			54ACT11245		74ACT11245		UNIT
			MIN	TYP	MAX	MIN	MAX	MIN	MAX	
t_{PLH}	A or B	B or A	1.5	6.2	9.2	1.5	10.6	1.5	10	ns
t_{PHL}			1.5	5.4	8.6	1.5	9.6	1.5	9.1	
t_{PZH}	$\overline{G}$	A or B	1.5	8.1	12	1.5	14.1	1.5	13.2	
t_{PZL}			1.5	8.2	11.7	1.5	13.7	1.5	12.9	
t_{PHZ}	$\overline{G}$	A or B	1.5	9.3	11.8	1.5	13.6	1.5	12.9	
t_{PLZ}			1.5	9.8	12.9	1.5	14.6	1.5	13.9	

operating characteristics, V_{CC} = 5 V, T_A = 25°C

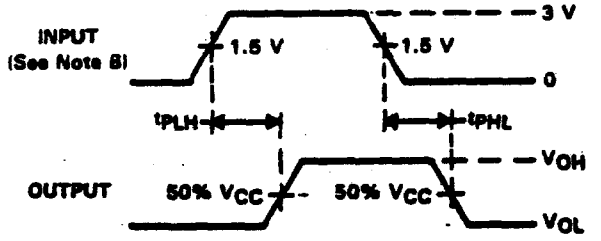
PARAMETER			TEST CONDITIONS	TYP	UNIT
C_{pd}	Power dissipation capacitance per transceiver	Outputs enabled	C_L = 50 pF, f = 1 MHz	66	pF
		Outputs disabled		19	

PARAMETER MEASUREMENT INFORMATION

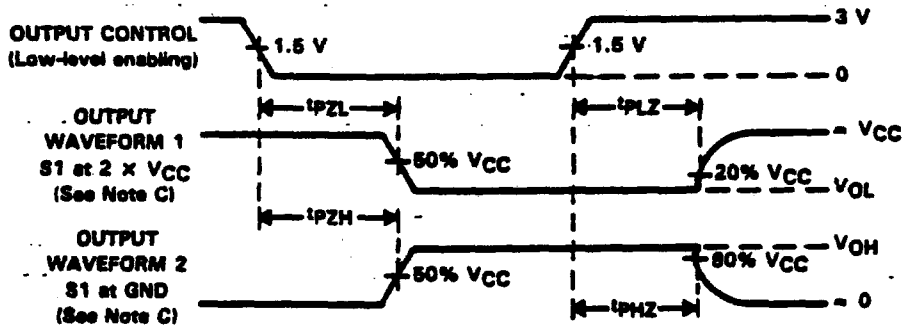


TEST	S1
t_{PLH}/t_{PHL}	OPEN
t_{PLZ}/t_{PZL}	2 × V_{CC}
t_{PHZ}/t_{PZH}	GND

LOAD CIRCUIT



PROPAGATION DELAY TIMES



ENABLE AND DISABLE TIMES

NOTES: A. C_L includes probe and jig capacitance.
B. Input pulses are supplied by generators having the following characteristics: PRR ≤ 10 MHz, Z_O = 50 Ω, t_r = 3 ns, t_f = 3 ns.
C. Waveform 1 is for an output with internal conditions such that the output is low except when disabled by the output control. Waveform 2 is for an output with internal conditions such that the output is high except when disabled by the output control.

FIGURE 1. LOAD CIRCUIT AND VOLTAGE WAVEFORMS

TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

- **Package Options Include Both Plastic and Ceramic Chip Carriers in Addition to Plastic and Ceramic DIPs**
- **Dependable Texas Instruments Quality and Reliability**

description

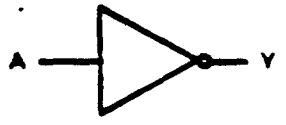
These devices contain six independent inverters.

The SN5404, SN54H04, SN54L04, SN54LS04 and SN54S04 are characterized for operation over the full military temperature range of −55°C to 125°C. The SN7404, SN74H04, SN74LS04 and SN74S04 are characterized for operation from 0°C to 70°C.

FUNCTION TABLE (each inverter)

INPUTS A	OUTPUT Y
H	L
L	H

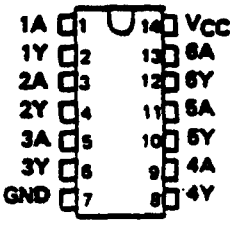
logic diagram (each inverter)



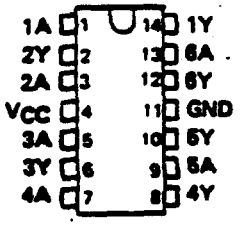
positive logic

$$Y = \overline{A}$$

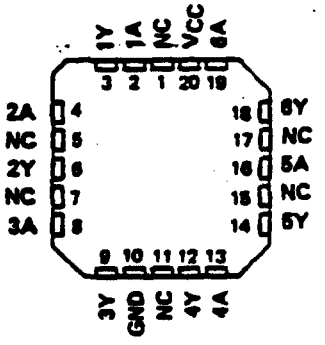
SN5404, SN54H04, SN54L04 . . . J PACKAGE
SN54LS04, SN54S04 . . . J OR W PACKAGE
SN7404, SN74H04 . . . J OR N PACKAGE
SN74LS04, SN74S04 . . . D, J OR N PACKAGE
(TOP VIEW)



SN5404, SN54H04 . . . W PACKAGE
(TOP VIEW)



SN54LS04, SN54S04 . . . FK PACKAGE
SN74LS04, SN74S04 . . . FN PACKAGE
(TOP VIEW)



NC – No internal connection

3 TTL DEVICES

National Semiconductor

LM111/LM211/LM311 Voltage Comparator

General Description

The LM111, LM211 and LM311 are voltage comparators that have input currents nearly a thousand times lower than devices like the LM106 or LM710. They are also designed to operate over a wider range of supply voltages: from standard ±15V op amp supplies down to the single 5V supply used for IC logic. Their output is compatible with RTL, DTL and TTL as well as MOS circuits. Further, they can drive lamps or relays, switching voltages up to 50V at currents as high as 50 mA.

Both the inputs and the outputs of the LM111, LM211 or the LM311 can be isolated from system ground, and the output can drive loads referred to ground, the positive supply or the negative supply. Offset balancing and strobe capability are provided and outputs can be wire OR'ed. Although slower than the LM106 and LM710 (200 ns response time vs 40 ns) the devices are also much less prone to spurious oscillations. The LM111 has the same pin configuration as the LM106 and LM710.

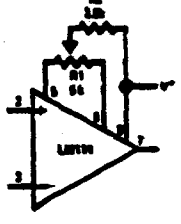
The LM211 is identical to the LM111, except that its performance is specified over a −25°C to +85°C temperature range instead of −55°C to +125°C. The LM311 has a temperature range of 0°C to +70°C.

Features

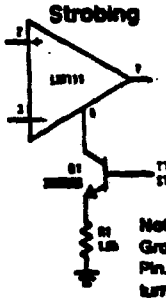
- Operates from single 5V supply
- Input current: 150 nA max. over temperature
- Offset current: 20 nA max. over temperature
- Differential input voltage range: ±30V
- Power consumption: 135 mW at ±15V

Typical Applications**

Offset Balancing



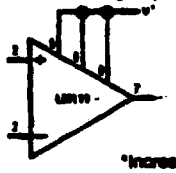
Strobing



Note: Do Not Ground Strobe Pin. Output is turned off when current is pulled from Strobe Pin.

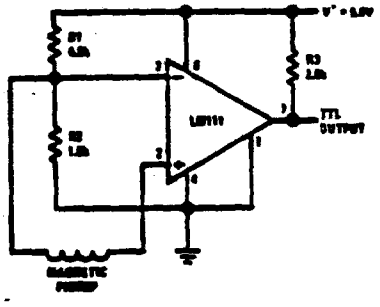
**Note: Pin connections shown on schematic diagram and typical applications are for H08 metal can package.

Increasing Input Stage Current*

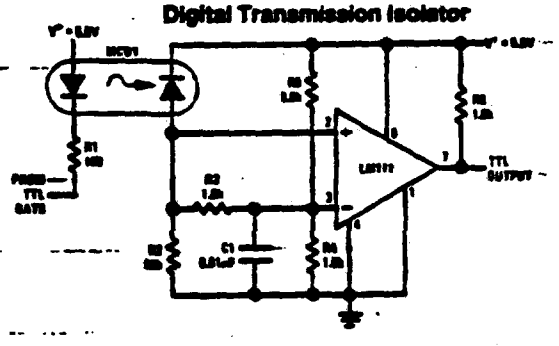


*Increases typical common-mode slew from 7.0V/μs to 18V/μs.

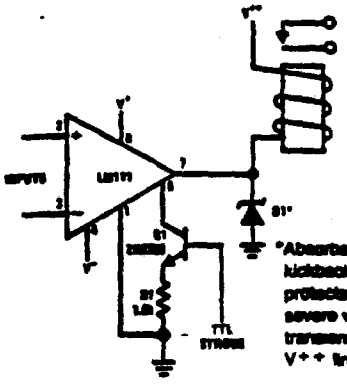
Detector for Magnetic Transducer



Digital Transmission Isolator



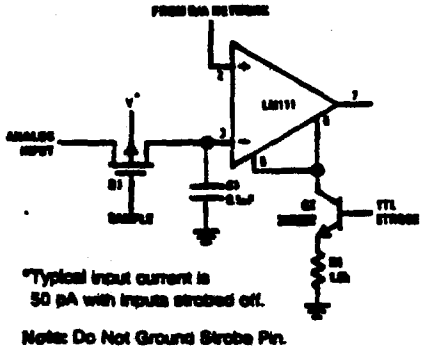
Relay Driver with Strobe



*Absorbs inductive kickback of relay and protects IC from severe voltage transients on V+ + line.

Note: Do Not Ground Strobe Pin.

Strobing off Both Input* and Output Stages



*Typical input current is 50 pA with inputs strobed off.

Note: Do Not Ground Strobe Pin.

TL/H/5704–1

5

21

Absolute Maximum Ratings for the LM111/LM211

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

(Note 7)	
Total Supply Voltage (V_{84})	36V
Output to Negative Supply Voltage (V_{74})	50V
Ground to Negative Supply Voltage (V_{14})	30V
Differential Input Voltage	±30V
Input Voltage (Note 1)	±15V
Power Dissipation (Note 2)	500 mW
Output Short Circuit Duration	10 sec
Operating Temperature Range LM111	−55°C to 125°C
LM211	−25°C to 85°C
Storage Temperature Range	−65°C to 15
Lead Temperature (Soldering, 10 sec)	260°C
Voltage at Strobe Pin	V+ −5V
Soldering Information	
Dual-In-Line Package	
Soldering (10 seconds)	260°C
Small Outline Package	
Vapor Phase (60 seconds)	215°C
Infrared (15 seconds)	220°C

See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" for other methods of soldering surface mount devices.

ESD Rating (Note 8) 300V

Electrical Characteristics for the LM111 and LM211 (Note 3)

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage (Note 4)	$T_A = 25°C$, $R_S \leq 50k$		0.7	3.0	mV
Input Offset Current (Note 4)	$T_A = 25°C$		4.0	10	nA
Input Bias Current	$T_A = 25°C$		60	100	nA
Voltage Gain	$T_A = 25°C$	40	200		V/mV
Response Time (Note 5)	$T_A = 25°C$		200		ns
Saturation Voltage	$V_{IN} \leq -5$ mV, $I_{OUT} = 50$ mA, $T_A = 25°C$		0.75	1.5	V
Strobe ON Current (Note 6)	$T_A = 25°C$	2.0	3.0	5.0	mA
Output Leakage Current	$V_{IN} \geq 5$ mV, $V_{OUT} = 35V$, $T_A = 25°C$, $I_{STROBE} = 3$ mA		0.2	10	nA
Input Offset Voltage (Note 4)	$R_S \leq 50$ k			4.0	mV
Input Offset Current (Note 4)				20	nA
Input Bias Current				150	nA
Input Voltage Range	$V^+ = 15V$, $V^- = -15V$, Pin 7 Pull-Up May Go To 5V	−14.5	13.8,-14.7	13.0	V
Saturation Voltage	$V^+ \geq 4.5V$, $V^- = 0$, $V_{IN} \leq -6$ mV, $I_{OUT} \leq 8$ mA		0.23	0.4	V
Output Leakage Current	$V_{IN} \geq 5$ mV, $V_{OUT} = 35V$		0.1	0.5	μA
Positive Supply Current	$T_A = 25°C$		5.1	6.0	mA
Negative Supply Current	$T_A = 25°C$		4.1	5.0	mA

Note 1: This rating applies for ±15 supplies. The positive input voltage limit is 30V above the negative supply. The negative input voltage limit is equal to the negative supply voltage or 30V below the positive supply, whichever is less.

Note 2: The maximum junction temperature of the LM111 is 150°C, while that of the LM211 is 110°C. For operating at elevated temperatures, devices in the H08 package must be derated based on a thermal resistance of 165°C/W, junction to ambient, or 20°C/W, junction to case. The thermal resistance of the dual-in-line package is 110°C/W, junction to ambient.

Note 3: These specifications apply for $V_S = \pm 15V$ and Ground pin at ground, and $-55°C \leq T_A \leq +125°C$, unless otherwise stated. With the LM211, however, all temperature specifications are limited to $-25°C \leq T_A \leq +85°C$. The offset voltage, offset current and bias current specifications apply for any supply voltage from a single 5V supply up to ±15V supplies.

Note 4: The offset voltages and offset currents given are the maximum values required to drive the output within a volt of either supply with a 1 mA load. Thus, these parameters define an error band and take into account the worst-case effects of voltage gain and input impedance.

Note 5: The response time specified (see definitions) is for a 100 mV input step with 5 mV overdrive.

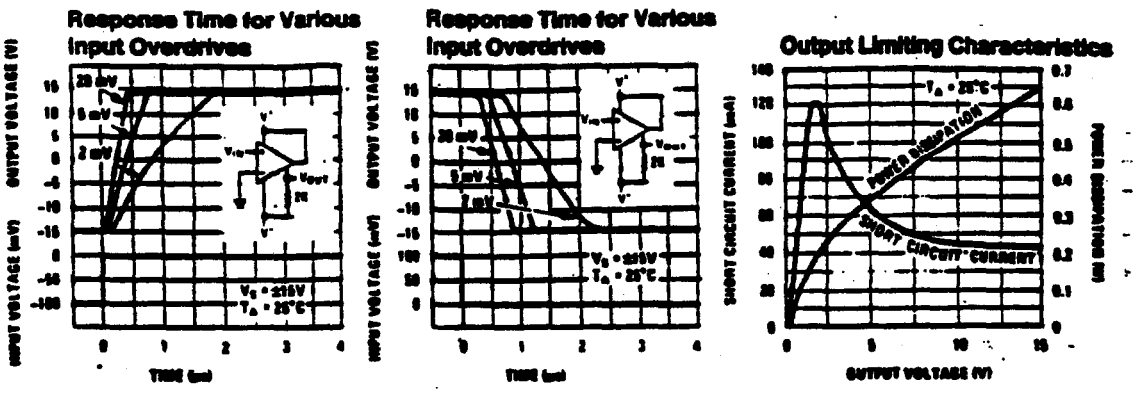
Note 6: This specification gives the range of current which must be drawn from the strobe pin to ensure the output is properly disabled. Do not short the strobe to ground; it should be current driven at 3 to 5 mA.

Note 7: Refer to RETS111X for the LM111H, LM111J and LM111J-8 military specifications.

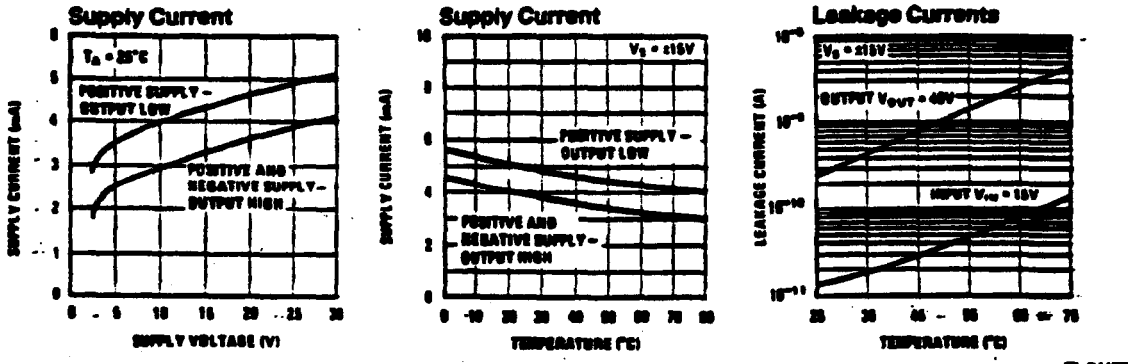
Note 8: Human body model, 1.5 kΩ in series with 100 pF.

(21)

LM311 Typical Performance Characteristics (Continued)



TL/H/5704–11



TL/H/5704–12

(21)

Absolute Maximum Ratings for the LM311

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (V_{84})	36V
Output to Negative Supply Voltage V_{74})	40V
Ground to Negative Supply Voltage V_{14})	30V
Differential Input Voltage	±30V
Input Voltage (Note 1)	±15V
Power Dissipation (Note 2)	500 mW
ESD Rating (Note 7)	300V
Output Short Circuit Duration	10 sec
Operating Temperature Range	0° to 70°C
Storage Temperature Range	−65°C to 150°C
Lead Temperature (soldering, 10 sec)	260°C
Voltage at Strobe Pin	$V^+ - 5V$

Soldering Information

Dual-in-Line Package

Soldering (10 seconds) ..... 260°C

Small Outline Package

Vapor Phase (60 seconds) ..... 215°C

Infrared (15 seconds) ..... 220°C

See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" for other methods of soldering surface mount devices.

Electrical Characteristics for the LM311 (Note 3)

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage (Note 4)	$T_A = 25°C$, $R_S \leq 50k$		2.0	7.5	mV
Input Offset Current (Note 4)	$T_A = 25°C$		6.0	50	nA
Input Bias Current	$T_A = 25°C$		100	250	nA
Voltage Gain	$T_A = 25°C$	40	200		V/mV
Response Time (Note 5)	$T_A = 25°C$		200		ns
Saturation Voltage	$V_{IN} \leq -10$ mV, $I_{OUT} = 50$ mA $T_A = 25°C$		0.75	1.5	V
Strobe ON Current	$T_A = 25°C$	1.5	3.0		mA
Output Leakage Current	$V_{IN} \geq 10$ mV, $V_{OUT} = 35V$ $T_A = 25°C$, $I_{STROBE} = 3$ mA $V^- = V_{GRND} = -5V$		0.2	50	nA
Input Offset Voltage (Note 4)	$R_S \leq 50K$			10	mV
Input Offset Current (Note 4)				70	nA
Input Bias Current				300	nA
Input Voltage Range		−14.5	13.8, −14.7	13.0	V
Saturation Voltage	$V^+ \geq 4.5V$, $V^- = 0$ $V_{IN} \leq -10$ mV, $I_{OUT} \leq 8$ mA		0.23	0.4	V
Positive Supply Current	$T_A = 25°C$		5.1	7.5	mA
Negative Supply Current	$T_A = 25°C$		4.1	5.0	mA

Note 1: This rating applies for ±15V supplies. The positive input voltage limit is 30V above the negative supply. The negative input voltage limit is equal to the negative supply voltage or 30V below the positive supply, whichever is less.

Note 2: The maximum junction temperature of the LM311 is 110°C. For operating at elevated temperature, devices in the H08 package must be derated based on a thermal resistance of 165°C/W, junction to ambient, or 20°C/W, junction to case. The thermal resistance of the dual-in-line package is 100°C/W, junction to ambient.

Note 3: These specifications apply for $V_S = \pm 15V$ and the Ground pin at ground, and $0°C < T_A < +70°C$, unless otherwise specified. The offset voltage, offset current and bias current specifications apply for any supply voltage from a single 5V supply up to ±15V supplies.

Note 4: The offset voltages and offset currents given are the maximum values required to drive the output within a volt of either supply with 1 mA load. Thus, these parameters define an error band and take into account the worst-case effects of voltage gain and input impedance.

Note 5: The response time specified (see definitions) is for a 100 mV input step with 5 mV overdrive.

Note 6: This specification gives the range of current which must be drawn from the strobe pin to ensure the output is properly disabled. Do not short the strobe pin to ground; it should be current driven at 3 to 5 mA.

Note 7: Human body model, 1.5 kΩ in series with 100 pF.

5

National Semiconductor

LM35/LM35A/LM35C/LM35CA/LM35D Precision Centigrade Temperature Sensors

General Description

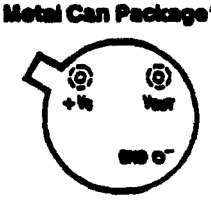
The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature. The LM35 thus has an advantage over linear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constant voltage from its output to obtain convenient Centigrade scaling. The LM35 does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55 to +150°C temperature range. Low cost is assured by trimming and calibration at the wafer level. The LM35's low output impedance, linear output, and precise inherent calibration make interfacing to readout or control circuitry especially easy. It can be used with single power supplies, or with plus and minus supplies. As it draws only 60 µA from its supply, it has very low self-heating, less than 0.1°C in still air. The LM35 is rated to operate over a −55° to +150°C temperature range, while the LM35C is rated for a −40° to +110°C range (−10° with improved accuracy). The LM35 series is available packaged in hermetic TO-46 transistor packages, while the LM35C is also available in the plastic TO-92 transistor package.

Features

- Calibrated directly in ° Celsius (Centigrade)
- Linear + 10.0 mV/°C scale factor
- 0.5°C accuracy guaranteeable (at +25°C)
- Rated for full −55° to +150°C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Less than 60 µA current drain
- Low self-heating, 0.08°C in still air
- Nonlinearity only ±¼°C typical
- Low impedance output, 0.1 Ω for 1 mA load

Connection Diagrams

TO-46
Metal Can Package*



BOTTOM VIEW

TL/H/5516–1

*Case is connected to negative pin

Order Number LM35H, LM35AH, LM35CH, LM35CAH or LM35DH
See NS Package Number H03H

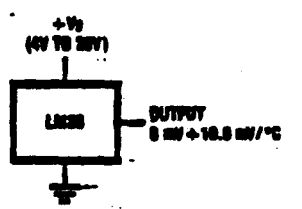
TO-92
Plastic Package

BOTTOM VIEW

TL/H/5516–2

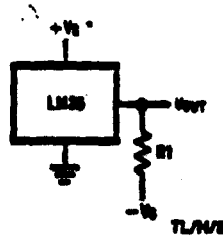
Order Number LM35CZ or LM35DZ
See NS Package Number Z03A

Typical Applications



TL/H/5516–3

FIGURE 1. Basic Centigrade Temperature Sensor (+2°C to +150°C)



Choose $R_1 = -V_S/50\ \mu A$

$V_{OUT} = +1,500$ mV at +150°C
$= +250$ mV at +25°C
$= -550$ mV at −55°C

TL/H/5516–4

FIGURE 2. Full-Range Centigrade Temperature Sensor

22

Absolute Maximum Ratings (Note 10)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage	+35V to −0.2V
Output Voltage	+6V to −1.0V
Output Current	10 mA
Storage Temp., TO-46 Package,	−60°C to +180°C
TO-92 Package,	−60°C to +150°C
Lead Temp. (Soldering, 10 seconds):	
TO-46 Package,	300°C
TO-92 Package,	260°C

Specified Operating Temperature Range: T_{MIN} to T_{MAX} (Note 2)

LM35, LM35A	−55°C to +150°C
LM35C, LM35CA	−40°C to +110°C
LM35D	0°C to +100°C

Electrical Characteristics (Note 1) (Note 6)

Parameter	Conditions	LM35A			LM35CA			Units (Max.)
		Typical	Tested Limit (Note 4)	Design Limit (Note 5)	Typical	Tested Limit (Note 4)	Design Limit (Note 5)	
Accuracy (Note 7)	$T_A = +25°C$	±0.2	±0.5		±0.2	±0.5		°C
	$T_A = -10°C$	±0.3			±0.3		±1.0	°C
	$T_A = T_{MAX}$	±0.4	±1.0		±0.4	±1.0		°C
	$T_A = T_{MIN}$	±0.4	±1.0		±0.4		±1.5	°C
Nonlinearity (Note 8)	$T_{MIN} \le T_A \le T_{MAX}$	±0.18		±0.35	±0.18		±0.3	°C
Sensor Gain (Average Slope)	$T_{MIN} \le T_A \le T_{MAX}$	+10.0	+9.8, +10.1		+10.0		+9.8, +10.1	mV/°C
Load Regulation (Note 3) $0 \le I_L \le 1$ mA	$T_A = +25°C$	±0.4	±1.0		±0.4	±1.0		mV/mA
	$T_{MIN} \le T_A \le T_{MAX}$	±0.5		±3.0	±0.5		±3.0	mV/mA
Line Regulation (Note 3)	$T_A = +25°C$	±0.01	±0.05		±0.01	±0.05		mV/V
	$4V \le V_S \le 30V$	±0.02		±0.1	±0.02		±0.1	mV/V
Quiescent Current (Note 9)	$V_S = +5V, +25°C$	56	67		56	67		µA
	$V_S = +5V$	105		131	91		114	µA
	$V_S = +30V, +25°C$	56.2	68		56.2	68		µA
	$V_S = +30V$	105.5		133	91.5		116	µA
Change of Quiescent Current (Note 3)	$4V \le V_S \le 30V, +25°C$	0.2	1.0		0.2	1.0		µA
	$4V \le V_S \le 30V$	0.5		2.0	0.5		2.0	µA
Temperature Coefficient of Quiescent Current		+0.39		+0.5	+0.39		+0.5	µA/°C
Minimum Temperature for Rated Accuracy	In circuit of *Figure 1*, $I_L = 0$	+1.5		+2.0	+1.5		+2.0	°C
Long Term Stability	$T_J = T_{MAX}$, for 1000 hours	±0.08			±0.08			°C

Note 1: Unless otherwise noted, these specifications apply: $-55°C \le T_J \le +150°C$ for the LM35 and LM35A; $-40° \le T_J \le +110°C$ for the LM35C and LM35CA; and $0° \le T_J \le +100°C$ for the LM35D. $V_S = +5$Vdc and $I_{LOAD} = 50$ µA, in the circuit of *Figure 2*. These specifications also apply from +2°C to T_{MAX} in the circuit of *Figure 1*. Specifications in boldface apply over the full rated temperature range.

Note 2: Thermal resistance of the TO-46 package is 440°C/W, junction to ambient, and 24°C/W junction to case. Thermal resistance of the TO-92 package is 180°C/W junction to ambient.

Electrical Characteristics (Note 1) (Note 6) (Continued)

Parameter	Conditions	LM35			LM35C, LM35D			Units (Max.)
		Typical	Tested Limit (Note 4)	Design Limit (Note 5)	Typical	Tested Limit (Note 4)	Design Limit (Note 5)	
Accuracy, LM35, LM35C (Note 7)	$T_A = +25°C$	±0.4	±1.0		±0.4	±1.0		°C
	$T_A = -10°C$	±0.5			±0.5		±1.5	°C
	$T_A = T_{MAX}$	±0.8	±1.5		±0.8		±1.5	°C
	$T_A = T_{MIN}$	±0.8		±1.5	±0.8		±2.0	°C
Accuracy, LM35D (Note 7)	$T_A = +25°C$				±0.6	±1.5		°C
	$T_A = T_{MAX}$				±0.9		±2.0	°C
	$T_A = T_{MIN}$				±0.9		±2.0	°C
Nonlinearity (Note 8)	$T_{MIN} \le T_A \le T_{MAX}$	±0.3		±0.5	±0.2		±0.5	°C
Sensor Gain (Average Slope)	$T_{MIN} \le T_A \le T_{MAX}$	+10.0	+9.8, +10.2		+10.0		+9.8, +10.2	mV/°C
Load Regulation (Note 3) $0 \le I_L \le 1$ mA	$T_A = +25°C$	±0.4	±2.0		±0.4	±2.0		mV/mA
	$T_{MIN} \le T_A \le T_{MAX}$	±0.5		±5.0	±0.5		±5.0	mV/mA
Line Regulation (Note 3)	$T_A = +25°C$	±0.01	±0.1		±0.01	±0.1		mV/V
	$4V \le V_S \le 30V$	±0.02		±0.2	±0.02		±0.2	mV/V
Quiescent Current (Note 9)	$V_S = +5V, +25°C$	56	80		56	80		µA
	$V_S = +5V$	105		158	91		138	µA
	$V_S = +30V, +25°C$	56.2	82		56.2	82		µA
	$V_S = +30V$	105.5		161	91.5		141	µA
Change of Quiescent Current (Note 3)	$4V \le V_S \le 30V, +25°C$	0.2	2.0		0.2	2.0		µA
	$4V \le V_S \le 30V$	0.5		3.0	0.5		3.0	µA
Temperature Coefficient of Quiescent Current		+0.39		+0.7	+0.39		+0.7	µA/°C
Minimum Temperature for Rated Accuracy	In circuit of *Figure 1*, $I_L = 0$	+1.5		+2.0	+1.5		+2.0	°C
Long Term Stability	$T_J = T_{MAX}$, for 1000 hours	±0.08			±0.08			°C

Note 2: Regulation is measured at constant junction temperature, using pulse testing with a low duty cycle. Changes in output due to heating effects can be computed by multiplying the internal dissipation by the thermal resistance.

Note 4: Tested Limits are guaranteed and 100% tested in production.

Note 5: Design Limits are guaranteed (but not 100% production tested) over the indicated temperature and supply voltage ranges. These limits are not used to calculate outgoing quality levels.

Note 6: Specifications in boldface apply over the full rated temperature range.

Note 7: Accuracy is defined as the error between the output voltage and 10mv/°C times the device's case temperature, at specified conditions of voltage, current, and temperature (expressed in °C).

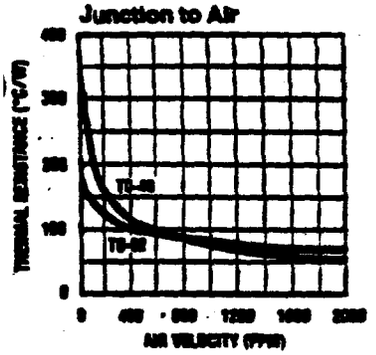
Note 8: Nonlinearity is defined as the deviation of the output-voltage-versus-temperature curve from the best-fit straight line, over the device's rated temperature range.

Note 9: Quiescent current is defined in the circuit of *Figure 1*.

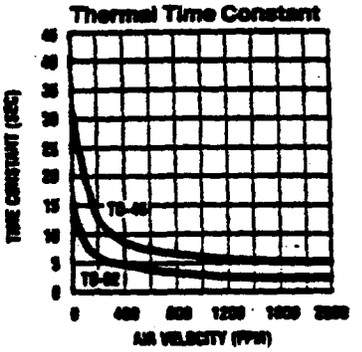
Note 10: Absolute Maximum Ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications do not apply when operating the device beyond its rated operating conditions. See Note 1.

Typical Performance Characteristics

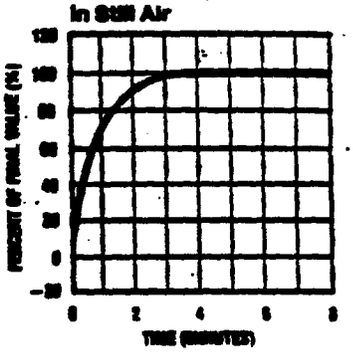
Thermal Resistance Junction to Air



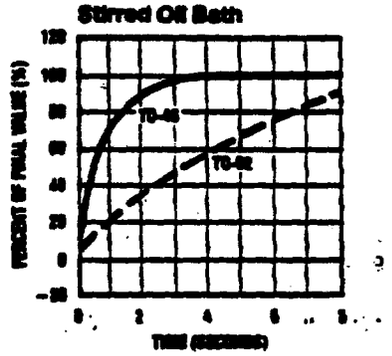
Thermal Time Constant



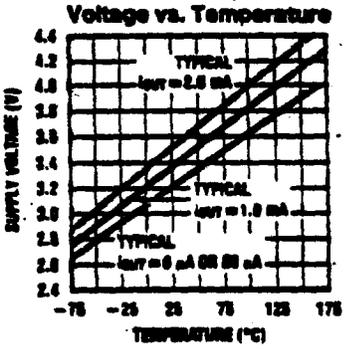
Thermal Response in Still Air



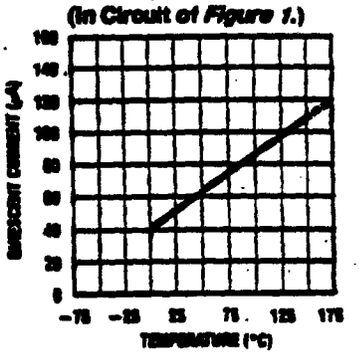
Thermal Response in Stirred Oil Bath



Minimum Supply Voltage vs. Temperature

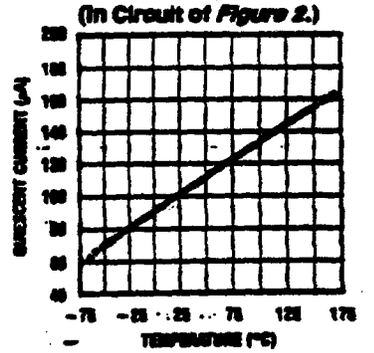


Quiescent Current vs. Temperature (In Circuit of *Figure 1.*)

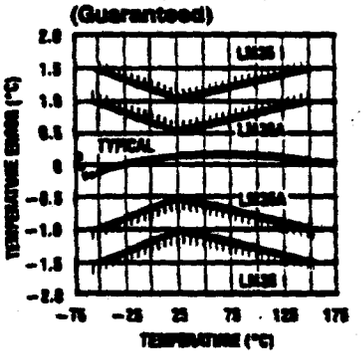


TL/H/5516–17

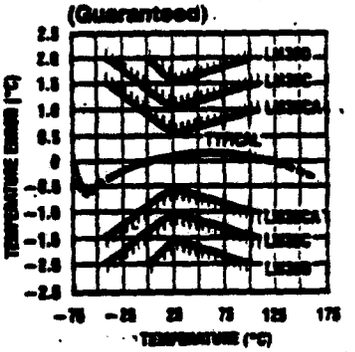
Quiescent Current vs. Temperature (In Circuit of *Figure 2.*)



Accuracy vs. Temperature (Guaranteed)



Accuracy vs. Temperature (Guaranteed)



TL/H/5516–18

6

22

Applications

The LM35 can be applied easily in the same way as other integrated-circuit temperature sensors. It can be glued or cemented to a surface and its temperature will be within about 0.01°C of the surface temperature.

This presumes that the ambient air temperature is almost the same as the surface temperature; if the air temperature were much higher or lower than the surface temperature, the actual temperature of the LM35 die would be at an intermediate temperature between the surface temperature and the air temperature. This is especially true for the TO-92 plastic package, where the copper leads are the principal thermal path to carry heat into the device, so its temperature might be closer to the air temperature than to the surface temperature.

To minimize this problem, be sure that the wiring to the LM35, as it leaves the device, is held at the same temperature as the surface of interest. The easiest way to do this is to cover up these wires with a bead of epoxy which will insure that the leads and wires are all at the same temperature as the surface, and that the LM35 die's temperature will not be affected by the air temperature.

The TO-46 metal package can also be soldered to a metal surface or pipe without damage. Of course, in that case the V− terminal of the circuit will be grounded to that metal. Alternatively, the LM35 can be mounted inside a sealed-end metal tube, and can then be dipped into a bath or screwed into a threaded hole in a tank. As with any IC, the LM35 and accompanying wiring and circuits must be kept insulated and dry, to avoid leakage and corrosion. This is especially true if the circuit may operate at cold temperatures where condensation can occur. Printed-circuit coatings and varnishes such as Humiseal and epoxy paints or dips are often used to insure that moisture cannot corrode the LM35 or its connections.

These devices are sometimes soldered to a small lightweight heat fin, to decrease the thermal time constant and speed up the response in slowly-moving air. On the other hand, a small thermal mass may be added to the sensor, to give the steadiest reading despite small deviations in the air temperature.

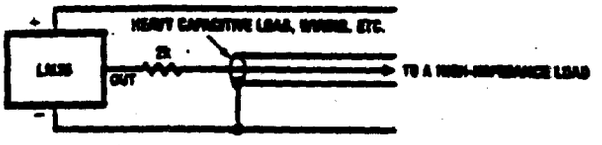
Temperature Rise of LM35 Due To Self-heating (Thermal Resistance)

	TO-46, no heat sink	TO-46, small heat fin*	TO-92, no heat sink	TO-92, small heat fin**
Still air	400°C/W	100°C/W	180°C/W	140°C/W
Moving air	100°C/W	40°C/W	90°C/W	70°C/W
Still oil	100°C/W	40°C/W	90°C/W	70°C/W
Stirred oil	50°C/W	30°C/W	45°C/W	40°C/W
(Clamped to metal, Infinite heat sink)	(24°C/W)			

* Wakefield type 201, or 1" disc of 0.020" sheet brass, soldered to case, or similar.

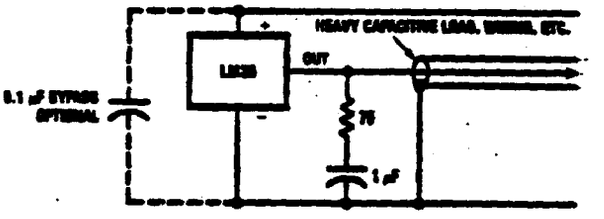
** TO-92 package glued and leads soldered to 1" square of 1/16" printed circuit board with 2 oz. foil or similar.

Typical Applications (Continued)



TL/H/5516–18

FIGURE 3. LM35 with Decoupling from Capacitive Load



TL/H/5516–20

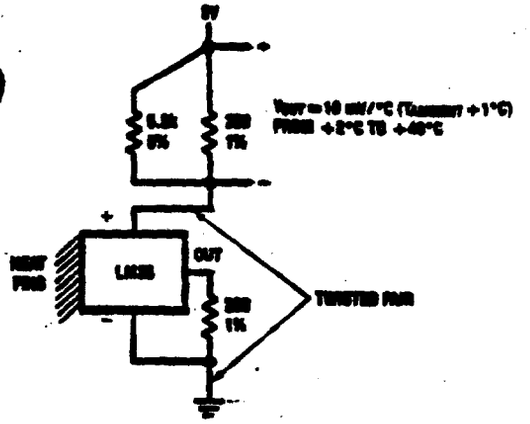
FIGURE 4. LM35 with R-C Damper

CAPACITIVE LOADS

Like most micropower circuits, the LM35 has a limited ability to drive heavy capacitive loads. The LM35 by itself is able to drive 50 pf without special precautions. If heavier loads are anticipated, it is easy to isolate or decouple the load with a resistor; see *Figure 3*. Or you can improve the tolerance of capacitance with a series R-C damper from output to ground; see *Figure 4*.

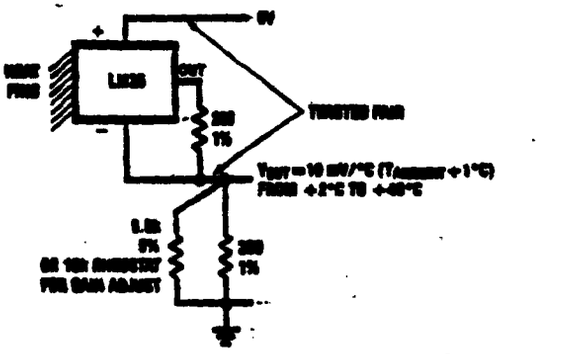
When the LM35 is applied with a 200Ω load resistor as shown in *Figure 5*, *6*, or *8*, it is relatively immune to wiring capacitance because the capacitance forms a bypass from ground to input, not on the output. However, as with any linear circuit connected to wires in a hostile environment, its performance can be affected adversely by intense electromagnetic sources such as relays, radio transmitters, motors with arcing brushes, SCR transients, etc, as its wiring can act as a receiving antenna and its internal junctions can act as rectifiers. For best results in such cases, a bypass capacitor from V_{IN} to ground and a series R-C damper such as 75Ω in series with 0.2 or 1 μF from output to ground are often useful. These are shown in *Figures 13, 14,* and *16*.

Typical Applications (Continued)



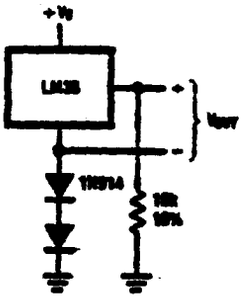
TL/H/5516–5

FIGURE 5. Two-Wire Remote Temperature Sensor (Grounded Sensor)



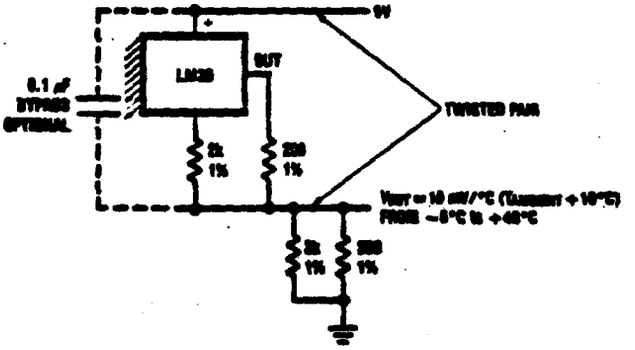
TL/H/5516–6

FIGURE 6. Two-Wire Remote Temperature Sensor (Output Referred to Ground)



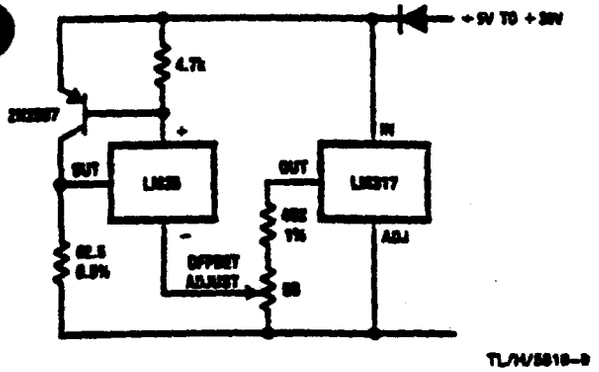
TL/H/5516–7

FIGURE 7. Temperature Sensor, Single Supply, −55° to +150°C



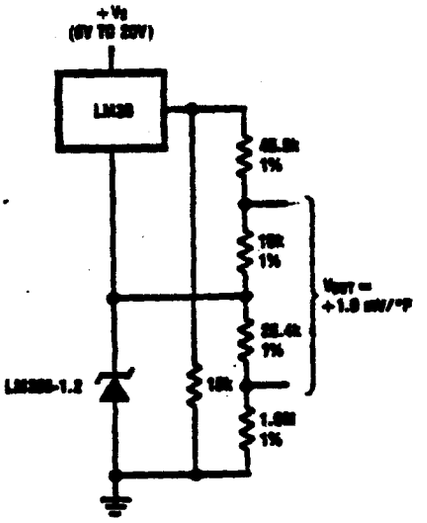
TL/H/5516–8

FIGURE 8. Two-Wire Remote Temperature Sensor (Output Referred to Ground)



TL/H/5516–9

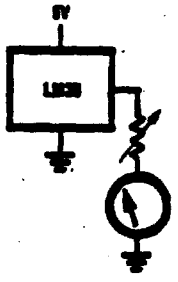
FIGURE 9. 4-To-20 mA Current Source (0°C to + 100°C)



TL/H/5516–10

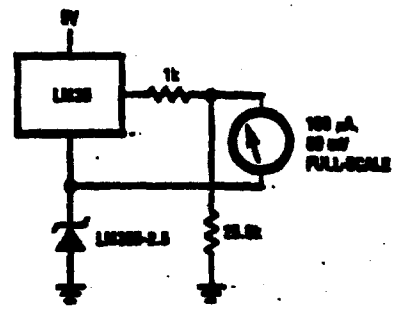
FIGURE 10. Fahrenheit Thermometer

Typical Applications (Continued)



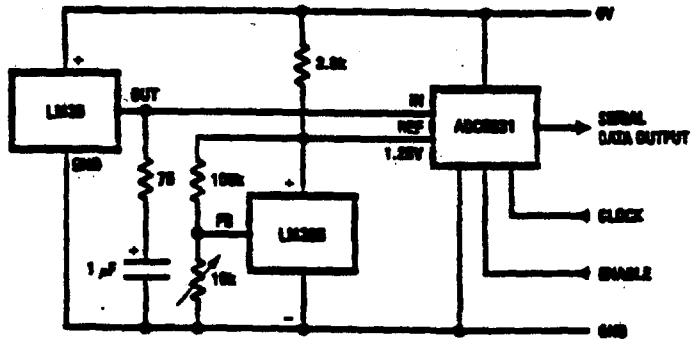
TL/H/5516–11

FIGURE 11. Centigrade Thermometer (Analog Meter)



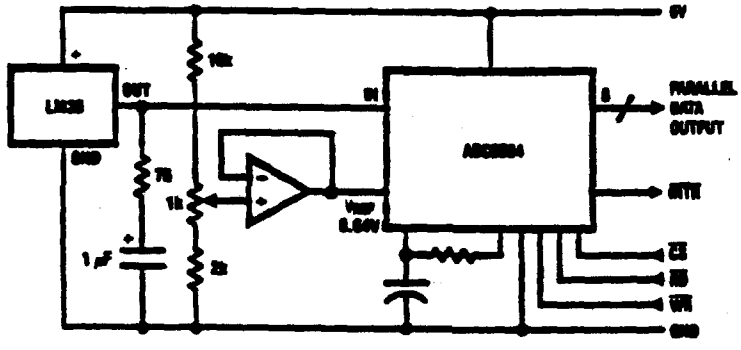
TL/H/5516–12

FIGURE 12. Expanded Scale Thermometer (50° to 80° Fahrenheit, for Example Shown)



TL/H/5516–13

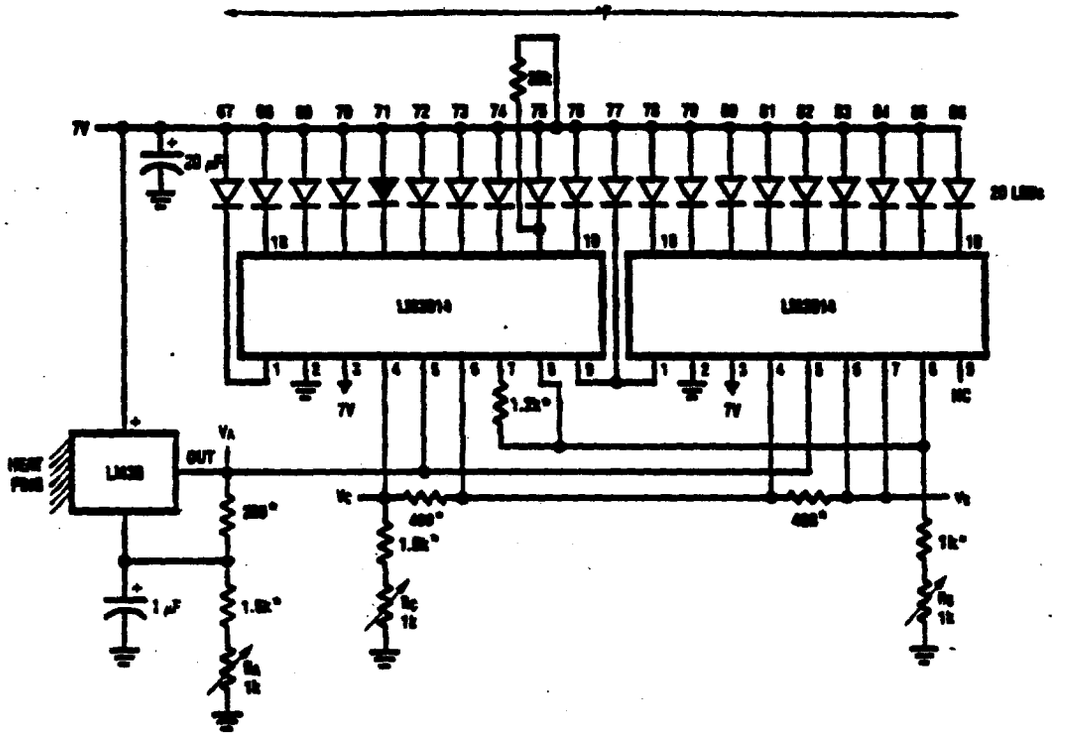
FIGURE 13. Temperature To Digital Converter (Serial Output) (+ 128°C Full Scale)



TL/H/5516–14

FIGURE 14. Temperature To Digital Converter (Parallel TRI-STATE® Outputs for Standard Data Bus to μP Interface) (128°C Full Scale)

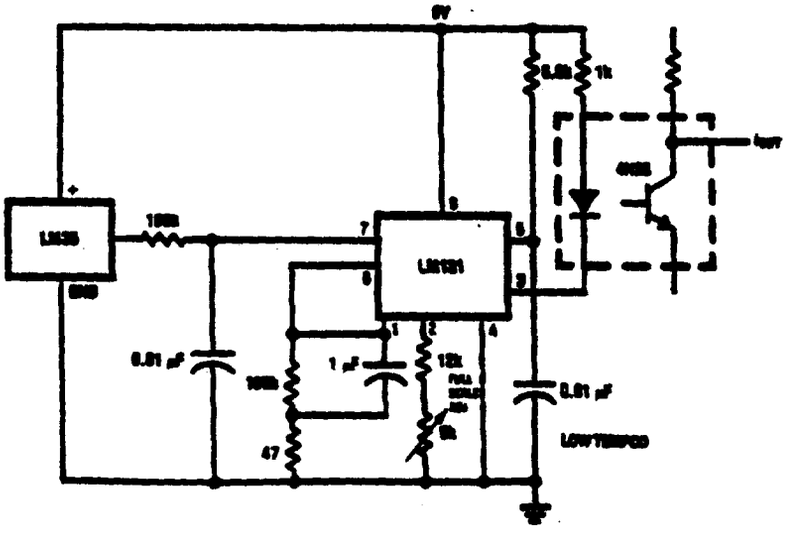
Typical Applications (Continued)



TL/H/5515–16

*=1% or 2% film resistor
-Trim R_B for V_B=3.075V
-Trim R_C for V_C=1.955V
-Trim R_A for V_A=0.075V + 100mV/°C × $T_{ambient}$
-Example, V_A=2.275V at 22°C

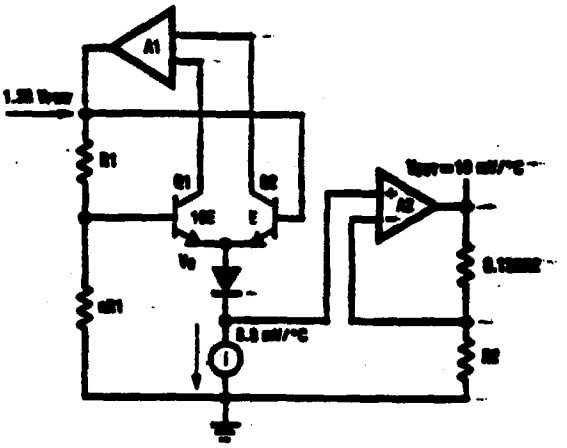
FIGURE 15. Bar-Graph Temperature Display (Dot Mode)



TL/H/5515–15

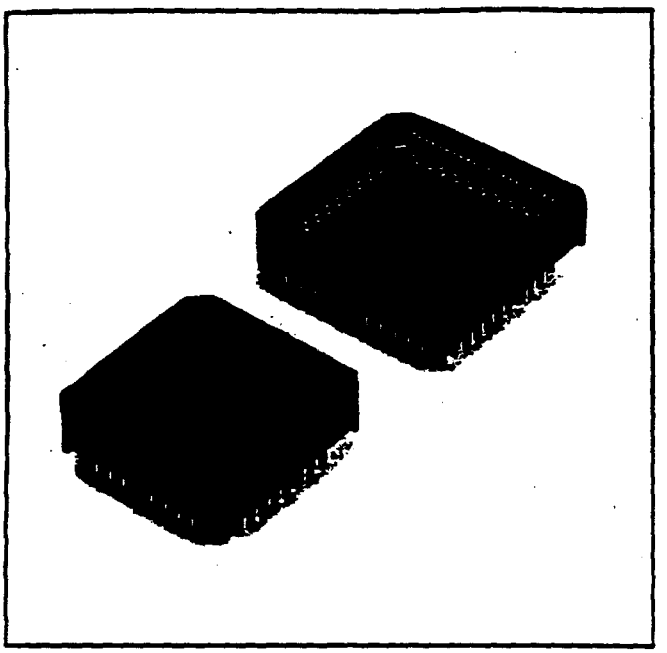
FIGURE 16. LM35 With Voltage-To-Frequency Converter And Isolated Output (2°C to +150°C; 20 Hz to 1500 Hz)

Block Diagram



TL/H/5510–21

Open Shroud PLCC



- Accepts JEDEC PLCCs in Registration MO-047 AA-AH
- Visual polarizing, PLCC corner, pin "1" I.D.
- Underside orientation to P.C. board plastic post feature option
- Contacts have discrete egress from socket bottom, prevents solder bridging and up-contact wicking
- Floor of socket has drainage holes for post solder cleaning
- Industry top extraction feature for package removal, some sizes
- Side package extraction feature, some sizes
- Stainless steel clip available for locking in PLCC package for high vibration — mechanical shock application

Physical

Insulation Material: Glass Fortified Polyethylene Teraphthalate
Flammability Rating: UL 94 V-0
Color: Black
Contact Material: Ni Ag Alloy 770
Contact Plating: 90/10 Bright SnPb Nickel Underplate

Electrical

Current Rating: 1 A
Insulation Resistance: $> 1 \times 10^{12}\ \Omega$
Withstanding Voltage: 1000 Vrms at Sea Level

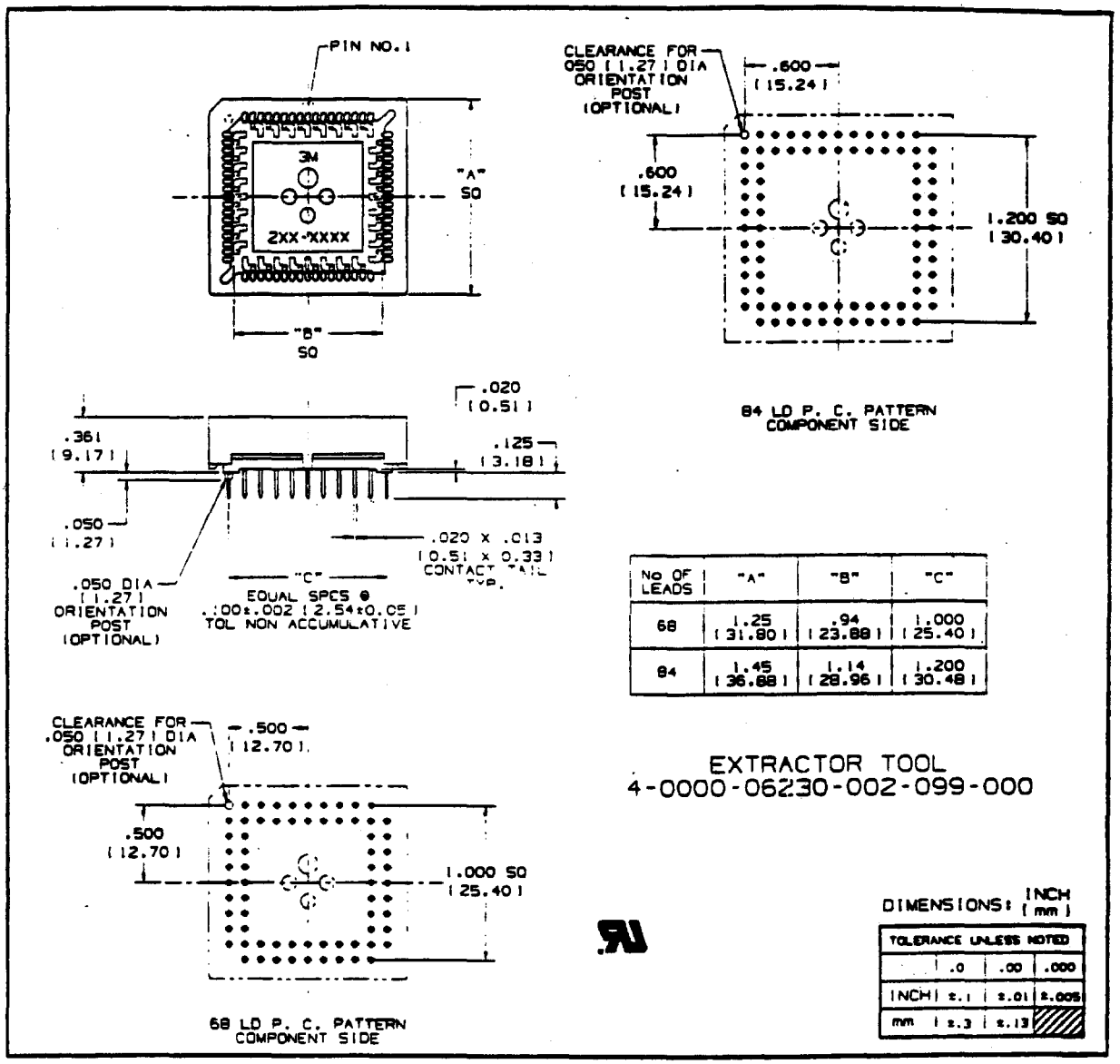
Environmental

Temperature Rating: – 67°F to + 221°F (– 55°C to + 105°C)

UL File No.: E68080

3M Electronic Products Division
PO Box 2963
Austin, TX 78769-2963

Open Shroud PLCC



No OF LEADS	"A"	"B"	"C"
68	1.25 [31.80]	.94 [23.88]	1.000 [25.40]
84	1.45 [36.88]	1.14 [28.96]	1.200 [30.48]

ORDER INFORMATION

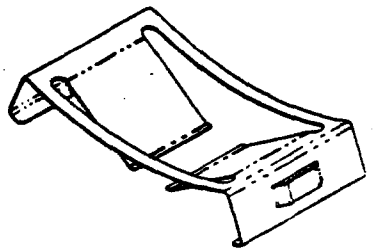
No OF LEADS	SOCKET PART NUMBER	OPTIONAL RETAINER CLIP
68	2-0068-06234-05X-038-077	2-0068-06234-007-080-000
84	2-0084-06235-05X-038-077	2-0084-06235-007-080-000

X = 5 (WITH ORIENTATION POST)
= 6 (WITHOUT ORIENTATION POST)

Open Shroud PLCC

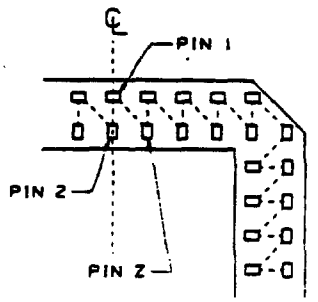
SHEET 3 OF 3.

TS/0326/02



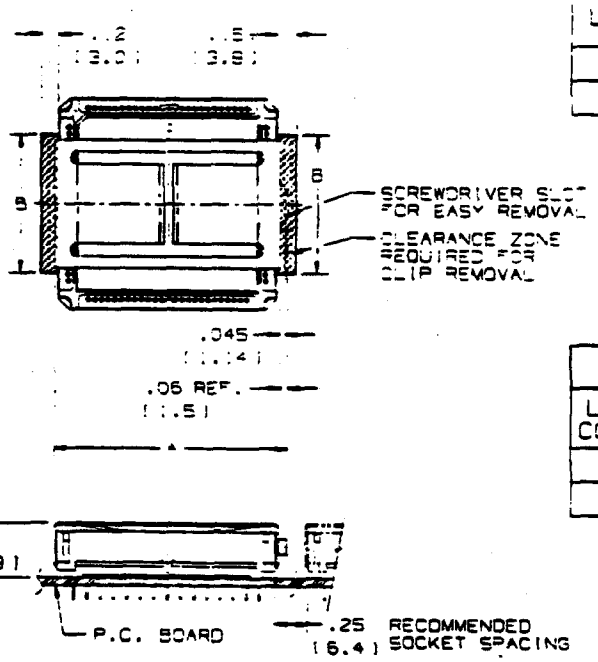
RETAINER CLIP

PIN OUT LOGIC
(VIEW FROM COMPONENT SIDE)



68 & 84
LEAD PLCC
(VIEW FROM COMPONENT SIDE)

LEAD COUNT	PIN NUMBERS Z
68	68
84	84



RETAINER CLIP		
LEAD COUNT	"A"	"B"
68	1.35 [34.26]	1.02 [25.89]
84	1.56 [39.59]	1.08 [27.41]

DIMENSIONS: INCH [mm]

TOLERANCE UNLESS NOTED			
	.0	.00	.000
INCH	±.1	±.01	±.005
mm	±.3	±.13	

Positive Adjustable Regulator

FEATURES

- *Guaranteed* 1% Output Voltage Tolerance
- *Guaranteed* max. 0.01%/V Line Regulation
- *Guaranteed* max. 0.3% Load Regulation
- Min. 1.5A Output Current
- 100% Burn-in in Thermal Overload

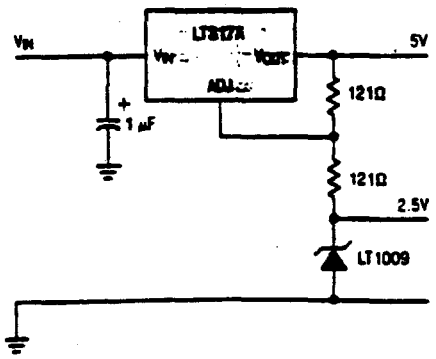
APPLICATIONS

- Wide Range Power Supplies
- Constant Current Supplies
- Voltage Programmable Supplies

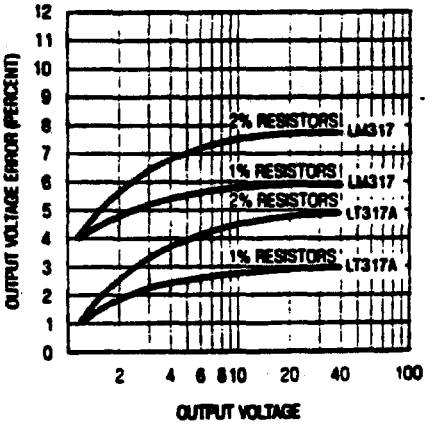
DESCRIPTION

The LT117A Series are 3-terminal positive adjustable voltage regulators which offer improved performance over earlier devices. A major feature of the LT117A is the output voltage tolerance is guaranteed at a maximum of ± 1%, allowing an overall power supply tolerance to be better than 3% using inexpensive 1% resistors. Line and load regulation performance has been improved as well. Additionally, the LT117A reference voltage is guaranteed not to exceed 2% when operating over the full load, line and power dissipation conditions. The LT117A adjustable regulators offer an improved solution for all positive voltage regulator requirements with load currents up to 1.5 amps.

Regulator with Reference



Output Voltage Error



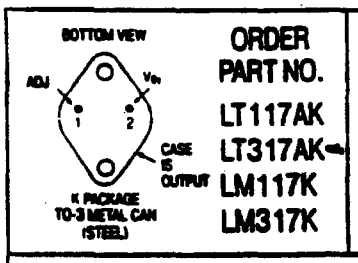
ABSOLUTE MAXIMUM RATINGS

Power Dissipation ..... Internally Limited
Input to Output Voltage Differential 40V
Operating Junction Temperature Range
LT117A/LM117 ..... −55°C to 150°C
LT317A/LM317 ..... 0°C to 125°C
Storage Temperature Range
LT117A/LM117 ..... −65°C to 150°C
LT317A/LM317 ..... −65°C to 150°C
Lead Temperature (Soldering, 10 sec.)..... 300°C

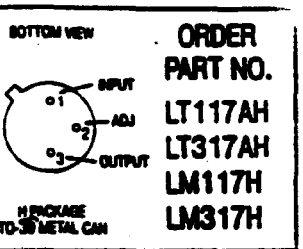
PRECONDITIONING:

100% THERMAL LIMIT BURN-IN

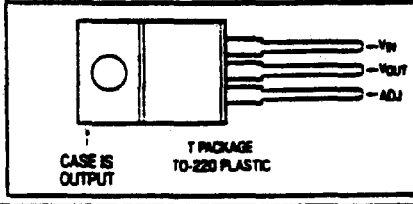
PACKAGE/ORDER INFORMATION



ORDER PART NO.
LT117AK
LT317AK
LM117K
LM317K



ORDER PART NO.
LT117AH
LT317AH
LM117H
LM317H



ORDER PART NO.
LT317AT
LM317T

ELECTRICAL CHARACTERISTICS (See Note 1) LT117A/LM117

SYMBOL	PARAMETER	CONDITIONS		LT117A MIN	LT117A TYP	LT117A MAX	LM117 MIN	LM117 TYP	LM117 MAX	UNITS
V_{REF}	Reference Voltage	I_{OUT} = 10mA T_J = 25°C		1.238	1.250	1.262				V
		3V ≤ (V_{IN} − V_{OUT}) ≤ 40V 10mA ≤ I_{OUT} ≤ I_{max}, P ≤ P_{max}	●	1.225	1.250	1.270	1.20	1.25	1.30	V
$\frac{\Delta V_{OUT}}{\Delta V_{IN}}$	Line Regulation	3V ≤ (V_{IN} − V_{OUT}) ≤ 40V. (See Note 2)			0.005	0.01		0.01	0.02	%/V
			●		0.01	0.02		0.02	0.05	%
$\frac{\Delta V_{OUT}}{\Delta I_{OUT}}$	Load Regulation	10mA ≤ I_{OUT} ≤ I_{max} (See Note 2) V_{OUT} ≤ 5V V_{OUT} ≥ 5V			5 0.1	15 0.3		5 0.1	15 0.3	mV %
		V_{OUT} ≤ 5V	●		20	50		20	50	mV
		V_{OUT} ≥ 5V	●		0.3	1		0.3	1	%
	Thermal Regulation	T_A = 25°C, 20msec Pulse			0.002	0.02		0.03	0.07	%/W
	Ripple Rejection	V_{OUT} = 10V, f = 120Hz C_{ADJ} = 0	●		65			65		dB
		C_{ADJ} = 10μF	●	66	80		66	80		dB
I_{ADJ}	Adjust Pin Current		●		50	100		50	100	μA
ΔI_{ADJ}	Adjust Pin Current Change	10mA ≤ I_{OUT} ≤ I_{max} 2.5V ≤ (V_{IN} − V_{OUT}) ≤ 40V	●		0.2	5		0.2	5	μA
I_{min}	Minimum Load Current	(V_{IN} − V_{OUT}) = 40V	●		3.5	5		3.5	5	mA
	Current Limit	(V_{IN} − V_{OUT}) ≤ 15V K Package H Package	● ●	1.5 0.5	2.2 0.8		1.5 0.5	2.2 0.8		A A
		(V_{IN} − V_{OUT}) = 40V, T_j = 25°C K Package H Package		0.3 0.15	0.5 0.2		0.3 0.15	0.4 0.2		A A
$\frac{\Delta V_{OUT}}{\Delta Temp}$	Temperature Stability	−55°C ≤ T_j ≤ +150°C			1	2		1		%
$\frac{\Delta V_{OUT}}{\Delta Time}$	Long Term Stability	T_A = 125°C			0.3	1		0.3	1	%
e_n	RMS Output Noise (% of V_{OUT})	T_A = 25°C, 10Hz ≤ f ≤ 10kHz			0.001			0.001		%
θ_{jc}	Thermal Resistance Junction to Case	H Package K Package			12 2.3	15 3		12 2.3	15 3	

ELECTRICAL CHARACTERISTICS (See Note 1) LT317A/LM317

SYMBOL	PARAMETER	CONDITIONS		LT317A MIN	LT317A TYP	LT317A MAX	LM317 MIN	LM317 TYP	LM317 MAX	UNITS
V_{REF}	Reference Voltage	I_{OUT} = 10mA T_J = 25°C		1.238	1.250	1.262				V
		3V ≤ (V_{IN} − V_O) ≤ 40V 10mA ≤ I_{OUT} ≤ I_{MAX}, P ≤ P_{MAX}	●	1.225	1.250	1.270	1.20	1.25	1.30	V
$\frac{\Delta V_{OUT}}{\Delta V_{IN}}$	Line Regulation	3V ≤ (V_{IN} − V_{OUT}) ≤ 40V, (See Note 2)			0.005	0.01		0.01	0.04	%/V
			●		0.01	0.02		0.02	0.07	%/V
$\frac{\Delta V_{OUT}}{\Delta I_{OUT}}$	Load Regulation	10mA ≤ I_{OUT} ≤ I_{MAX}, (See Note 2) V_O ≤ 5V V_O ≥ 5V			5 0.1	25 0.5		5 0.1	25 0.5	mV %
		V_O ≤ 5V V_O ≥ 5V	● ●		20 0.3	50 1		20 0.3	70 1.5	mV %
	Thermal Regulation	T_A = 25°C, 20msec Pulse	●		0.002	0.02		0.04	0.07	%/W
	Ripple Rejection	V_O = 10V, f = 120Hz C_{ADJ} = 0			65			65		dB
		C_{ADJ} = 10μF		66	80		66	80		dB
I_{ADJ}	Adjust Pin Current				50	100		50	100	μA
ΔI_{ADJ}	Adjust Pin Current Change	10mA ≤ I_{OUT} ≤ I_{MAX} 2.5V ≤ (V_{IN} − V_{OUT}) ≤ 40V	●		0.2	5		0.2	5	μA
I_{MIN}	Minimum Load Current	(V_{IN} − V_{OUT}) = 40V	●		3.5	10		3.5	10	mA
	Current Limit	(V_{IN} − V_{OUT}) ≤ 15V K and T Package H Package	● ●	1.5 0.5	2.2 0.8		1.5 0.5	2.2 0.8		A A
		(V_{IN} − V_{OUT}) = 40V, T_J = 25°C K and T Package H Package		0.15 0.075	0.4 0.2		0.15 0.075	0.4 0.2		A A
$\frac{\Delta V_{OUT}}{\Delta Temp}$	Temperature Stability	0°C ≤ T_J ≤ 125°C			1	2		1		%
$\frac{\Delta V_{OUT}}{\Delta time}$	Long Term Stability	T_A = 125°C			0.3	1		0.3	1	%
e_n	RMS Output Noise (% of V_{OUT})	T_A = 25°C, 10Hz ≤ f ≤ 10kHz			0.001			0.001		%
θ_{JC}	Thermal Resistance Junction to Case	H Package K Package T Package			12 2.3 4	15 3 5		12 2.3 4	15 3	°C/W °C/W °C/W

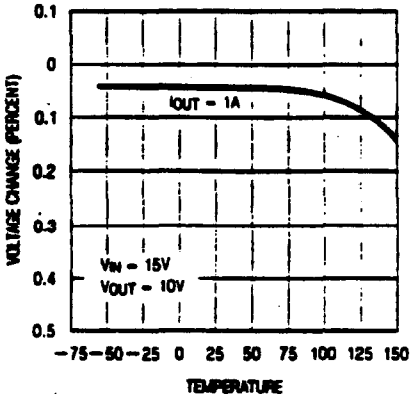
The ● denotes the specifications which apply over the full operating temperature range.

Note 1: Unless otherwise specified, these specifications apply for V_{IN} − V_{OUT} = 5V; and I_{OUT} = 0.1A for the TO-39 and I_{OUT} = 0.5A for the TO-3 and TO-220 packages. Although power dissipation is internally limited, these specifications are applicable for power dissipations of 2W for the TO-39, and 20W for the TO-3 and TO-220. I_{MAX} is 1.5A for the TO-3 and TO-220 packages and 0.5A for the TO-39.

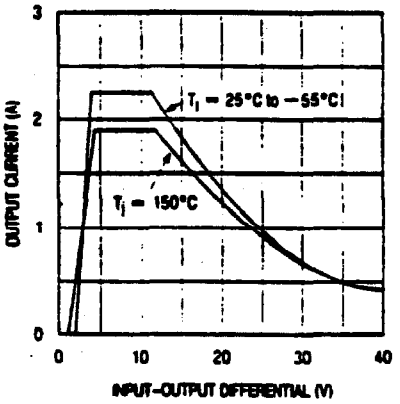
Note 2: Regulation is measured at constant junction temperature, using pulse testing with a low duty cycle. Changes in output voltage due to heating effects are covered under the specification for thermal regulation. Load regulation is measured on the output pin at a point ½" below the base of the K and H package and at the junction of the wide and narrow portion of the lead on the T package.

TYPICAL PERFORMANCE CHARACTERISTICS

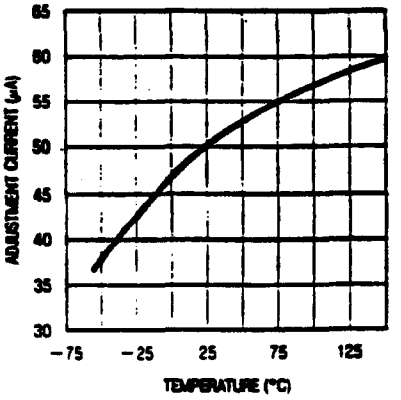
Load Regulation



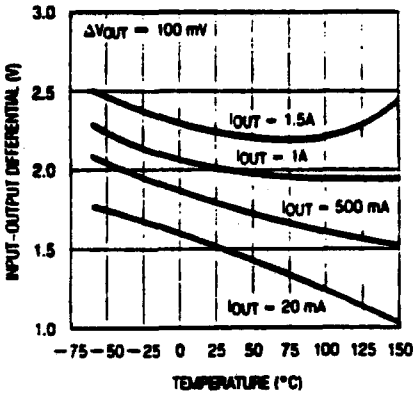
Current Limit (TO-3 and TO-220 Package)



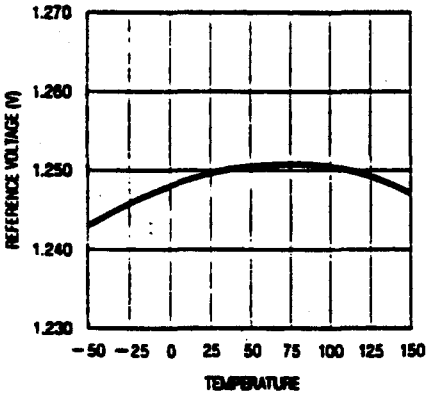
Adjustment Current



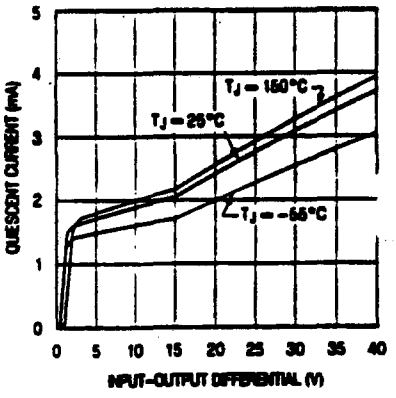
Dropout Voltage



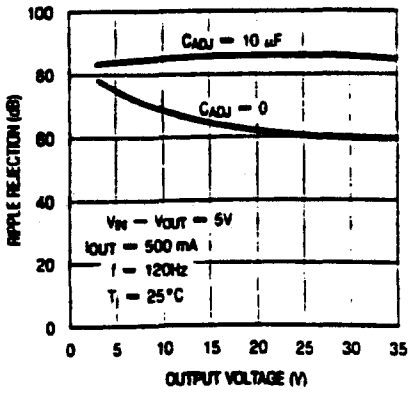
Temperature Stability



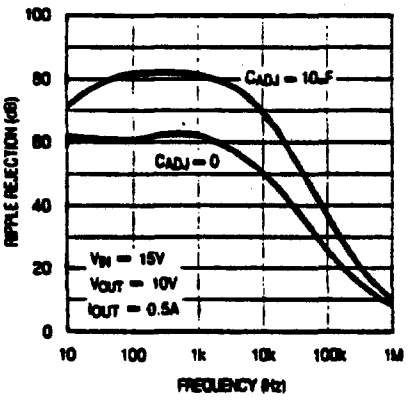
Minimum Operating Current



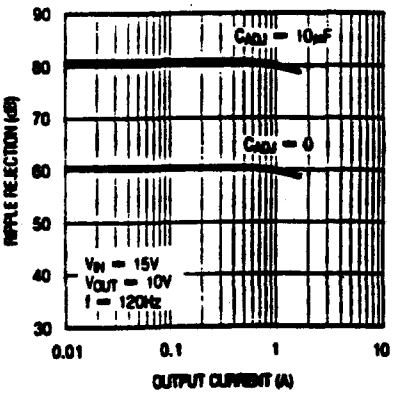
Ripple Rejection



Ripple Rejection (dB)

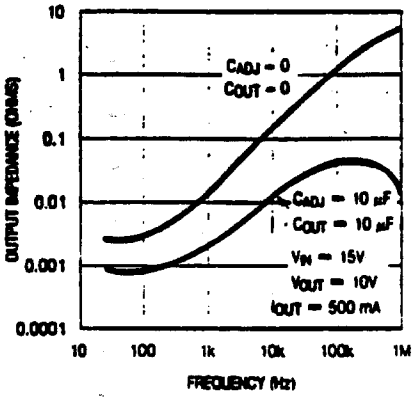


Ripple Rejection

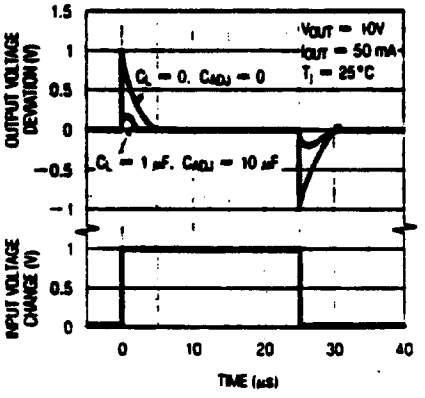


TYPICAL PERFORMANCE CHARACTERISTICS

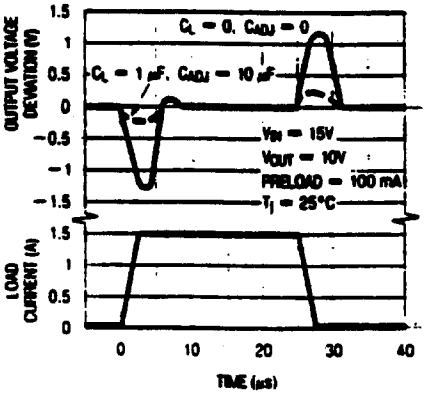
Output Impedance



Line Transient Response



Load Transient Response



APPLICATIONS INFORMATION

General: The LT117A develops a 1.25V reference voltage between the output and the adjustable terminal (see Figure 1). By placing a resistor, R1, between these two terminals, a constant current is caused to flow through R1 and down through R2 to set the overall output voltage. Normally this current is the specified minimum load current of 5mA or 10mA.

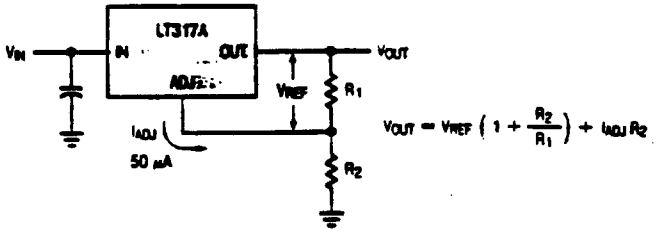


Figure 1

Because I_{ADJ} is very small and constant when compared with the current through R1, it represents a small error and can usually be ignored.

It is easily seen from the above equation, that even if the resistors were of exact value, the accuracy of the output is limited by the accuracy of V_{REF}. Earlier adjustable regulators had a reference tolerance of ±4%. This tolerance is dangerously close to the ±5% supply tolerance required in many logic and analog systems. Further, many 1% resistors can drift 0.01%/°C adding another 1% to the output voltage tolerance.

For example, using 2% resistors and ±4% tolerance for V_{REF}, calculations will show that the expected range of a 5V regulator design would be $4.66V < V_{OUT} < 5.36V$ or approximately ±7%. If the same example were used for a 15V regulator, the expected tolerance would be ±8%. With these results most applications require some method of trimming, usually a trim pot. This solution is both expensive and not conducive to volume production.

One of the enhancements of Linear Technology's adjustable regulators over existing devices is tightened initial tolerance. This allows relatively inexpensive 1% or 2% film resistors to be used for R1 and R2 while setting output voltage within an acceptable tolerance range.

With a guaranteed 1% reference, a 5V power supply design, using ±2% resistors, would have a worst case manufacturing tolerance of ±4%. If 1% resistors were used, the tolerance would drop to ±2.5%. A plot of the worst case output voltage tolerance as a function of resistor tolerance is shown on the front page.

4

For convenience, a table of standard 1% resistor values is shown below.

Table of ½% and 1% Standard Resistance Values

1.00	1.47	2.15	3.16	4.64	6.81
1.02	1.50	2.21	3.24	4.75	6.98
1.05	1.54	2.26	3.32	4.87	7.15
1.07	1.58	2.32	3.40	4.99	7.32
1.10	1.62	2.37	3.48	5.11	7.50
1.13	1.65	2.43	3.57	5.23	7.68
1.15	1.69	2.49	3.65	5.36	7.87
1.18	1.74	2.55	3.74	5.49	8.06
1.21	1.78	2.61	3.83	5.62	8.25
1.24	1.82	2.67	3.92	5.76	8.45
1.27	1.87	2.74	4.02	5.90	8.66
1.30	1.91	2.80	4.12	6.04	8.87
1.33	1.96	2.87	4.22	6.19	9.09
1.37	2.00	2.94	4.32	6.34	9.31
1.40	2.05	3.01	4.42	6.49	9.53
1.43	2.10	3.09	4.53	6.65	9.76

Standard Resistance Values are obtained from the Decade Table by multiplying by multiples of 10. As an example, 1.21 can represent 1.21Ω, 12.1Ω, 121Ω, 1.21KΩ etc.

Bypass Capacitors: Input bypassing using a 1μF tantalum or 25μF electrolytic is recommended when the input filter capacitors are more than 5 inches from the device. Improved ripple rejection (80 dB) can be accomplished by adding a 10μF capacitor from the adjust pin to ground. Increasing the size of the capacitor to 20μF will help ripple rejection at low output voltage since the reactance of this capacitor should be small compared to the voltage setting resistor, R2. For improved AC transient response and to prevent the possibility of oscillation due to unknown reactive load, a 1μF capacitor is also recommended at the output. Because of their low impedance at high frequencies, the best type of capacitor to use is solid tantalum.

Protection Diodes: The LT117A/317A do not require a protection diode from the adjustment terminal to the output (see Figure 2). Improved internal circuitry eliminates the need for this diode when the adjustment pin is bypassed with a capacitor to improve ripple rejection.

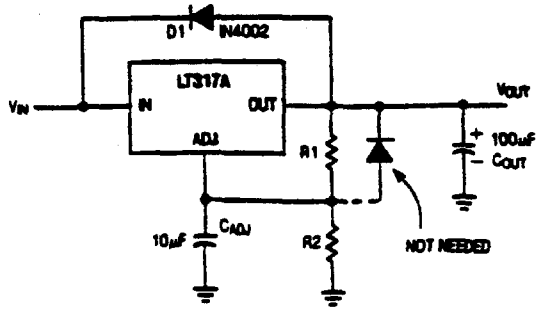


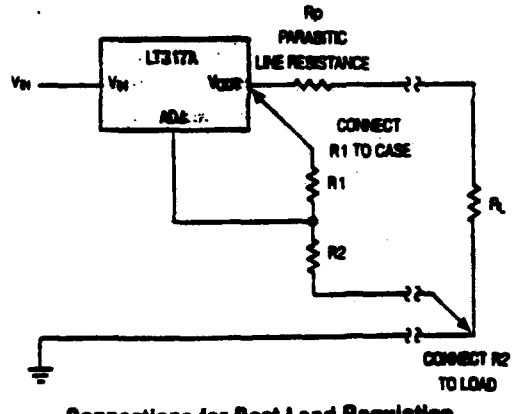
Figure 2

If a very large output capacitor is used, such as a 100μF shown in Figure 2, the regulator could be damaged or destroyed if the input is accidentally shorted to ground or crowbarred. This is due to the output capacitor discharging into the output terminal of the regulator. To prevent damage a diode D1 is recommended to safely discharge the capacitor.

Load Regulation: Because the LT117A is a three-terminal device, it is not possible to provide true remote load sensing. Load regulation will be limited by the resistance of the wire connecting the regulator to the load. For the data sheet specification, regulation is measured at the bottom of the package. Negative side sensing is a true Kelvin connection, with the bottom of the output divider returned to the negative side of the load. Although it may not be immediately obvious, best load regulation is obtained when the top of the divider is connected *directly* to the case *not to the load.* This is illustrated in *Figure 3.* If R1 were connected to the load, the effective resistance between the regulator and the load would be

$$R_p \times \left(\frac{R2 + R1}{R1}\right), \quad R_p = \text{Parasitic Line Resistance.}$$

Connected as shown, R_p is not multiplied by the divider ratio. R_p is about 0.004Ω per foot using 16 guage wire. This translates to 4mV/ft at 1A load current, so it important to keep the positive lead between regulator and load as short as possible.

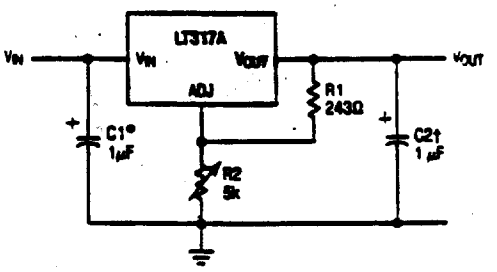


Connections for Best Load Regulation

Figure 3

TYPICAL APPLICATIONS

1.2V–25V Adjustable Regulator

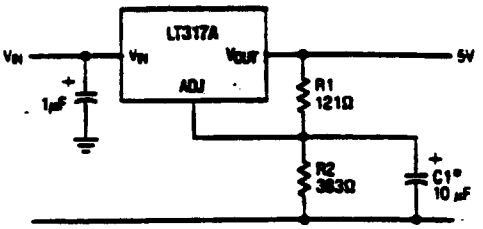


Optional — improves transient response

Needed if device is far from filter capacitors

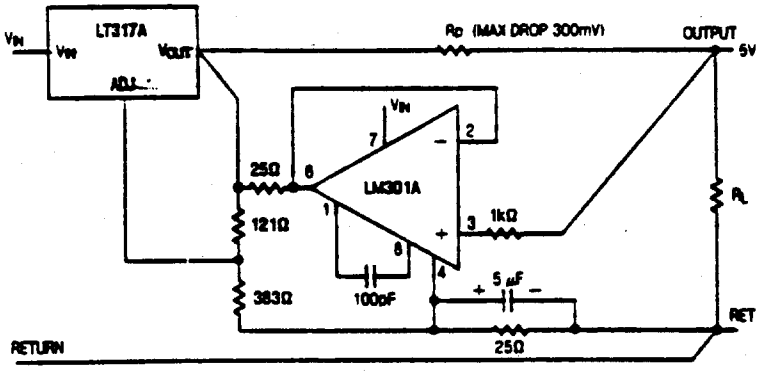
$$V_{OUT} = 1.25V \left(1 + \frac{R2}{R1}\right)$$

Improving Ripple Rejection

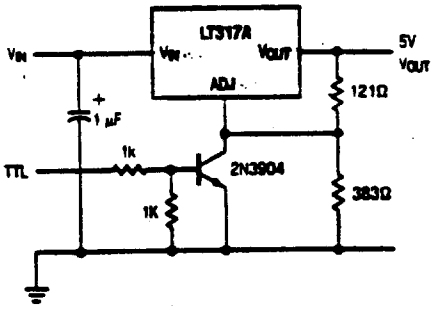


*C1 IMPROVES RIPPLE REJECTION
X_C SHOULD BE SMALL COMPARED TO R2

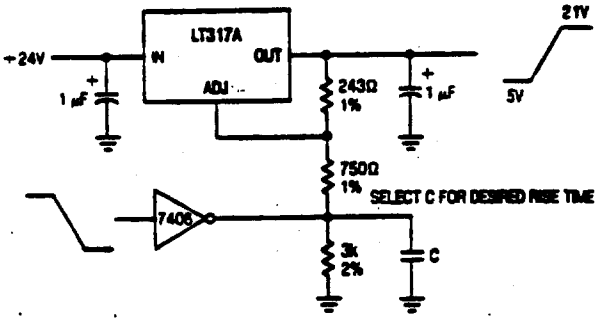
Remote Sensing



5V Regulator with Shut Down

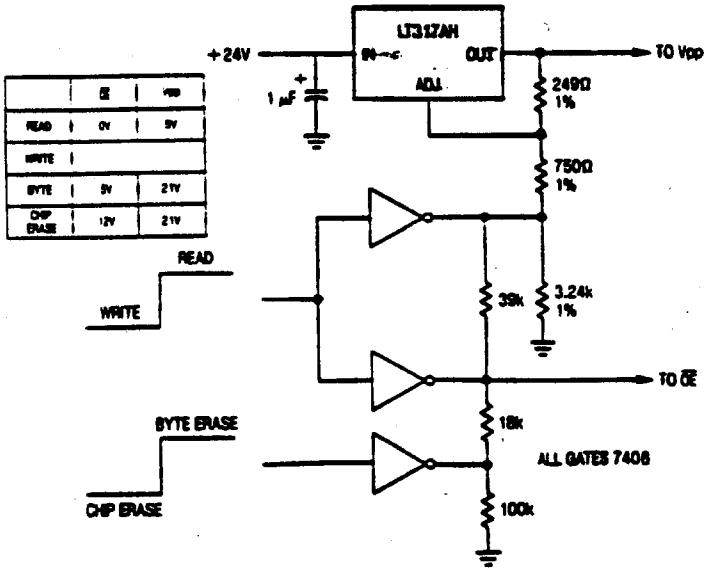


21V Programming Supply for UV PROM/EEROM

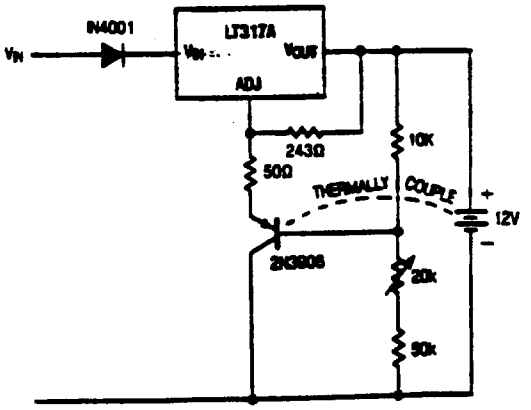


2816 EEPROM Supply Programmer for Read/Write Control

	OE	VPP
READ	0V	5V
WRITE		
BYTE	5V	21V
CHIP ERASE	12V	21V



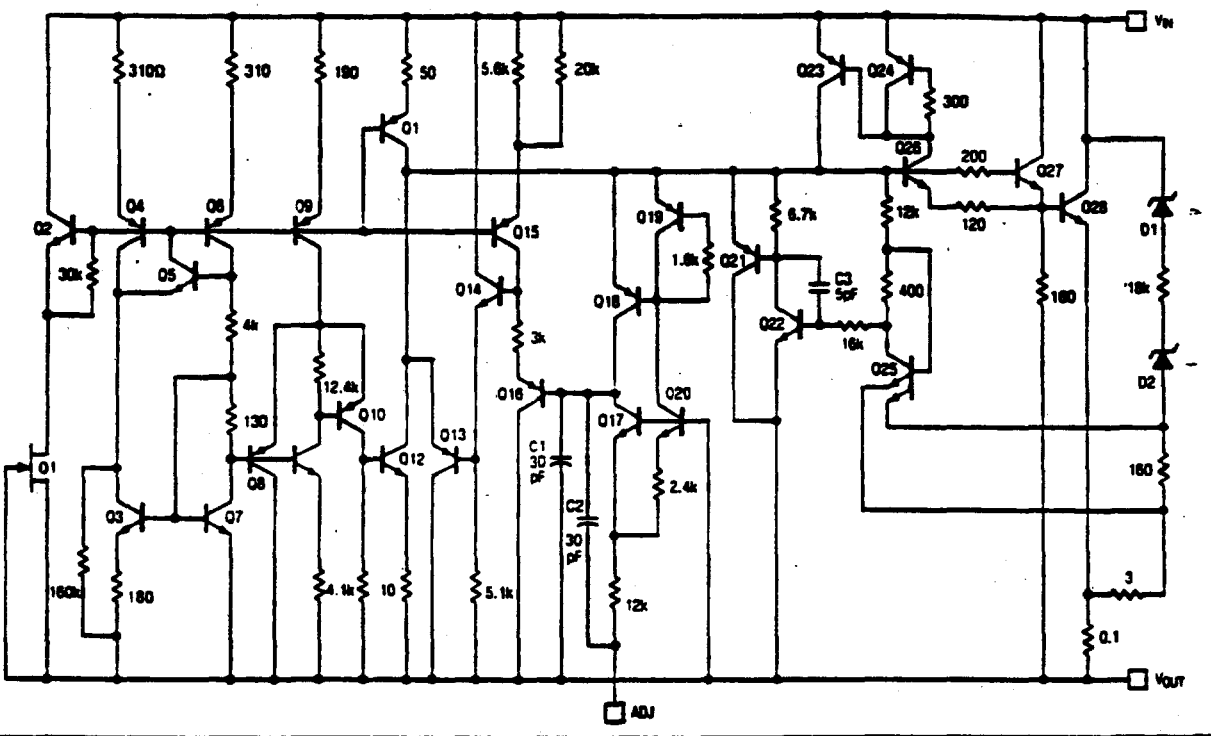
Temperature Compensated Lead Acid Battery Charger



4

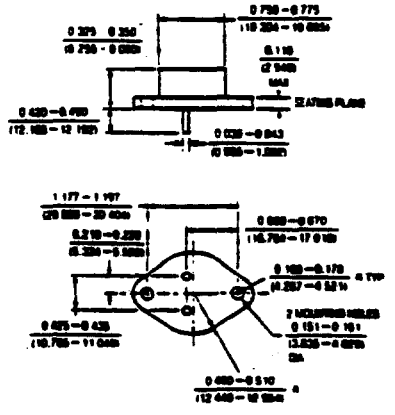
SCHEMATIC DIAGRAM

LT117A/LT317A



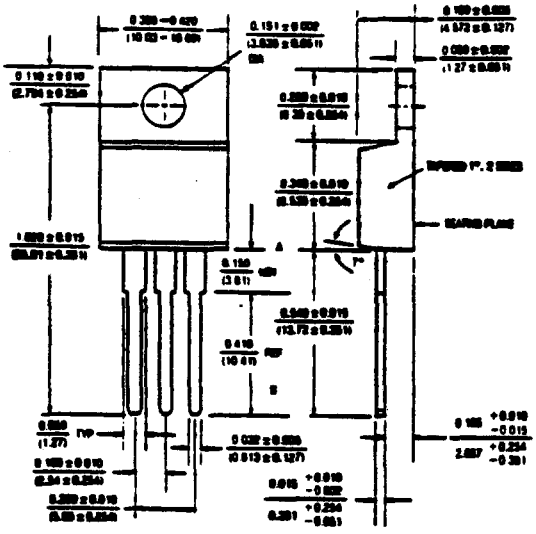
PACKAGE DESCRIPTION

K Package TO-3 STEEL Metal Can



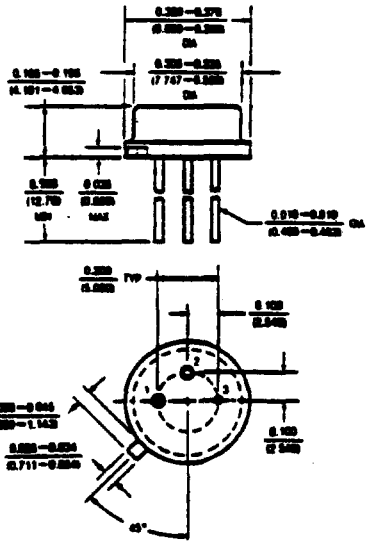
	T_j max.	θ_{JA}	θ_{JC}
117A 117	150°C	35°C/W	3°C/W
317A 317	125°C	35°C/W	3°C/W

T Package TO-220 Plastic



	T_j max.	θ_{JA}	θ_{JC}
317A 317	125°C	50°C/W	5°C/W

H Package 3-Lead Metal Can



	T_j max.	θ_{JA}	θ_{JC}
117A 117	150°C	150°C/W	15°C/W
317A 317	125°C	150°C/W	15°C

FDC TYPE

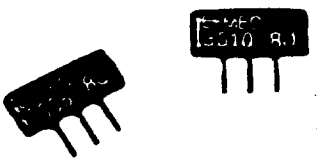
■ SPECIFICATIONS

Part Number	Impedance	Delay Time	Rise Time	3dB Bandwidth
[illegible]	50Ω ± 10%	0.1 ns ± 50 ps	150 ps or less	2.5 GHz or more
[illegible]		0.2 ns ± 50 ps	150 ps or less	2.0 GHz or more
[illegible]		0.3 ns ± 50 ps	150 ps or less	2.0 GHz or more
[illegible]		0.4 ns ± 50 ps	150 ps or less	2.0 GHz or more
[illegible]		0.5 ns ± 50 ps	200 ps or less	1.5 GHz or more
[illegible]0605		0.6 ns ± 50 ps	200 ps or less	1.5 GHz or more
FDC-0705		0.7 ns ± 50 ps	200 ps or less	1.5 GHz or more
FDC-0805		0.8 ns ± 50 ps	200 ps or less	1.0 GHz or more
FDC-0905		0.9 ns ± 50 ps	200 ps or less	1.0 GHz or more
FDC-1005		1.0 ns ± 50 ps	200 ps or less	1.0 GHz or more
FDC-1105		1.1 ns ± 50 ps	250 ps or less	900 MHz or more
FDC-1205		1.2 ns ± 50 ps	250 ps or less	900 MHz or more
FDC-1305		1.3 ns ± 50 ps	250 ps or less	900 MHz or more
FDC-1405		1.4 ns ± 50 ps	250 ps or less	900 MHz or more
FDC-1505		1.5 ns ± 50 ps	250 ps or less	900 MHz or more
FDC-1605		1.6 ns ± 50 ps	300 ps or less	800 MHz or more
FDC-1705		1.7 ns ± 50 ps	300 ps or less	800 MHz or more
FDC-1805		1.8 ns ± 50 ps	300 ps or less	800 MHz or more
FDC-1905		1.9 ns ± 50 ps	300 ps or less	800 MHz or more
FDC-2005		2.0 ns ± 50 ps	300 ps or less	800 MHz or more
FDC-2505		2.5 ns ± 0.20 ns	500 ps or less	450 MHz or more
FDC-3005		3.0 ns ± 0.20 ns	500 ps or less	450 MHz or more
FDC-3505		3.5 ns ± 0.25 ns	600 ps or less	400 MHz or more
FDC-4005		4.0 ns ± 0.30 ns	700 ps or less	350 MHz or more
FDC-4505		4.5 ns ± 0.30 ns	700 ps or less	350 MHz or more
FDC-5005		5.0 ns ± 0.30 ns	700 ps or less	350 MHz or more
FDC-0510	100Ω ± 10%	0.5 ns ± 0.10 ns	300 ps or less	800 MHz or more
FDC-1010		1.0 ns ± 0.10 ns	300 ps or less	800 MHz or more
FDC-1510		1.5 ns ± 0.15 ns	400 ps or less	550 MHz or more
FDC-2010		2.0 ns ± 0.20 ns	400 ps or less	550 MHz or more
FDC-2510		2.5 ns ± 0.20 ns	500 ps or less	450 MHz or more
FDC-3010		3.0 ns ± 0.20 ns	500 ps or less	450 MHz or more
FDC-3510		3.5 ns ± 0.25 ns	600 ps or less	400 MHz or more
FDC-4010		4.0 ns ± 0.30 ns	700 ps or less	350 MHz or more
FDC-4510		4.5 ns ± 0.30 ns	700 ps or less	350 MHz or more
FDC-5010		5.0 ns ± 0.30 ns	700 ps or less	350 MHz or more

FDC TYPE

Single-In Line Ultra High-Speed Fixed Delay Line

FDC type is an ultra high-speed compact designed fixed delay line with our high performance and hig density delay line elements housed in a single-in line package.
Units are 0.236-inch high, 0.488-inch long and 0.098-inch wide and suitable for high-density installation. It can be fully matched to high-speed logical elements such as ECL 100K, 10KH and 10K series and has extensive range of the applications including analog signal circuit.
The pin arrangement of the FDC type is the same (except height) as that of our fixed delay line FDD type so that the delay time can be adjusted at intervals of 500ps over the range of 0.5ns to 10ns.
(However, adjustment is possible at interval of 100ps over the range of 100ps to 2ns for the 50 ohms impedance systems.)

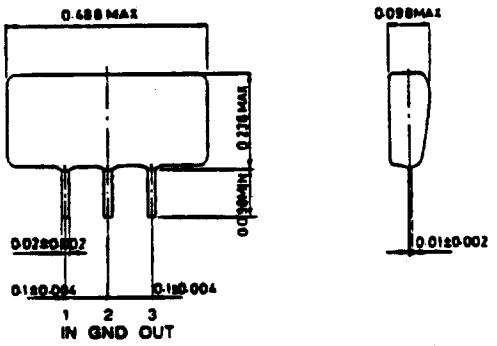


■ COMMON SPECIFICATIONS

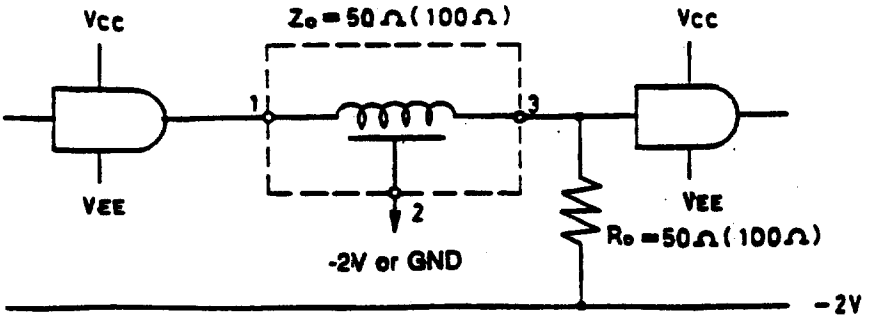
- Distortion of waveform: Over shoot / Pre shoot under ± 20%
- Delay time temp. coefficient: ±100ppm/°C
- Insulation resistance: DC 50V, 100MΩ
- Durable voltage: DC 50V, 1 minute
- Operating temperature range: –10°C ~ +80°C
- Storage temperature range: –40°C ~ +120°C

■ OUTER DIMENSIONS AND PIN ARRANGEMENT

Unit: inch



■ APPLICATION TO ECL



FUSES
SUBMINIATURE

PICO II™ Fast-Acting Type

ELECTRICAL CHARACTERISTICS:

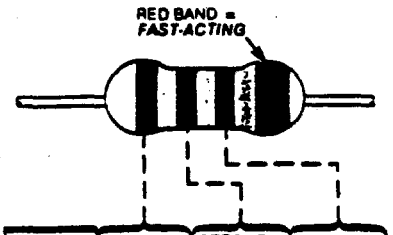
RATING	AMPERAGE	BLOW TIME
100%	1/16—15	4 hours, minimum
200%	1/16—10	5 seconds, maximum
200%	12—15	10 seconds, maximum

APPROVALS: Recognized under the Components Program of Underwriters Laboratories through 10 amperes. Certified by CSA through 7 amperes.

PATENTS: U.S. Patent #4,385,281.

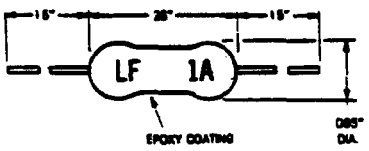
FUSES TO MIL SPEC: See Military Section.

OPTIONAL COLOR CODING: PICO II™ Fuses can be color-coded per IEC (International Electrotechnical Commission) Standards Publication 127. The first three bands indicate current rating in milliamperes. The fourth and wider band designates the time-current characteristics of the fuse (red is fast-acting).



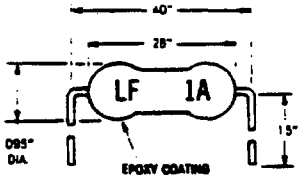
AMPERE RATING	FIRST SIGNIFICANT FIGURE	SECOND SIGNIFICANT FIGURE	MULTIPLIER
1/16	Blue	Red	Black
1/8	Brown	Red	Brown
1/4	Red	Green	Brown
3/8	Orange	Violet	Brown
1/2	Green	Black	Brown
3/4	Violet	Green	Brown
1	Brown	Black	Red
1-1/2	Brown	Green	Red
2	Red	Black	Red
2-1/2	Red	Green	Red
3	Orange	Black	Red
3-1/2	Orange	Green	Red
4	Yellow	Black	Red
5	Green	Black	Red
7	Violet	Black	Red
10	Brown	Black	Orange
12	Brown	Red	Orange
15	Brown	Green	Orange

251 000 Series
(Non color-coded)



NOTE: Leads are solder-coated copper, .025" diameter for 1/16 — 10A, .032" diameter for 12 — 15A.

252 000 Series
(Non color-coded)

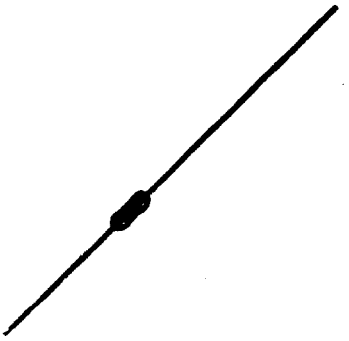


NOTE: LF and amperage rating are LASER-MARKED on both sides of all non color-coded PICO II™ fuses.

PART NUMBER			
AXIAL LEADS	RADIAL LEADS	AMPERE RATING	VOLTAGE RATING
251.062	252.062	1/16	125
251.125	252.125	1/8	125
251.250	252.250	1/4	125
251.375	252.375	3/8	125
251.500	252.500	1/2	125
251.750	252.750	3/4	125
251 001	252 001	1	125
251 01.5	252 01.5	1-1/2	125
251 002	252 002	2	125
251 02.5	252 02.5	2-1/2	125
251 003	252 003	3	125
251 03.5	252 03.5	3-1/2	125
251 004	252 004	4	125
251 005	252 005	5	125
251 007	252 007	7	125
251 010	252 010	10	125
251 012	252 012	12	32
251 015	252 015	15	32

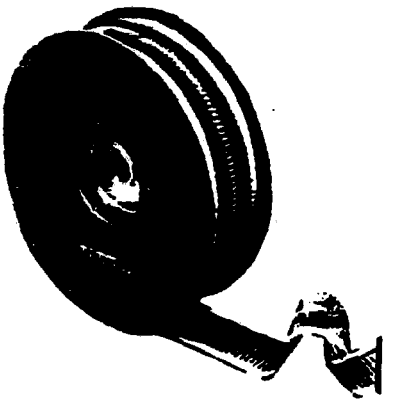
NOTE: To order color-coded PICO II™ fuses, use 255 Series (for Axial leads) or 256 Series (for Radial leads) in part number table above.

251 000 Series — Axial Leads



252 000 Series — Radial Leads

TAPED FUSES: PICO II™ Fuses are available on tape for use with automatic insertion equipment . . . Contact factory.



GENERAL SEMICONDUCTOR

TRANSZORB® UNIDIRECTIONAL 5KP5.0 THRU 5KP110A

ELECTRICAL CHARACTERISTICS @ 25°C

GENERAL SEMICONDUCTOR PART NUMBER	BREAKDOWN VOLTAGE BV VOLTS MIN.	BREAKDOWN VOLTAGE @ IT mA	MAXIMUM REVERSE LEAKAGE @ VR ID µA	MAXIMUM VOLTAGE TEMPERATURE VARIATION OF BV mV/°C
5KP5.0	6.40	50	2000	4.0
5KP5.0A	6.40	50	2000	4.0
5KP6.0	6.67	50	5000	4.0
5KP6.0A	6.67	50	5000	4.0
5KP6.5	7.22	50	2000	4.0
5KP6.5A	7.22	50	2000	4.0
5KP7.0	7.78	50	1000	5.0
5KP7.0A	7.78	50	1000	5.0
5KP7.5	8.33	5	250	6.0
5KP7.5A	8.33	5	250	6.0
5KP8.0	8.89	5	150	6.0
5KP8.0A	8.89	5	150	6.0
5KP8.5	9.44	5	50	7.0
5KP8.5A	9.44	5	50	7.0
5KP9.0	10.0	5	20	8.0
5KP9.0A	10.0	5	20	8.0
5KP10	11.1	5	15	9.0
5KP10A	11.1	5	15	9.0
5KP11	12.2	5	10	10
5KP11A	12.2	5	10	10
5KP12	13.3	5	10	11
5KP12A	13.3	5	10	11
5KP13	14.4	5	10	12
5KP13A	14.4	5	10	12
5KP14	15.6	5	10	13
5KP14A	15.6	5	10	13
5KP15	16.7	5	10	15
5KP15A	16.7	5	10	15
5KP16	17.8	5	10	18
5KP16A	17.8	5	10	16
5KP17	18.9	5	10	19
5KP17A	18.9	5	10	18
5KP18	20.0	5	10	20
5KP18A	20.0	5	10	19
5KP20	22.2	5	10	24
5KP20A	22.2	5	10	22
5KP22	24.4	5	10	27
5KP22A	24.4	5	10	24
5KP24	26.7	5	10	30
5KP24A	26.7	5	10	27
5KP26	28.9	5	10	33
5KP26A	28.9	5	10	29
5KP28	31.1	5	10	34
5KP28A	31.1	5	10	30
5KP30	33.3	5	10	38
5KP30A	33.3	5	10	35
5KP33	36.7	5	10	41
5KP33A	36.7	5	10	38
5KP36	40.0	5	10	45
5KP36A	40.0	5	10	40
5KP40	44.4	5	10	50
5KP40A	44.4	5	10	45
5KP43	47.8	5	10	54
5KP43A	47.8	5	10	49
5KP45	50.0	5	10	57
5KP45A	50.0	5	10	51
5KP48	53.3	5	10	62
5KP48A	53.3	5	10	55
5KP51	56.7	5	10	65
5KP51A	56.7	5	10	60
5KP54	60.0	5	10	70
5KP54A	60.0	5	10	64
5KP58	64.4	5	10	77
5KP58A	64.4	5	10	69
5KP60	66.7	5	10	79
5KP60A	66.7	5	10	70
5KP64	71.1	5	10	85
5KP64A	71.1	5	10	75
5KP70	77.8	5	10	93
5KP70A	77.8	5	10	84
5KP75	83.3	5	10	100
5KP75A	83.3	5	10	90
5KP78	86.7	5	10	104
5KP78A	86.7	5	10	94
5KP85	94.4	5	10	113
5KP85A	94.4	5	10	102
5KP90	100	5	10	120
5KP90A	100	5	10	109
5KP100	111	5	10	134
5KP100A	111	5	10	122
5KP110	122	5	10	147
5KP110A	122	5	10	132

V_f at 100 amps peak, 8.3 msec sine wave = 3.5 volts maximum.

FIGURE 3—Pulse Waveform

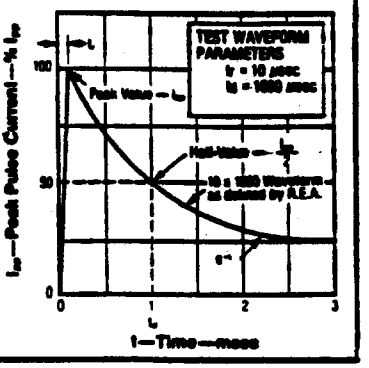
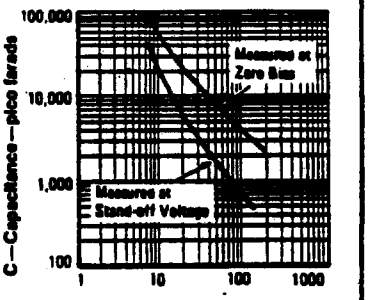


FIGURE 4—Typical Capacitance vs Breakdown Voltage



NOTES

Note 1: A TransZorb is normally selected according to the reverse "Stand Off Voltage" (V_R) which should be equal to or greater than the DC or continuous peak operating voltage level.

ABBREVIATIONS & SYMBOLS

V_R — Stand-Off Voltage: Applied Reverse Voltage to assure a nonconductive condition. (See Note 1)

BV(min) This is the minimum Breakdown Voltage the device will exhibit and is used to assure that conduction does not occur prior to this voltage level at 25°C.

V_C(max) Maximum Clamping Voltage. The maximum peak voltage appearing across the TransZorb when subjected to the peak pulse current in a one millisecond time interval. The peak pulse voltages are the combination of voltage rise due to both the series resistance and thermal rise.

I_{pp} — Peak Pulse Current — See Figure

P_p — Peak Pulse Power

I_R — Reverse Leakage

INTERNATIONAL RECTIFIER IOR

1N5401 SERIES

3 Amp Medium Power Silicon Rectifier Diodes

Major Ratings and Characteristics

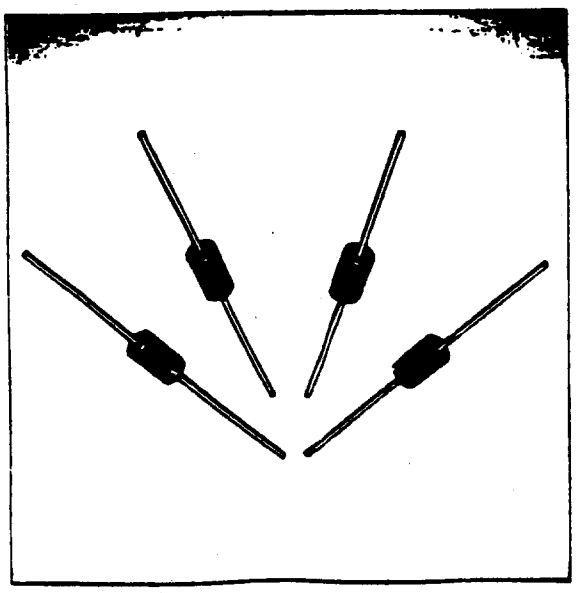
		1N5401	Units
$I_{F(AV)}$		3.0*	A
@ max T_L		105*	°C
I_{FSM}	50Hz	191	A
	60Hz	200*	A
$I^2\sqrt{t}$		2580*	$A^2\sqrt{s}$
T_L Range		–65 to 170*	°C
V_{RRM} Range		100 to 1000*	V

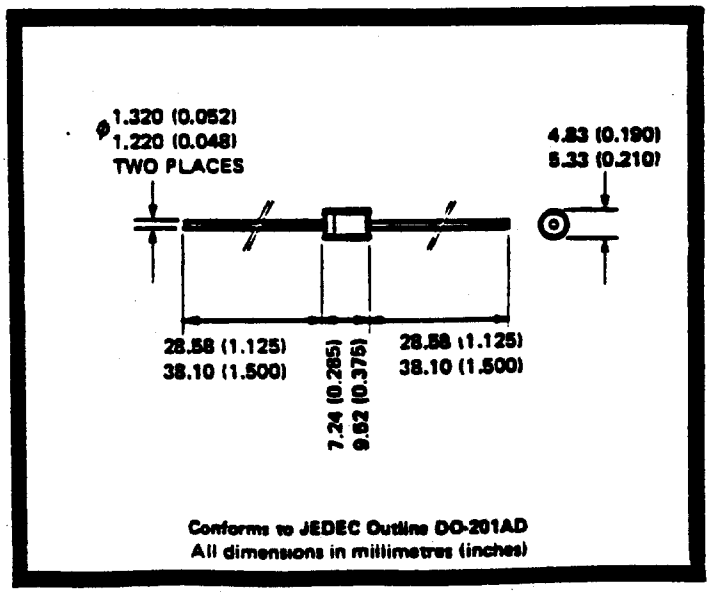
*JEDEC registered values

Description/Features

- 3A lead mounted rectifier.
- Subminiature molded package.
- Corrosion resistant surfaces.
- Peak reverse voltage from 100 to 1000V.
- Improved environmental operating capability.
- High surge current capability.

B





Voltage Ratings

	V_{RRM} – Maximum Repetitive Peak Reverse Voltage (V)	V_R – Maximum Direct Reverse Voltage (V)
Part numbers	T_L = –65 to 175°C	T_L = –65 to 150°C
1N5401	100*	100*
1N5402	200*	200*
1N5404	400*	400*
1N5406	600*	600*
1N5407	800*	800*
1N5408	1000*	1000*

Electrical Specifications

		1N5401	Units	Conditions
$I_{F(AV)}$	Maximum average forward current @ Max T_L	3.0*	A	180° sine wave conduction
		105*	°C	
I_{FSM}	Maximum peak one cycle non-repetitive surge current	191	A	t = 10ms, Following any rated load condition and with rated V_{RRM} reapplied
		200*	A	t = 8.3ms
$I^2\sqrt{t}$	Maximum $I^2\sqrt{t}$ for fusing ①	2580	$A^2\sqrt{s}$	t = 0.1 to 10ms, V_{RRM} = 0 following surge
V_{FM}	Maximum peak forward voltage	1.2*	A	$I_{F(AV)}$ = 3A (9.4 A_{pk}), T_J = 25°C
I_{RM}	Maximum average reverse current	100	μA	T_L = 105°C, V_{RRM} = rated V_{RRM}, $I_{F(AV)}$ = 3A
I_R	Maximum direct reverse current	500*	μA	T_L = 150°C, V_R = rated V_R

① I^2t for time $t_x = I^2\sqrt{t} \cdot \sqrt{t_x}$

*JEDEC registered value.

Thermal and Mechanical Specifications

T_L	Lead operating temperature range	–65 to 170*	°C	
T_{stg}	Storage temperature range	–65 to 175*	°C	
wt	Approximate weight	0.65 (0.023)	g (oz)	
T_{sld}	Max. lead temperature during soldering	240*	°C	Duration, 10s max. measured 3.2mm (0.125in) from device case

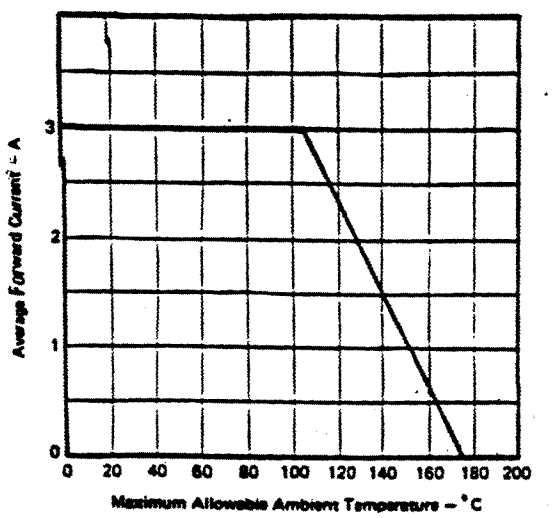


Fig. 1 – Average Forward Current Vs. Maximum Allowable Ambient Temperature

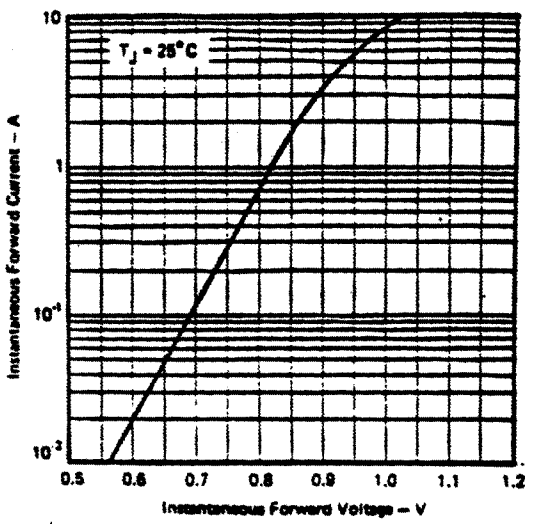


Fig. 2 – Maximum Forward Voltage Vs. Forward Current

B

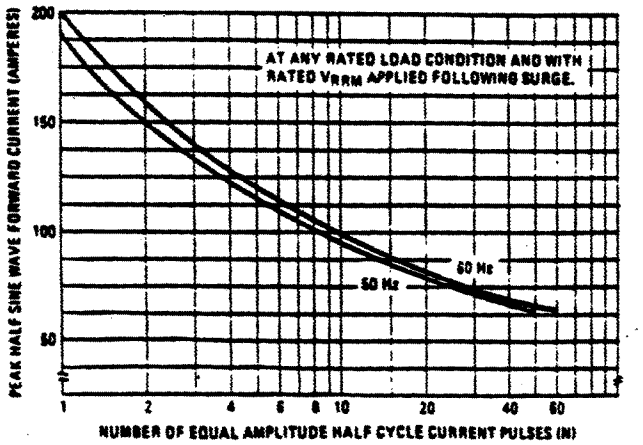


Fig. 3 – Maximum Non-Repetitive Surge Current Vs. Number of Current Pulses

INTERNATIONAL RECTIFIER IOR

1N4001 SERIES

1.0 Amp Silicon Rectifier Diodes

Major Ratings and Characteristics

	1N4001	Units
$I_{F(AV)}$	1.0 *	A
@ Max. T_A	75 *	°C
I_{FSM}		
@ 50 Hz	28.7	A
@ 60 Hz	30 *	A
$I^2\sqrt{t}$	58.1	$A^2\sqrt{s}$
T_J Range	-65 to 175 *	°C
V_{RRM} Range	50 to 1000 *	V

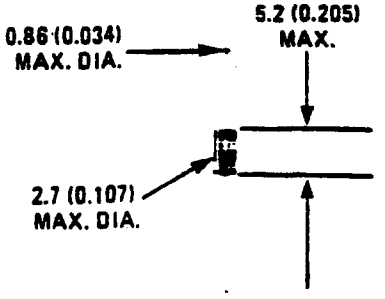
Description/Features

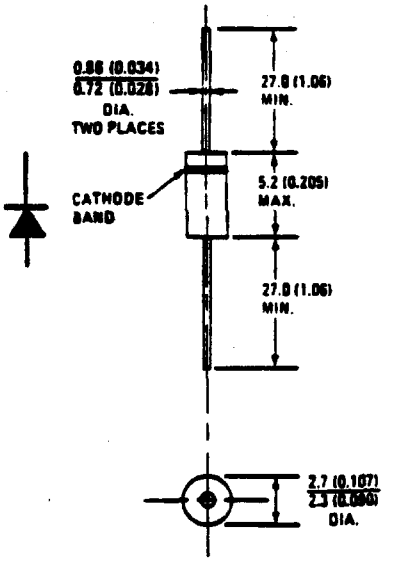
- Economical 1 ampere general purpose diode for industrial application
- Molded epoxy DO-204AL case style
- Low forward voltage drop and high surge rating
- Excellent reliability

B

*JEDEC registered value.

CASE STYLE AND DIMENSIONS





Case Style DO-204AL (DO-41)
Dimensions in Millimeters and (Inches)

VOLTAGE RATINGS

Part Number	V_{RRM} – Max. Repetitive Peak Reverse Voltage (V) T = –65 to 175°C	$V_{R(RMS)}$ – Max. RMS Reverse Voltage (V) T = –65 to 175°C	V_R – Max. DC Blocking Voltage (V) T = –65 to 165°C
1N4001	50*	35	50*
1N4002	100*	70	100*
1N4003	200*	140	200*
1N4004	400*	280	400*
1N4005	600*	420	600*
1N4006	800*	560	800*
1N4007	1000*	700	1000*

ELECTRICAL SPECIFICATIONS

		1N4001	Units	Conditions
$I_{F(AV)}$	Max. average forward current	1.0*	A	Half sine wave conduction
	@ Max. T_A	75*	°C	① , double side cooled.
I_{FSM}	Max. peak one cycle, non-repetitive surge current	28.7	A	Half cycle 60 Hz sine wave or 8 ms rectangular pulse. Following any rated load condition, and with rated V_{RRM} reapplied.
		30*		Half cycle 60 Hz sine wave or 5 ms rectangular pulse.
$I^2\sqrt{t}$	Max. $I^2\sqrt{t}$ for fusing ①	58.1	$A^2\sqrt{s}$	t = 0.1 to 10 ms with V_{RRM} following surge = rated V_{RRM}.
V_{FM}	Max. peak forward voltage	1.1*	V	T_A = –65°C to 75°C, I_F = 1 Adc
		1.6*	V	T_A = –65°C to 75°C, $I_{F(AV)}$ = 1A (3.14A peak)
I_R	Max. dc reverse current	10*	μA	T_A = 25°C V_R = Rated V_R.
		50*	μA	T_A = 100°C
$I_{R(AV)}$	Max. average reverse current	30*	μA	T_A = 75°C, $I_{F(AV)}$ = 1A, V_{RRM} = rated V_{RRM}

THERMAL-MECHANICAL SPECIFICATIONS

T_J	Max. operating junction temperature range	–65* to 175*	°C	
T_{stg}	Max. storage temperature range	–65* to 200*	°C	
wt	Approximate weight	0.33 (0.012)	g (oz)	
	Case Style	DO-204AL (DO-41)		

① T_L is measured 8.7 mm (0.344 in.) to 9.5 mm (0.375 in.) from device case.

① I^2t for time $t_x = I^2\sqrt{t} \cdot \sqrt{t_x}$.

*JEDEC registered values.

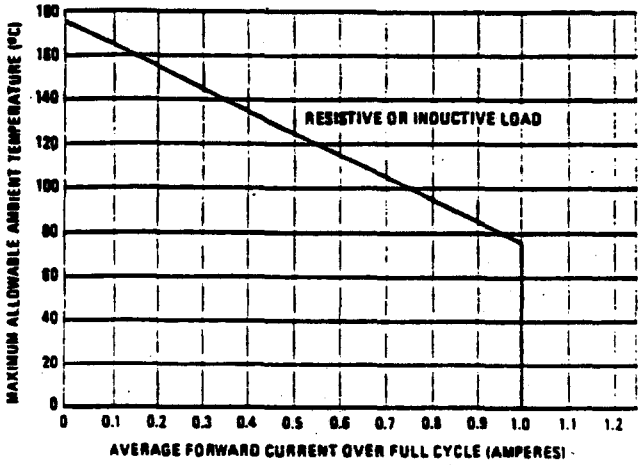


Fig. 1 – Average Forward Current Vs. Maximum Allowable Ambient Temperature

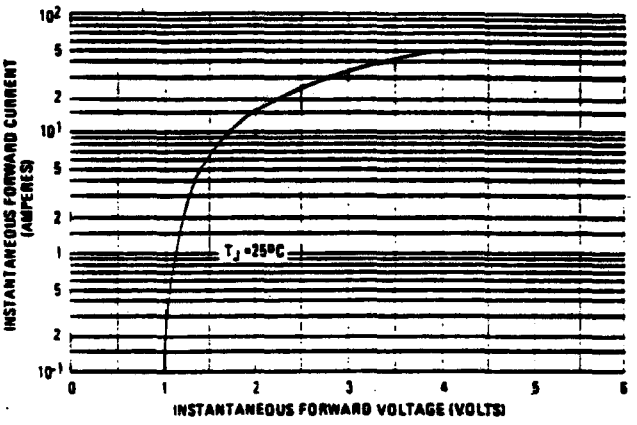


Fig. 2 – Maximum Forward Voltage vs. Forward Current

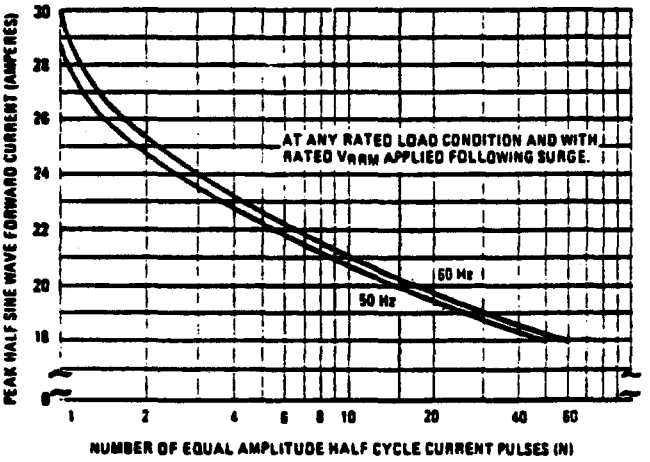


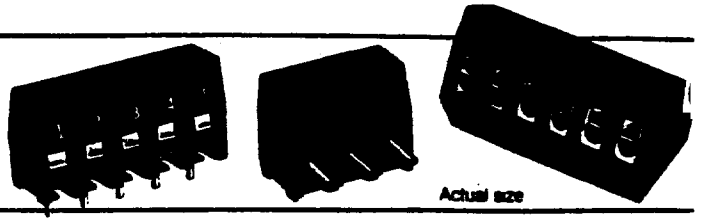
Fig. 3 – Maximum Non-Repetitive Surge Current Vs. Number of Current Pulses

Beau® Eurostyle® PC Terminal Strips

Single Row, Closed Side

15 amps
.197" centers (5,0)
thermoplastic

85



Specifications

Series No.	85
Construction	Closed side: feed thru
Terminal centers	.197" (5,0)
Current rating, amps (UL/CSA)	15 / 15
Voltage rating, rms 1. UL Recognized. . . Class B: Comm'l equip Class C: Gen'l ind'l Class C: Limited rating 2. CSA Certified Type B: Comm'l equip Type D: Special components 3. Withstand volts, rms	 250 50 300 300 300 6500
Insulator material	Thermoplastic. UL temp index 130°C. UL flame retardant rating 94V-0. Color, black.
Wire size recommended, AWG	14 (max); 30 (min)
Terminal screws (standard; also see options below)	No. M2.5 × 0.45 captive head screws. Steel, zinc plated.
Bottom terminal (3rd digit of part no.) ▲	4 45° PC terminal 5 Straight-thru PC terminal 8 Right-angle PC terminal
No. of circuits (4th & 5th digits of part no.)▲	2 to 24
Options (add dash numbers to part no.) ▲	−10A thru −11D Imprinting (on the open side) −49 Nickel-plated brass captive screws −72 Without wire guard −P With standoff pads, .080" dia × .025" (2,03 × 0,64). Not available on 45° style. −10mm 10mm spacing (12 circuit max)

Dimensions in parentheses are millimeters.

High circuit density. .197" (5 mm) contact spacing provides 5 terminations in less than an inch.

Very small footprint. Takes very little PCB space.

Made in USA. Beau Eurostyle PC terminal strips are a direct equivalent to popular 5 mm European style terminal strips.

Touchproof. Recessed screws prevent accidental shorting.

Captive screws won't fall out. Prevents accidents from screws falling into assembly. Furnished backed out, ready to wire.

Captive wire protectors hold wires securely in place. Assures a good connection and increases pullout strength.

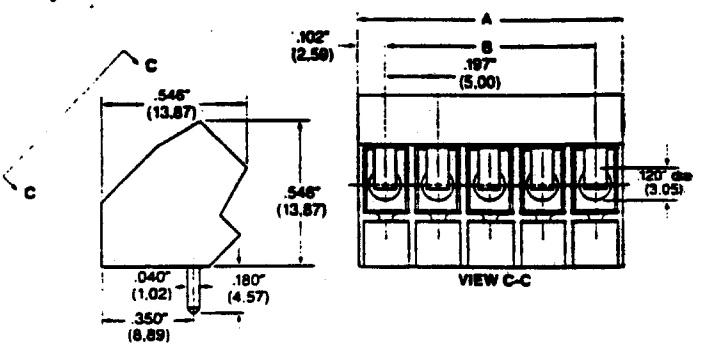
Large wire entry. Easy entry for stranded or multiple wires.

Easy identification. Imprinting, .070" high, can be provided on the front side of the strip.

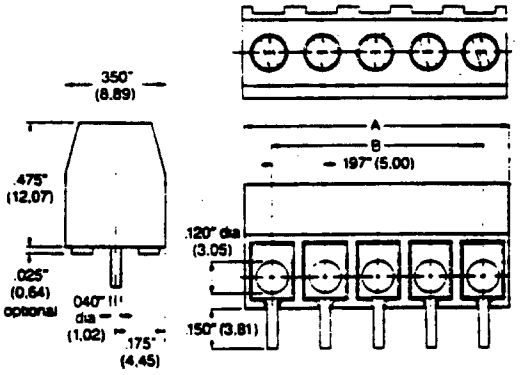
Closed side acts as wire stop. Wires can't pass through to short out components. Saves space by closer component mounting.

▲HOW TO ORDER. First two digits of part number are series number. 3rd digit is bottom terminal. 4th and 5th digits are no. of circuits. For options, add dash numbers. Example: 85509-49-P.

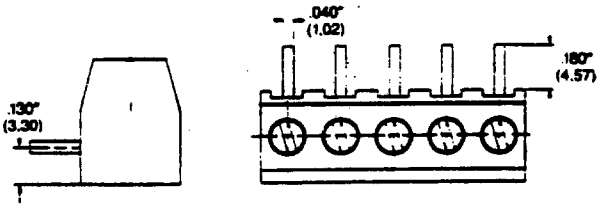
Style 4: 45° PC terminals



Style 5: Straight-thru PC terminals



Style 8: Right-angle PC terminals



No. of circuits	A in.	A mm	B in.	B mm
2	.400	10.15	.197	5,00
3	.597	15.15	.394	10,00
4	.793	20.15	.591	15,00
5	.990	25.15	.787	20,00
6	1.187	30.15	.984	25,00
7	1.384	35.15	1.181	30,00
8	1.581	40.15	1.378	35,00
9	1.778	45.15	1.575	40,00
10	1.975	50.15	1.772	45,00
11	2.171	55.15	1.969	50,00
12	2.368	60.15	2.165	55,00
13	2.566	65.15	2.362	60,00
14	2.762	70.15	2.560	65,00
15	2.959	75.15	2.756	70,00
16	3.156	80.15	2.953	75,00
17	3.352	85.15	3.150	80,00
18	3.549	90.15	3.346	85,00
19	3.746	95.15	3.543	90,00
20	3.943	100.15	3.740	95,00
21	4.140	105.15	3.937	100,00
22	4.337	110.15	4.134	105,00
23	4.534	115.15	4.331	110,00
24	4.730	120.15	4.528	115,00

Tolerance on length ± (.005" + .002" per inch) . . . ± (0.13 + 0.002 per mm)

Inches (millimeters)

ELECTRICAL RATINGS

Impedance: 29 ohms
Frequency Range: 0 - 2 GHz
Working Voltage: 300 Vrms
Dielectric Withstanding Voltage: 2000 VDC
Contact Resistance: Center contact: 6 milliohms
Outer conductor: 2.5 milliohms

MECHANICAL RATINGS

Insertion Force: Inner conductor: 32 oz. max. 2 oz. min.
Outer conductor: 48 oz. max. 6 oz. min.

Durability: 250 cycles

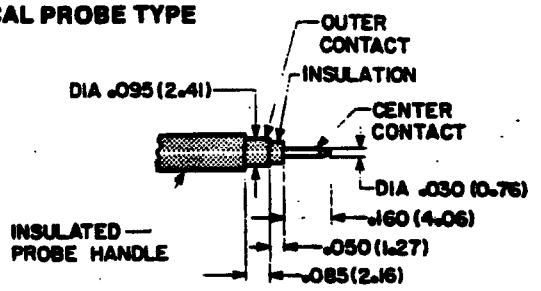
ENVIRONMENTAL RATINGS

Temperature Range: -65° C to +85° C
Corrosion: Salt spray, 24 hours
Shock: Method 213, Test Condition B, MIL-STD-202
Vibration: Method 204, Test Condition B, MIL-STD-202

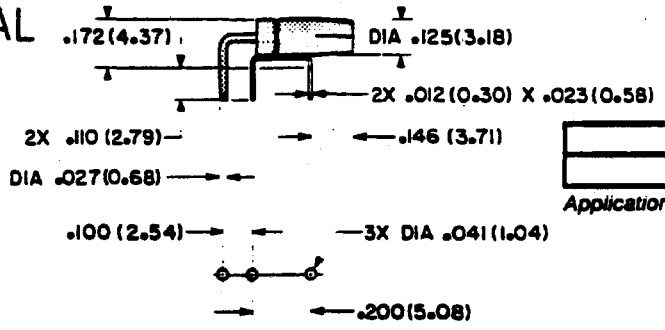
MATERIAL SPECIFICATIONS

Contact: Beryllium copper per QQ-C-533/530
Insulator Material: TFE Fluorocarbon per L-P-403
Finish: Center Contact and Body: Bright Tin

SHIELDED TEST JACK INTERFACE — TYPICAL PROBE TYPE



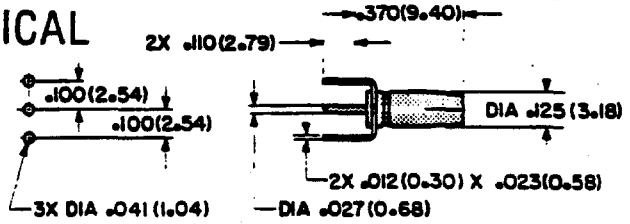
HORIZONTAL



BRIGHT TIN PLATED
129-0701-301

Application detail Page 68

VERTICAL



BRIGHT TIN PLATED
129-0701-201

Application detail Page 68

TIP JACKS

APPLICATION • Connection to test equipment • Power supplies • Electrical instruments

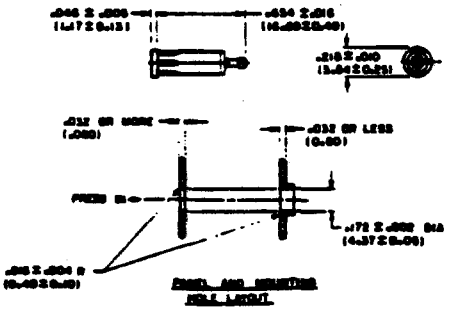
Insulated Standard: Rib-Loc® Type

SPECIFICATIONS

Contact: Brass body per QQ-B-626D with wrap around beryllium copper spring per QQ-C-533
Finish: Silver
Body: Nylon 66 per ASTM D4066
Std. Colors: Ten per FED-STD-595
Panel Thickness: Up to .375" (9.05mm)

RATINGS

Current: 5 amps
Breakdown Voltage: 3500 Vrms minimum†
Contact Resistance: 0.005 ohms maximum
Contact-To-Panel C: 1 pF nominal



For Standard Tip Plug: .080" (2mm) Diameter

FEATURES:

- Machined contact with turret terminal
- Installs by pressing into mounting hole, no mounting hardware required
- Closed entry blocks access of probes greater than .085" (2.16mm)
- Nylon UL approved for self extinguishing

SILVER PLATED			
PART NO.	COLOR	PART NO.	COLOR
105-1041-001	White	105-1047-001	Yellow
105-1042-001	Red	105-1048-001	Brown
105-1043-001	Black	105-1050-001	Blue
105-1044-001	Green	105-1052-001	Violet
105-1046-001	Orange	105-1053-001	Gray

E.F. JOHNSON COMPANY, Component Products, P.O. Box 59089, Minneapolis, Minnesota 55459
OUTSTATE CALL TOLL FREE: 1-800-247-8256 IN MINN. CALL: 1-507-835-6222 TLX: 290470 FAX: 1-507-835-6310

† Avoid user injury due to misapplication.

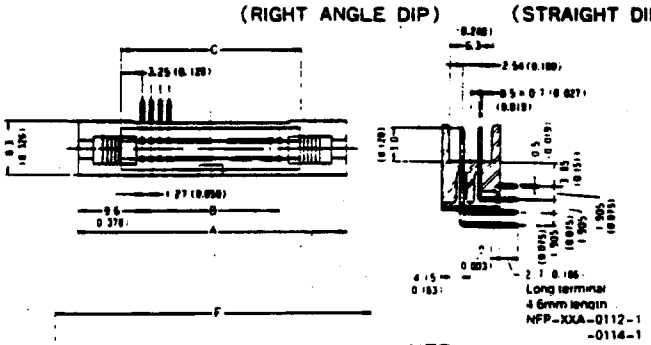
HEADER A TYPE

0.635mm PITCH (25MIL PITCH)

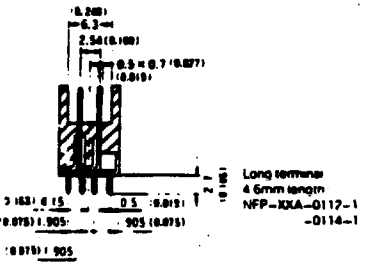
NFP—00A—0112、0114

NFP—00A—0112、NFP—00A—0114

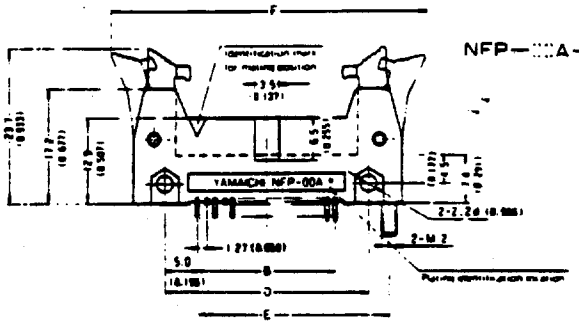
(RIGHT ANGLE DIP) (STRAIGHT DIP)



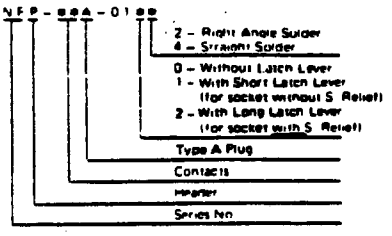
NFP—00A—0112



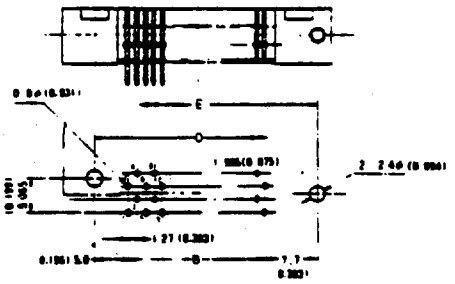
NFP—00A—0114



PART No. IDENTIFICATION



Note:
1. Add "A" at the end of Part No. for optional grade A gold plating.
2. Optional right angle or straight longer solder tail for multi layer P.C. Board is available. Contact factory for details.



P.C. BOARD HOLE SIZE

	TYPE NO	A	B	C	D	E	F
☆	NFP-10A-0112-0114	24.28 (0.955)	5.08 (0.200)	11.58 (0.455)	15.08 (0.593)	0.48 (0.806)	38.2 (1.503)
☆	NFP-16A-0112-0114	28.09 (1.105)	8.89 (0.350)	15.39 (0.605)	18.89 (0.743)	21.29 (0.838)	42.0 (1.653)
☆	NFP-20A-0112-0114	30.63 (1.205)	11.43 (0.450)	17.93 (0.705)	21.43 (0.843)	26.83 (1.056)	44.6 (1.755)
☆	NFP-26A-0112-0114	34.44 (1.355)	15.24 (0.600)	21.74 (0.855)	25.24 (0.993)	30.34 (1.246)	48.4 (1.905)
☆	NFP-34A-0112-0114	39.52 (1.555)	20.32 (0.800)	26.82 (1.055)	30.32 (1.193)	35.72 (1.406)	53.5 (2.106)
☆	NFP-40A-0112-0114	43.33 (1.705)	24.13 (0.950)	30.63 (1.205)	34.13 (1.343)	39.53 (1.556)	57.3 (2.255)
☆	NFP-50A-0112-0114	49.68 (1.955)	30.48 (1.200)	36.98 (1.455)	40.48 (1.593)	45.93 (1.808)	63.6 (2.503)
☆	NFP-60A-0112-0114	56.03 (2.205)	36.83 (1.450)	43.33 (1.705)	46.83 (1.843)	52.23 (2.056)	70.0 (2.755)
☆	NFP-64A-0112-0114	58.57 (2.305)	39.37 (1.550)	45.87 (1.805)	49.37 (1.943)	54.77 (2.156)	72.5 (2.854)
☆	NFP-80A-0112-0114	68.73 (2.705)	49.53 (1.950)	56.03 (2.205)	59.53 (2.343)	64.93 (2.556)	82.7 (3.255)
☆	NFP-100A-0112-0114	81.43 (3.206)	62.23 (2.450)	68.73 (2.706)	72.23 (2.844)	77.63 (3.056)	95.40 (3.756)

☆ Tooled

Dimension: mm (inch)

SPECIFICATION

1. Insulation Resistance : 1,000 MΩ Minimum at 500V D.C.
2. Break Down Voltage : 500V A.C. for One Minute
3. Voltage Rating : 150V
4. Current Rating : 0.5A
5. Operating Temperature : −20°C to 105°C
6. Mating Cable : Solid Conductor Flat Cable AWG 30 (with PVC and FEP Insulator)
   : Stranded Conductor Flat Cable AWG 30 and AWG 32 (with PVC and FEP Insulator)
   * FEP (Teflon)

MATERIAL

1. Body, Strain Relief, Latch : Polybutylene terephthalate (UL 94-VO)

2. Contact

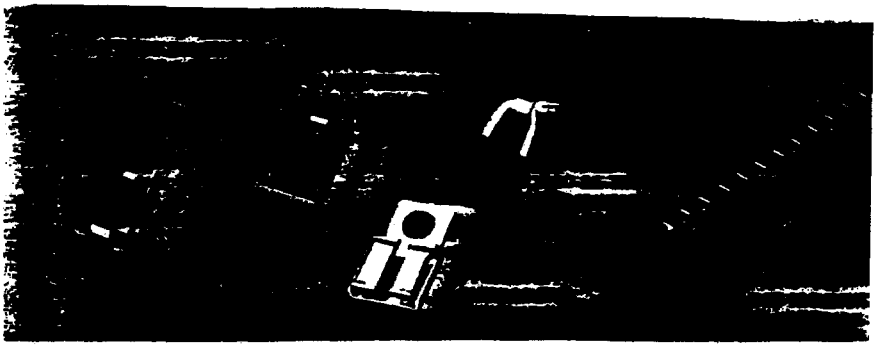
NFP-	A	: Phosphor Bronze
NFS-	A	: Beryllium Copper
NFP-	G	: Beryllium Copper
NBP	-1001	: Phosphor Bronze
NBS	-1001	: Phosphor Bronze
NBP	-1200	: Copper Nickel Lead
NBS	-1200	: Phosphor Bronze
NFP-	A-013	: Phosphor Bronze
NFS-	A-1314	: Phosphor Bronze

3. Metal Shell : Steel (Nickel Finish)

PLATING SPECIFICATION

TYPE No.	THICKNESS	CONTACT AREA	IDENTIFICATION
NFP- A-012 NFS- A NFP- A-0132	Grade A	Ni 2.5 – 4.5 micron (98 – 177 μinch) Au 0.76 micron (30 μinch)	A
	Grade B	Ni 2.5 – 4.5 micron (98 – 177 μinch) Au 0.3 – 0.5 micron (12 μinch)	STANDARD NO MARK
NFP- A-014 NFP- G NBP -1001 NBS -1001	Grade AF	Ni 2.5 – 4.5 micron (98 – 177 μinch) Au 0.76 micron (30 μinch)	A
	Grade BF	Ni 2.5 – 4.5 micron (98 – 177 μinch) Au 0.3 – 0.5 micron (12 μinch)	STANDARD NO MARK
NBP- -1200 NBS- -1200 NFP- A-0134 NFS- A-1314	Grade AS	Ni 2.5 – 4.5 micron (98 – 177 μinch) Au 0.76 micron (30 μinch)	A
	Grade BS	Ni 2.5 – 4.5 micron (98 – 177 μinch) Au 0.3 – 0.5 micron (12 μinch)	STANDARD NO MARK

Samtec



SHUNTS & JUMPERS SNT, MNT, SNL & JL SERIES

.100 Centerline Shunts with G.F. Polyester Insulator

Mates with Samtec TSW series and most other .025" Square headers.

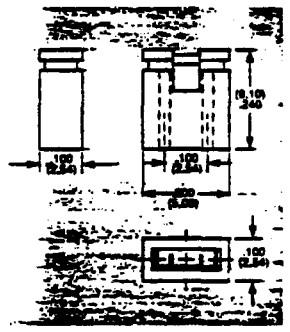
Available as Single (SNT) or Multi (MNT) Position.

Available with fast delivery on large quantities or small.

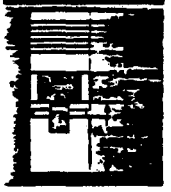
Specifications: SNT & MNT

Insulator Material: Glass Filled Polyester
Flammability Rating: UL 94V-O
Insulation Resistance: 5000 megohm @ 1000 VDC
Temperature Range: –40°C to +90°C
Withstanding Voltage: 1 KV eff. 60 sec
Contact Material: Phosphor Bronze
Contact Plating: Au over Ni or Sn over Ni
Current Rating: 2.5 amp
Contact Resistance: 5 milliohms @ 200 milliamp
Retention in Body: 1 lb
Lead size Range: .022" SQ to .028" SQ
Insertion Depth: .170" minimum
Insertion Force: 15 oz avg (.025" SQ pin)
Withdrawal Force: 12.8 oz avg (.025" SQ pin)

SNT	100	BODY COLOR	PLATING OPTION	HANDLE OPTION



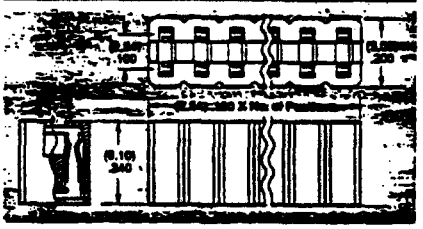
BODY COLOR	PLATING OPTION	HANDLE OPTION
–BK Black	–T = 200µ" Tin	–H Handle (Black only)
–OR Orange	–G = 10µ" Gold	
–RD Red	–H = 30µ" Gold on Contact, Gold Flash on Balance	
–BL Blue		
–YW Yellow		



Mates with: TSW, MTSW, STW, BST, LCW, TST

MNT	1	NO. OF POSITIONS	BODY COLOR	PLATING OPTION

02 thru 20 (20 positions standard)



BODY COLOR	PLATING OPTION
–BK = Black	–T = 200µ" Tin
	–G = 10µ" Gold
	–H = 30µ" Gold on Contact, Gold Flash on Balance

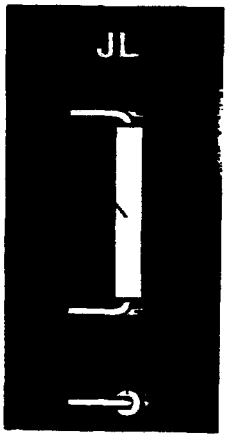
Mates with: TSW, MTSW, STW, BST, LCW, TST

Note: Non-standard strip lengths are non-returnable.

JL Series Jumper Links Meet Mil Specs

Specifications: JL

Insulator Material: White Teflon per MIL–T–16878 Type E
Flammability Rating: VWI
Insulation Resistance: >10^{13} ohm-cm
Temperature Range: –105°C to +200°C
Withstanding Voltage: 600 V continuous
Terminal Material: Tinned 22 gauge Copper Wire
Plating: Ag per MIL–QQW–343
Current Rating: 1.6 amp @ 60°C
Packaging: 1000 per bag



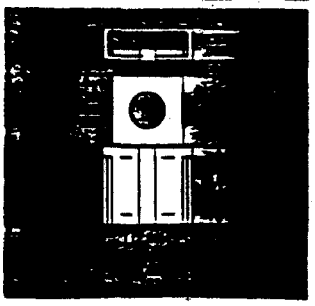
Part No.	A
JL-100-25-T	(2,54) .100
JL-250-25-T	(6,35) .250
JL-300-25-T	(7,62) .300
JL-400-25-T	(10,16) .400
JL-500-25-T	(12,70) .500
JL-600-25-T	(15,24) .600
JL-1000-25-T	(25,40) 1.000

Economical SNL Series Shunt with No Insulator for Lowest Profile

Specifications: SNL

Same as SNT except:
Insertion Force: 16 oz avg (.025" SQ pin)
Withdrawal Force: 8 oz avg (.025" SQ pin)
Insulation Material: None

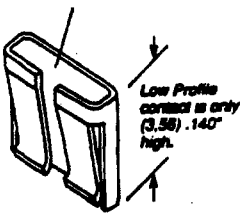
SNL	100	PLATING OPTION



PLATING OPTION
–T = 200µ" Tin
–G = 10µ" Gold

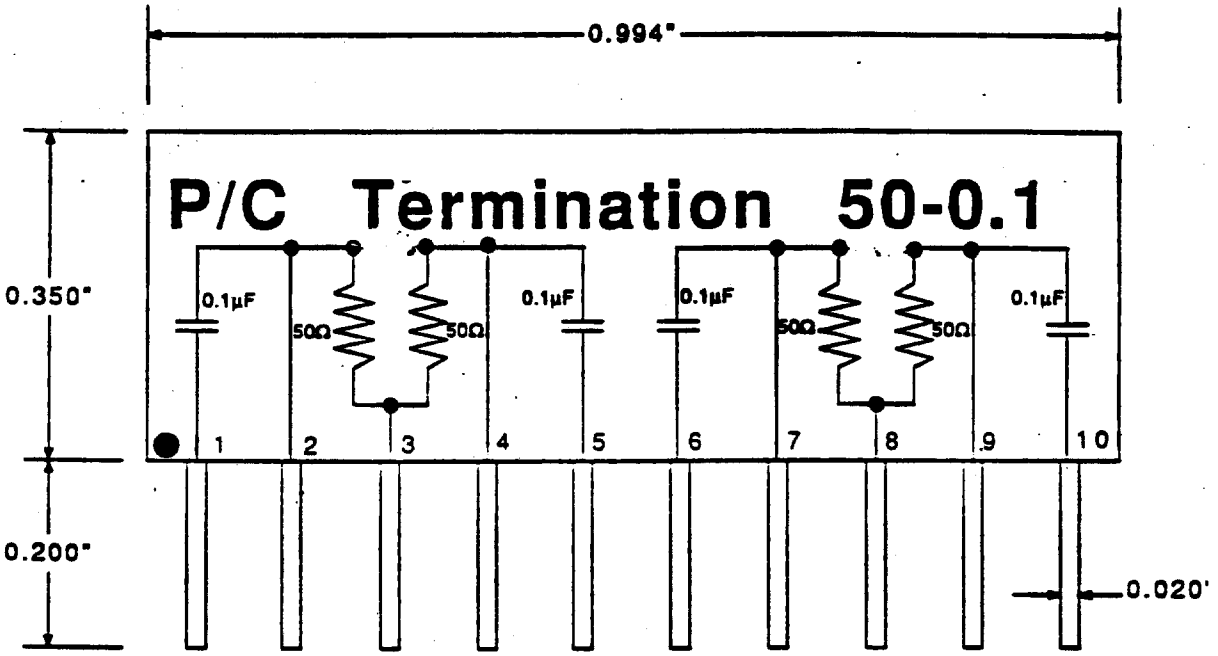
Mates with: TSW, MTSW, STW, BST, LCW, TST

Economical shunt has no body, supplied with convenient break-off tab.



Postamp/Comparator Module Termination Networ

Package Marking



Specifications:

Physical Requirements

Maximum Package Height: 0.350"
Maximum Package Length: 1.000"
Maximum Package Thickness: 0.100"
Package Marking: "P/C TERMINATION 50-0.1"

Electrical Requirements

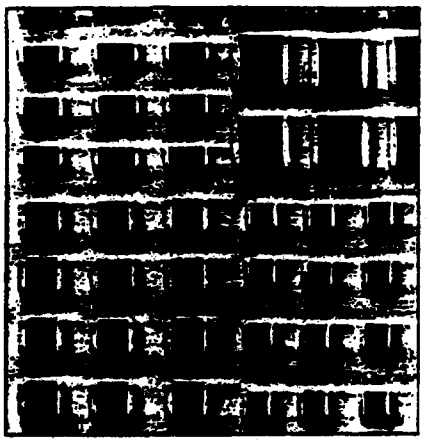
Resistors
Value: 50Ω, +/- 2%
Power: 100 mW @ 70 C
TCR: +/-100PPM
Ratio Match between all four resistors +/-1%

Capacitors
Dielectric Type: Z5U
Value: 0.1µF, +80,-20%
Working Voltage: 25 Volts

Merle Haldeman
10-3-89

SURFACE MOUNTED MONOLITHIC CERAMIC CHIP CAPACITORS

Series GRM



FEATURES

- **Miniature size**
- **Wide capacitance, TC, voltage and tolerance range**
- **Industry standard sizes**
- **8 mm and 12 mm tape & reel for auto-placement**
- **Nickel barrier layer termination is standard**
- **Largest production volume and capacity in the industry**

PART NUMBERING SYSTEM

GRM 40 --- X7R 103 K 050 A D

CAPACITOR TYPE AND SIZE
See below and following pages.

3-digit code appears as necessary to indicate special thickness requirements. Please consult your local sales office for details.

TEMPERATURE CHARACTERISTICS
Standard TC's (described herein)
COG = 0, ±30ppm (Notes 1 & 2)
X7R = ±15%
Z5U = +22, −56%
Y5V = +22, −82%
(See Note 2 below)
P2H = N150 ± 60ppm
R2H = N220 ± 60ppm
S2H = N330 = ±60ppm
T2H = N470 ± 60ppm
U2J = N750 ± 120ppm

CAPACITANCE VALUE
Expressed in picofarads and identified by a three-digit number. First two digits represent significant figures. Last digit specifies the number of zeros to follow. For fractional values below 10pF, the letter "R" is used as the decimal point and the last digit becomes significant. (5.7pF = 5R7)

CAPACITANCE TOLERANCE
(Note 2)
COG: (10pF or less)
B = ±.1pF
C = ±.25pF
D = ±.5pF
F = ±1pF on special request
(over 10pF)
F = ±1%
G = ±2%
J = ±5%
K = ±10%
X7R: K = ±10%
M = ±20%
J = ±5% on special request
Z5U: M = ±20%
Z = +80, −20%
Y5V: Z = +80, −20%

VOLTAGE
Identified by a three-digit number.

MARKING
A = Unmarked
B = EIA Marking (standard)
C = Non-standard

PACKAGING

Reel Diameter Tape Material	EIA-481-A Standard
7" Paper Tape	D
7" Plastic Tape	L
13" Paper Tape	J
13" Plastic Tape	K
Bulk	B

See page 35 for additional marking and packaging information.

NOTE 1: TC Tolerance (COG)

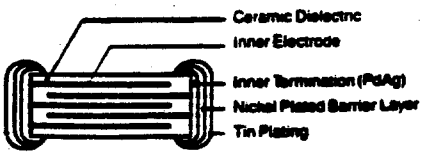
Capacitance (pF)	TC Tolerance (ppm)
10 or over	± 30 (G)
4.0-9.9	± 60 (H)
2.1-3.9	±120 (J)
.4-2.0	±250 (K)

NOTE 2: Refer to EIA-RS 198 for limitations

CHIP TERMINATION DIAGRAMS

Nickel Barrier Layer (Standard)

GRM Series



NOTE: Other Terminations Available Upon Request. Please Contact Local Sales Office.

39,40,43,44

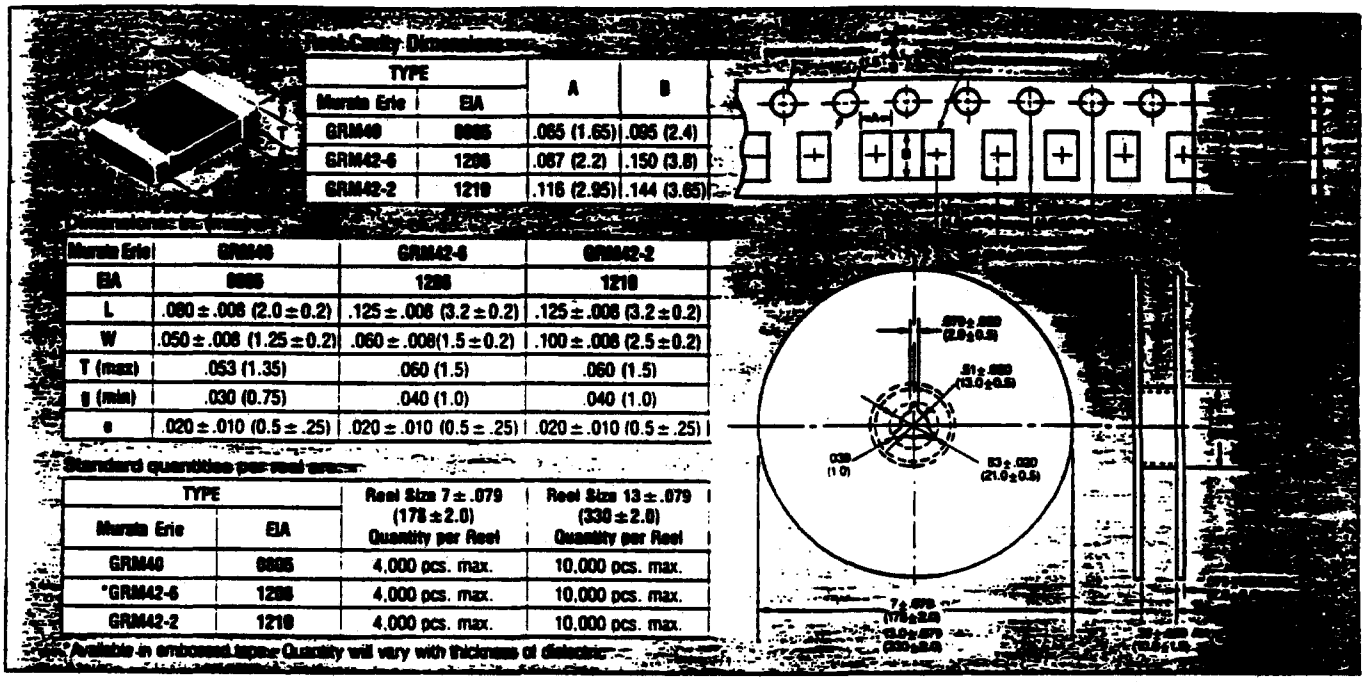
SURFACE MOUNTED CHIP CAPACITORS
TAPE & REEL
8mm to EIA RS481 Paper Tape

.01, .1, 470pF solf

Murata ERIE

Series GRM

DIMENSIONS: In. (mm)



TYPE		A	B
Murata Erie	EIA		
GRM40	0805	.065 (1.65)	.095 (2.4)
GRM42-6	1206	.087 (2.2)	.150 (3.8)
GRM42-2	1210	.116 (2.95)	.144 (3.65)

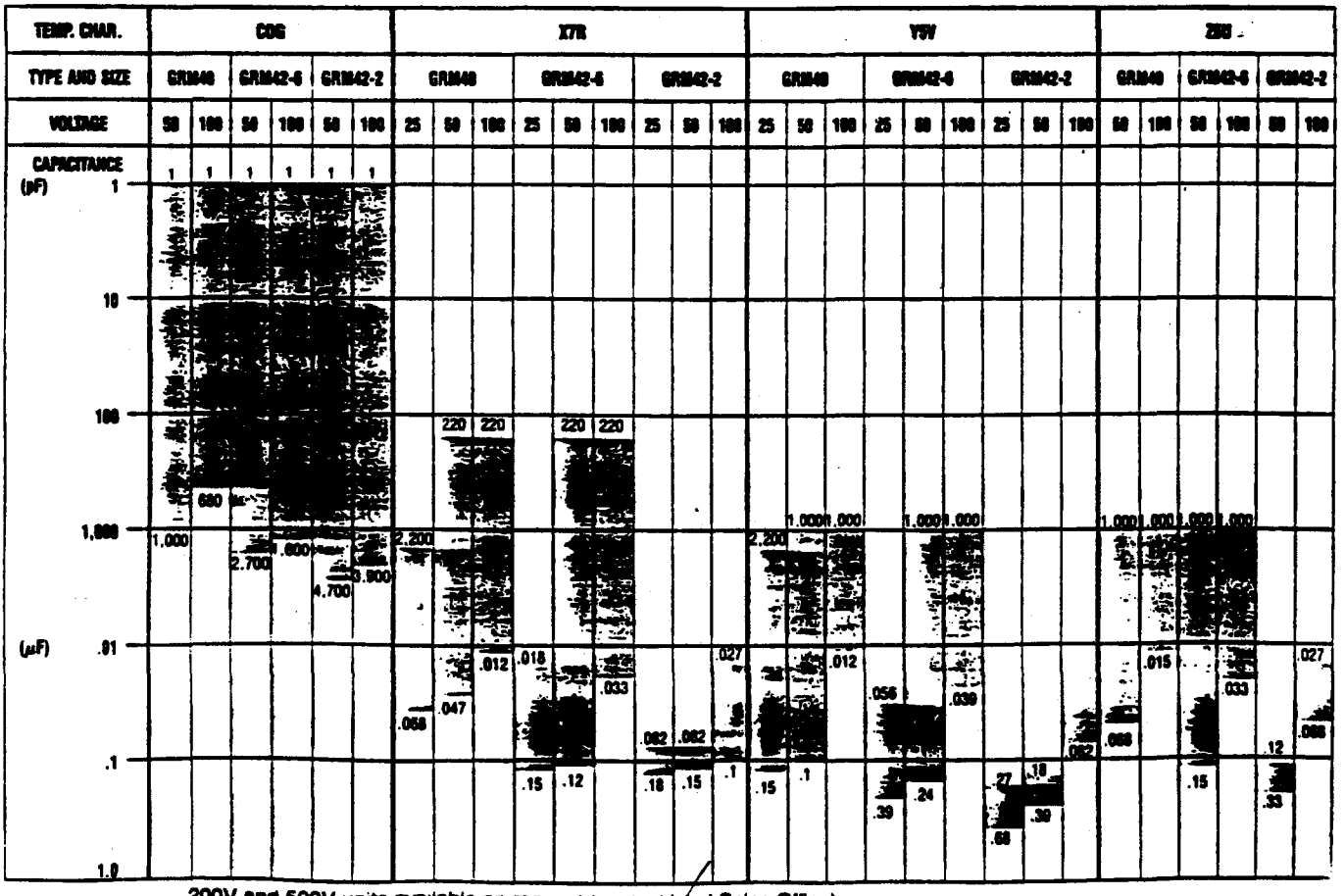
Murata Erie	GRM40	GRM42-6	GRM42-2
EIA	0805	1206	1210
L	.080 ± .008 (2.0 ± 0.2)	.125 ± .008 (3.2 ± 0.2)	.125 ± .008 (3.2 ± 0.2)
W	.050 ± .008 (1.25 ± 0.2)	.060 ± .008 (1.5 ± 0.2)	.100 ± .008 (2.5 ± 0.2)
T (max)	.053 (1.35)	.060 (1.5)	.060 (1.5)
g (min)	.030 (0.75)	.040 (1.0)	.040 (1.0)
a	.020 ± .010 (0.5 ± .25)	.020 ± .010 (0.5 ± .25)	.020 ± .010 (0.5 ± .25)

Standard quantities per reel are:

TYPE		Reel Size 7 ± .079 (178 ± 2.0)	Reel Size 13 ± .079 (330 ± 2.0)
Murata Erie	EIA	Quantity per Reel	Quantity per Reel
GRM40	0805	4,000 pcs. max.	10,000 pcs. max.
*GRM42-6	1206	4,000 pcs. max.	10,000 pcs. max.
GRM42-2	1210	4,000 pcs. max.	10,000 pcs. max.

*Available in embossed tape. Quantity will vary with thickness of dielectric.

CAPACITANCE RANGE—8mm TAPE & REEL



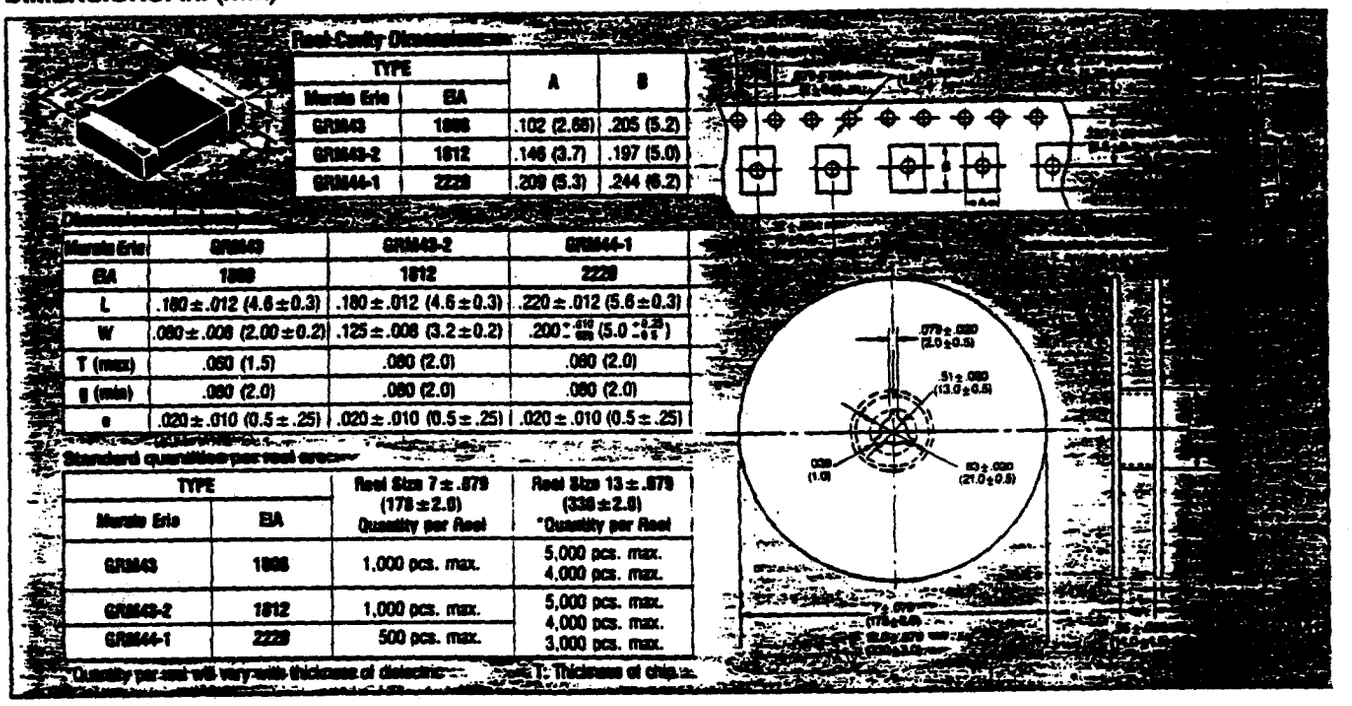
200V and 500V units available on request (contact local Sales Office).

(39,40,43,44

SURFACE MOUNTED CHIP CAPACITORS
TAPE & REEL
12mm To EIA RS481 Embossed Plastic Tape

Series GRM

DIMENSIONS: in. (mm)



Reel Cavity Dimensions

TYPE		A	B
Murata Erie	EIA		
GRM43	1808	.102 (2.60)	.205 (5.2)
GRM43-2	1812	.146 (3.7)	.197 (5.0)
GRM44-1	2220	.209 (5.3)	.244 (6.2)

Murata Erie	GRM43	GRM43-2	GRM44-1
EIA	1808	1812	2220
L	.180±.012 (4.6±0.3)	.180±.012 (4.6±0.3)	.220±.012 (5.6±0.3)
W	.080±.008 (2.00±0.2)	.125±.008 (3.2±0.2)	.200 (5.0)
T (max)	.060 (1.5)	.060 (2.0)	.060 (2.0)
g (min)	.060 (2.0)	.060 (2.0)	.060 (2.0)
e	.020±.010 (0.5±.25)	.020±.010 (0.5±.25)	.020±.010 (0.5±.25)

Standard quantities per reel are:

TYPE		Reel Size 7±.079 (178±2.0)	Reel Size 13±.079 (330±2.0)
Murata Erie	EIA	Quantity per Reel	*Quantity per Reel
GRM43	1808	1,000 pcs. max.	5,000 pcs. max. 4,000 pcs. max.
GRM43-2	1812	1,000 pcs. max.	5,000 pcs. max. 4,000 pcs. max.
GRM44-1	2220	500 pcs. max.	3,000 pcs. max.

*Quantity per reel will vary with thickness of dielectric. T: Thickness of chip.

CAPACITANCE RANGE—12mm TAPE & REEL

TEMP. CHAR.	C0G						X7R						Y5V									Z5U					
TYPE AND SIZE	GRM43		GRM43-2		GRM44-1		GRM43		GRM43-2		GRM44-1		GRM43			GRM43-2			GRM44-1			GRM43		GRM43-2		GRM44-1	
VOLTAGE	50	100	50	100	50	100	25/50	100	25/50	100	25/50	100	25	50	100	25	50	100	25	50	100	50	100	50	100	50	100
CAPACITANCE (pF) 1																											
10																											
100	150	150																									
1,000	4,300	3,900	3,900	8,800	3,300	6,200	5,600	5,600	3,900	3,900												.01	.01				
(μF) .01							.016	.015			.10	.082			.068			.068					.12		.082		
.1							.22	.12	.39	.18	.27	.39	.22	.47	.12	.22	.39	.22	1.0	1.0	.56	.47		.39	.18	1.0	.33
1.											.62		.68			.82	.68		2.2	1.5				.68		.15	.39
10																											

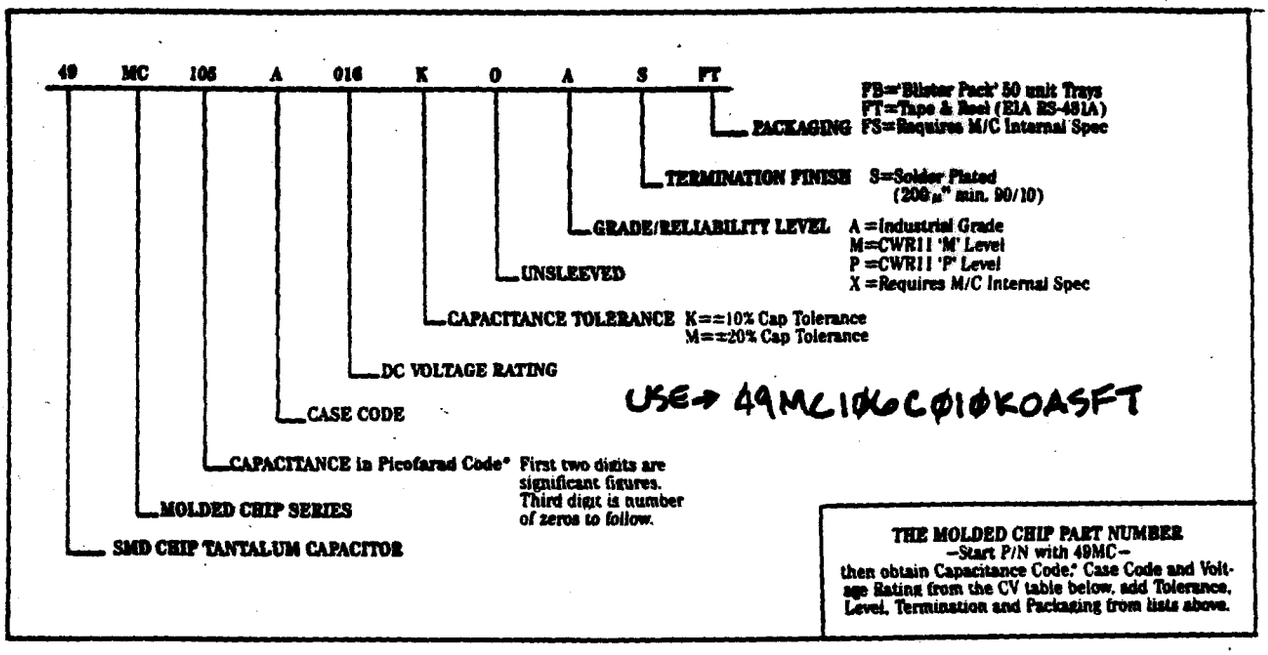
200V and 500V units available on request (contact local Sales Office).

39, 40, 43, 4

Mepco / Centralab

PART NUMBER DESIGNATION

MEPCO/CENTRALAB Series 49MC Molded Chip Capacitors can be completely specified using the following designation:



PHYSICAL SPECIFICATIONS

DIMENSIONS—Inches

CASE CODE EIA	CASE CODE 49MC	(L) LENGTH	(W) WIDTH	(H) HEIGHT	(B) PAD WIDTH	TAB DIMENSIONS (W2) min-max	TAB DIMENSIONS (H2)
3216	A	.126 ± .008	.063 ± .008	.063 ± .008	.031 ± .012	.043-.051	.028 min
3528	B	.138 ± .008	.110 ± .008	.075 ± .008	.031 ± .012	.083-.091	.028 min
6032	C	.236 ± .012	.126 ± .012	.098 ± .012	.051 ± .012	.083-.091	.039 min
7343	D	.287 ± .012	.169 ± .012	.110 ± .012	.051 ± .012	.091-.098	.039 min

DIMENSIONS—mm

CASE CODE EIA	CASE CODE 49MC	(L) LENGTH	(W) WIDTH	(H) HEIGHT	(B) PAD WIDTH	TAB DIMENSIONS (W2) min-max	TAB DIMENSIONS (H2)
3216	A	3.2 ± 0.2	1.6 ± 0.2	1.6 ± 0.2	0.8 ± 0.3	1.1-1.3	0.7 min
3528	B	3.5 ± 0.2	2.8 ± 0.2	1.9 ± 0.2	0.8 ± 0.3	2.1-2.3	0.7 min
6032	C	6.0 ± 0.3	3.2 ± 0.3	2.5 ± 0.3	1.3 ± 0.3	2.1-2.3	1.0 min
7343	D	7.3 ± 0.3	4.3 ± 0.3	2.8 ± 0.3	1.3 ± 0.3	2.3-2.5	1.0 min

41

49MC CAPACITANCE—VOLTAGE—CASE CODE TABLE

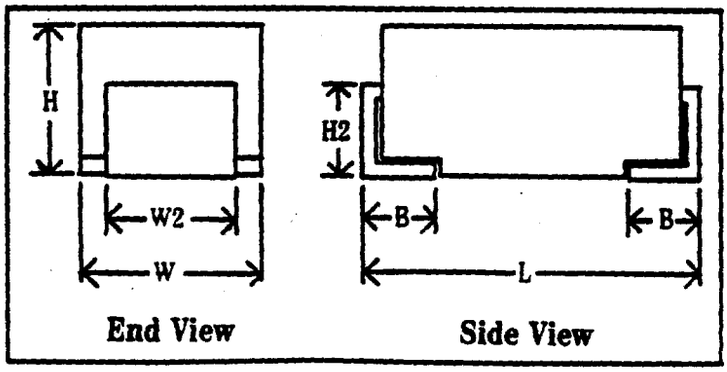
Select standard Capacitance/Voltage ratings and corresponding standard Molded Chip case codes from this table.

CAP μF	PICOFARAD CODE*	004 V	006 V	010 V	016 V	020 V	025 V	035 V	050 V
0.10	104	→	→	→	→	→	→	A	A
0.15	154	→	→	→	→	→	→	A	B
0.22	224	→	→	→	→	→	→	A	B
0.33	334	→	→	→	→	→	→	A	B
0.47	474	→	→	→	→	→	A	B	C
0.68	684	→	→	→	→	A	→	B	C
1.0	105	→	→	→	A	→	→	B	C
1.5	155	→	→	A	→	→	B	C	D
2.2	225	→	A	→	→	B	→	C	D
3.3	335	A	→	→	B	C	C	D	D
4.7	475	→	→	B	C	C	D	D	
6.8	685	→	B	C	C	→	D	D	
10.0	106	B	→	C	C	D	D		
15.0	156	→	C	C	→	D	D		
22.0	226	C	→	→	D	D			
33.0	336	C	→	D	D				
47.0	476	→	D	D					
68.0	686	→	D						

Arrow indicates that next higher voltage is the standard rating available. Units will be marked with the highest voltage available for that case size and capacitance rating.

Devices rated at 6.3 volts will be marked 6 volts.

CASE OUTLINE DRAWING



MARKINGS

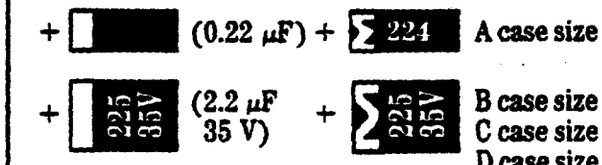
■ The small physical size of the MOLDED CHIP dicates a minimum amount of alpha-numeric marking on the body of the capacitor. The capacitance in picofarad code and the Rated Working Voltage (DC) will be marked on the B, C, and D case size units. The A case size units may have the capacitance only marked in picofarad code (see "How To Specify" on page 48). All case sizes will have positive polarity indicated at the anode termination. Capacitance may be marked in μF at the manufacturer's option.

POLARITY
(Mepco/Centralab Unit Identifier)
(Positive Termination Indicator)

■ Series 49MC capacitors are polar devices. Proper polarity must be observed or damage to the capacitor and/or the circuit will result. Polarity is marked with a white stripe or the letter "M" (in white) on the positive (anode) end. The letter "M" will serve as the Mepco/Centralab identifier as well as indicating positive polarity.

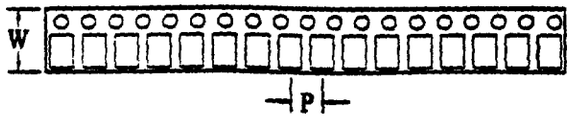
MARKING EXAMPLES BY CASE SIZE

Polarity indicated by either a White Stripe... or... a White "M"



PACKAGING
TAPE & REEL

■ 49MC Series capacitors are available in Tape & Reel packaging to facilitate the use of automatic placement equipment. Tape & Reel is per EIA (Electronic Industries Association) specification RS-481A. Capacitor orientation within tape pockets is negative terminal toward sprocket holes and mounting side down. 8 and 12 mm tape widths are used according to case size. Quantities less than a full reel will be shipped in Blister Pack trays.



MAXIMUM SOLDERING PROFILE

■ All MOLDED CHIP Capacitors may be exposed to (±5°C) for a period of 5 (±0.5) seconds. Soldering te atures in excess of 265°C and a duration of longer t 5.5 seconds are not recommended.

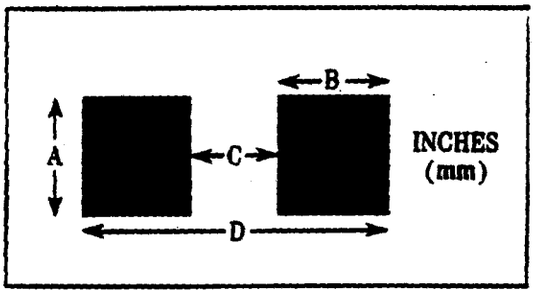
SOLDERABILITY INSPECTION CRITERIA

■ Magnification: 10X

Both termination end faces, excluding the areas of t terminations egressing from the package, shall be co with a smooth and bright solder coating with no mor than a small amount of scattered imperfections, such pinholes or un-wetted or de-wetted areas. The imperfect shall not be concentrated in one area.

Termination edges are not solderable surfaces.

RECOMMENDED PAD DIMENSIONS FOR REFLOW SOLDERING



Case Size	A Min.	B Nom.	C Nom.	D Nom.
A	.071 (1.80)	.085 (2.15)	.053 (1.35)	.223 (5.65)
B	.110 (2.80)	.085 (2.15)	.065 (1.65)	.235 (5.95)
C	.110 (2.80)	.107 (2.70)	.124 (3.15)	.337 (8.55)
D	.118 (3.00)	.107 (2.70)	.176 (4.45)	.389 (9.85)

TAPE & REEL SPECIFICATIONS

EIA Case Size	49MC Case Code	(W) Tape Width	(P) Unit Pitch Ctr/Ctr	Quantity per full 7" Reel	Quanti per ful 13" Ree
3216	A	8 mm	4 mm	2,000	9,000
3528	B	8 mm	4 mm	2,000	8,000
6032	C	12 mm	8 mm	500	3,000
7343	D	12 mm	8 mm	500	?

CASE C (CENTERED)

CAP, CASE C-TOP

Mike; How is this? It's size 1210 and gives us plenty of Murata Erie values to use if 1.5MF doesn't work

Meri

CHIPS–GRM Series to REPLACE TANTALUMS IN SURFACE MOUNT APPLICATIONS

These new chip ceramic capacitors are specifically designed to replace tantalum units in surface mount applications. They offer the long term reliability and stability inherent to ceramic devices and are specifically designed for bypassing applications. Their high frequency D.F. and ESR performance is considerably improved over that possible with tantalum.

FEATURES
- Long term reliability
- Higher breakdown voltage
- No polarity considerations
- Low D.F. and ESR at higher frequencies
- Equivalent performance with lesser values in many applications

***EIA Preferred Size**

MURATA ERIE DESIGNATION		GRM39	GRM40	GRM42-6	GRM42-2	
EIA TYPE DESIGNATION		0603	0805	1206	1210	1812
DIMENSIONS: in. (mm)	L	.060±.008 (1.5±0.2)	.080±.008 (2.0±0.2)	.125±.008 (3.2±0.2)	.125±.008 (3.2±0.2)	
	W	.030±.008 (0.75±0.2)	.050±.008 (1.25±0.2)	.063±.008 (1.5±0.2)	.100±.008 (2.5±0.2)	
	T max.	.035 (0.9)	.063 (1.35)	.060 (1.5)	.060 (1.5)	
	g min.	.020 (0.5)	.030 (0.75)	.040 (1.0)	.040 (1.0)	
	e	.014±.006 (0.35±0.2)	.020±.010 (0.5±0.25)	.020±.010 (0.5±0.25)	.020±.010 (0.5±0.25)	.020±.010 (0.5±0.25)
WVDC		16	16	16	16	16
Temperature Characteristic:		X7R / Y5V	X7R / Y5V	X7R / Y5V	X7R / Y5V	X7R / Y5V
Capacitance (μF)	.01					
	.012	.012 / .027	.018			
	.015					
	.018					
	.022					
	.027	.027				
	.033					
	.039					
	.047					
	.056					
	.068					
	.082					
	.1		.068			
	.12	.1			.12	
	.15			.15		
	.18					
	.22		.22	.22		
	.27					
	.33				.33	
	.39					
	.47					
	.56					.56
	.68				.68	
	.82					
	1.0			1.0		
	1.2					
	1.5					
	1.8					
	2.2					2.2

SUBCHIP CAPACITORS for LOW PROFILE & SUB-PLCC APPLICATIONS

FEATURES
- Sub-PLCC mounting of capacitor minimizes circuit inductance and allows higher packaging density.
- Available in standard Palladium/Silver (GR) or Nickel Barrier (GRM) Terminations.
- Available in Bulk or Tape & Reel.
- Reliability data available on request.
- Please contact factory for other values or dimensional requirements.

DIMENSIONS: in. (mm)

.125 ± .008 (3.2 ± 0.2)

.100 ± .008 (2.54 ± 0.2)

MURATA ERIE PART NUMBER	Value (μF)	Temp. Char.	WV	T max in. (mm)
GRM42-225Z5U1642025	.15	Z5U	25	.024 (0.6)
GRM42-2212Z5U2242025	.22	Z5U	25	.026 (0.66)
GRM42-226Z5U3342025	.33	Z5U	25	.028 (0.7)
GRM42-224Y5V1542025	.15	Y5V	25	.020 (0.5)
GRM42-225Y5V2242025	.22	Y5V	25	.024 (0.6)
GRM42-226Y5V3342025	.33	Y5V	25	.028 (0.7)

PART NUMBERING SYSTEM

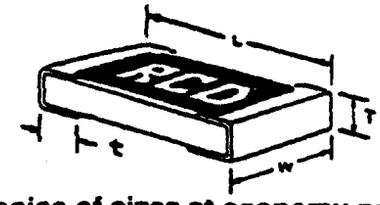
GRM42-221 Z5U 224 Z 025 A

- Capacitor type and size
- Variation code indicating thickness controlled design
- Temperature characteristic
- Nominal capacitance
- Tolerance (+80, −20%)
- 3-digit voltage rating
- Unmarked
- *Packaging code
  - *B=Bulk
  - *D=T/R (7" reel, paper tape)

GRM42- · Y5V 155 Z 016 (B)

1/16 WATT TO 3 WATT

MC SERIES



FEATURES

- Wraparound termination with *No Leach*™ nickel barrier
- Heavy solder plating facilitates the soldering process whether vapor phase, infrared reflow, or wave
- 1% and 5% chips available from stock (refer to pg. 4)
- **Available on exclusive *'SWIFT'* delivery program**
- ZC is zero-ohm jumper, MC is 200 ppm, MCR is 100 ppm
- 4 digit marking available on 1% (except 0603) 3 digit on 2%, 5%, 10% tolerances — 1210 | 122
- Untrimmed chips available
- Bondable chips available

Wide choice of sizes at economy prices!

RCD's chip resistors were designed to meet the stringer ronmental requirements of MIL-R-55342. The MC Series therefore offer significant performance improvements ov try average. Pricing remains extremely competitive due t automated production. Delivery of the international stanc 1206 size is from stock in all 1% and 5% values from 10 **RCD now offers a turnkey surface mount assembly se Why not consider us to assemble your next SM projec**

ACTUAL SIZE	RCD Type MC-,MCR-,ZC-	MC/MCR Wattage at 70°C	MC/MCR Dual Rating*	MC/MCR Voltage Rating	TYPE ZC Amp Rating	STANDARD Resistance Range	L (Length)	W (Width)	T (Thickness)	t (ter
New!→	0603	.0625	0.1	50V	NA	10Ω to 1MΩ	.061 ± .005 [1.55 ± .12]	.031 ± .004 [.8 ± .1]	.016 ± .006 [.40 ± .15]	.010 [.25
	0805	0.1	0.125	100V	1A	10Ω to 1 MΩ	.079 ± .006 [2.0 ± .15]	.050 ± .006 [1.25 ± .15]	.020 ± .006 [.50 ± .15]	.016 [.4
	1A (Stock Item)	0.125	0.25	200V	2A	2.2Ω to 22 MΩ	.126 ± .008 [3.2 ± .2]	.061 ± .006 [1.55 ± .15]	.024 ± .006 [.61 ± .15]	.020 [.51
	1206	0.25	0.30	200V	2A	3.3Ω to 10 MΩ	.126 ± .008 [3.2 ± .2]	.061 ± .006 [1.55 ± .15]	.024 ± .006 [.61 ± .15]	.020 [.51
	1210	0.25	0.50	200V	2A	10Ω to 2.2 MΩ	.126 ± .008 [3.2 ± .2]	.098 ± .008 [2.5 ± .2]	.024 ± .010 [.61 ± .25]	.020 [.5 ±
	2010	0.50	0.75	200V	NA	10Ω to 2.2 MΩ	.197 ± .008 [5.0 ± .2]	.102 ± .008 [2.6 ± .2]	.025 ± .010 [.63 ± .25]	.020 [.50 ±
	2512	1.0	1.5	200V	3A	10Ω to 2.2 MΩ	.245 ± .015 [6.22 ± .38]	.125 ± .010 [3.2 ± .25]	.025 ± .010 [.63 ± .25]	
	4020**	2.0	3.0	350V	NA	10Ω to 2.2 MΩ	.394 ± .01 [10.0 ± .25]	.197 ± .010 [5.0 ± .25]	.028 ± .012 [.71 ± .3]	.035 [.89 ±

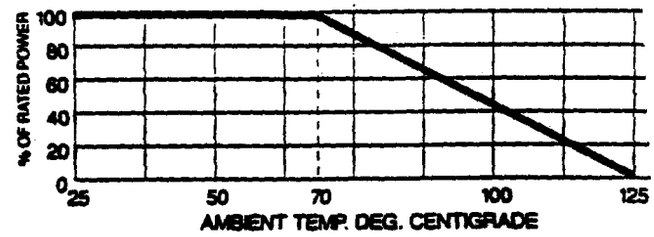
*Chips may be operated up to the dual rated level with consideration of mounting density, P.C. board material and ambient temperature to control self-heating to 125°C max.
**Information on MC 4020 is preliminary. Consult factory for availability.

PERFORMANCE CHARACTERISTICS

Characteristic	Test Method	Max. ΔR + .05Ω (10Ω to 1MΩ range)
Temperature Coefficient (Typ.)	-55 to + 125°C	±100 or ±200¹ppm
Thermal Shock (-55° to + 150°C)	MIL-Std-202, M 107	0.5%
Short Time Overload (2.5 × 5 sec)²	M 55342 p. 4, 7, 5	0.5%
Low Temp. Operation (-55°C)	M 55342 p. 4, 7, 4	0.5%
High Temp. Exposure (125°C, 100 hrs)	M 55342 p. 4, 7, 6	0.5%
Resistance to Bonding Exposure	M 55342 p. 4, 7, 7	0.25%
Moisture Resistance	MIL -Std-202, M 106	0.5%
Load Life (1000 hours)	MIL-Std-202, M 108	1.0%*
Solderability	MIL-Std-202, M 208	95% Min. Coverage

*Load life for dual rating is ± 6% + .05Ω Max.
¹"MCR" is 100ppm, "MC" is 200ppm 10Ω-1M, 400ppm <10Ω or > 1M
²STD test at 2× rated power on MC1206 type. STD test does not apply to dual wattage ratings.

DERATING CURVE



HOW TO ORDER:

MCR 1206 | 100Ω (ohm) | 1% | T

RCD Type
(MC is 200 ppm, MCR is 100 ppm, ZC is chip jumper)
Resistance Value (.05Ω Max. on ZC type)
Tolerance (±1%, ±2%, ±5%, ±10%)
Packaging - 'B' is bulk, 'T' is Tape & Reel, 'M' is Magazine Cartridge
Also advise at the time of ordering whether marking on the chip is required. (RCD option if not specified by customer)

PACKAGING

Type MC-, MCR-, ZC-	Packaging Options*
0603, 0805	bulk, 8mm tape
1A, 1206	bulk, magazine, 8mm tape
1210	bulk, 8mm tape
2010, 2012, 2512	bulk, 12mm tape
4020	bulk, 16mm tape

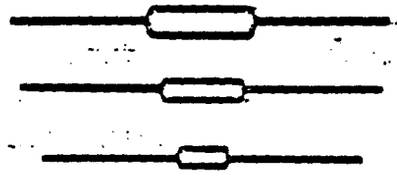
*Standard 8mm tape is paper, other sizes are embossed plastic. Plastic tape and reels c on 8mm size but must be requested at the time of ordering.

RCD Components, Inc., 520 East Industrial Park Dr., Manchester, NH USA 03103
Tel: (603) 669-0054 Fax: (603) 669-5455 Tlx: 943512

45, 47-51, 53-58, 64, 65

OHMITE® AXIAL LEAD WIREWOUND RESISTORS

20 SERIES VITREOUS ENAMEL CONFORMAL COATING RESISTORS



Rugged, economical wirewound resistors for industrial applications. ±5% tolerance. 1kV dielectric breakdown voltage (500V for 1 watt size.) Typical temperature coefficient for 10 ohm and above: ±30 ppm/°C.

20 SERIES STANDARD OHMIC VALUES

1.0	12	36	82	250	500	1100	3300	7000	20000
1.5	15	39	100	270	560	1200	3500	7500	22000
2.0	18	40	120	300	600	1500	3900	8000	25000
2.2	20	47	125	330	680	1800	4000	9000	35000
3.0	22	50	150	350	750	2000	4600	10000	40000
4.5	25	56	180	360	800	2200	4700	12000	50000
5.0	27	62	200	400	820	2500	5000	13000	.....
7.5	30	68	220	450	900	2700	6000	15000	.....
10	33	75	225	470	1000	3000	6800	.....	.....

21J SERIES 1 WATT
Dimensions: .406"L x .156"D

Stock No.	Ohms	1-24	25-99	100-249
13F2000	1.0 to 7.5	2.57	2.44	2.01
13F2004	10 to 82	2.51	2.14	1.77
13F2008	100 to 909	2.77	2.36	1.94
13F2012	1K to 1.8K	2.84	2.41	1.99
13F2019	2K to 3K	2.84	2.41	1.99

22J SERIES 2 WATT
Dimensions: .562"L x .250"D

Stock No.	Ohms	1-24	25-99	100-249
13F2000	1.0 to 7.5	1.16	1.01	.82
13F2001	10 to 470	1.09	.84	.70
13F2001	500 to 900	1.26	1.07	.89
13F2001	1K	1.44	1.22	1.02
13F2005	1.1K to 2.7K	1.44	1.22	1.02
13F2002	3K to 4.7K	1.37	1.16	1.02
13F2003	5K	1.84	1.76	1.49
13F2005	6K to 10K	2.09	1.78	1.48

23J SERIES 3 WATT
Dimensions: 1.0"L x .250"D

Stock No.	Ohms	1-24	25-99	100-249
13F2004	1.0 to 7.5	1.34	1.09	.87
13F2005	10 to 470	1.10	.84	.67
13F2006	500 to 900	1.34	1.13	.94
13F2008	1K	1.66	1.33	1.09
13F2009	1.1K to 2.5K	1.66	1.33	1.09
13F2006	3K to 4.7K	1.66	1.33	1.07
13F2008	5K	2.09	1.67	1.34
13F2007	6K to 10K	2.09	1.67	1.34
13F2008	12K to 18K	2.09	1.67	1.34
13F2005	20K to 25K	2.09	2.34	1.84

26J SERIES 10 WATT
Dimensions: 1.797"L x .343"D

Stock No.	Ohms	1-24	25-99	100-249
13F2000	1.0 to 7.5	1.78	1.51	1.34
13F2001	10 to 18	1.69	1.52	1.19
13F2001	20 to 470	1.69	1.69	1.11
13F2001	500 to 900	1.82	1.69	1.32
13F2001	1K	2.09	1.78	1.49
13F2002	1.2K to 2.7K	2.09	1.78	1.49
13F2003	3K to 4.7K	2.09	1.69	1.49
13F2005	5K	2.09	2.09	1.69
13F2004	6K to 7.5K	2.09	2.09	1.69
13F2005	12K to 18K	2.09	2.09	1.69
13F2006	20K	2.11	2.64	2.19
13F2008	22K to 50K	2.11	2.64	2.19

40 SERIES SILICONE-CERAMIC CONFORMAL

Silicone-ceramic conformal axial lead resistors with ±5% tolerance, 1000VAC dielectric breakdown voltage. Temperature coefficient: less than 1 ohm - 60 ppm/°C; 1 to 10 ohms - ±40 ppm/°C; 10 ohms and greater - ±30 ppm/°C.

40 SERIES STANDARD OHMIC VALUES

.1	.5	10.0	50.0	1K
.15	.75	12.0	100	2K
.2	1.0	15.0	270	3.9K
.3	5.0	22.0	330	10.0K
.....	.....	.....	360	.....

40J SERIES 2 WATT
Dimensions: .562"L x .250"D

Stock No.	Ohms	1-24	25-99	100-249
44F0212	0.1 to 0.2	1.34	1.09	1.38
44F0213	0.3	1.58	1.19	.88
44F0215	0.5 to 0.75	1.58	1.19	.88
44F0220	1.0 to 5.0	.88	.88	.88
44F0214	10 to 300	.88	.79	.77
44F0216	1K	1.58	1.58	.84
44F0218	2K	1.58	1.58	.84
44F0219	3.9K	1.58	1.34	2.02
44F0217	10K	1.58	1.44	1.19

40J SERIES 5 WATT
Dimensions: .937"L x .343"D

Stock No.	Ohms	1-24	25-99	100-249
44F0210	0.1 to 0.2	2.32	1.98	1.83
44F0216	0.3	1.98	1.41	1.19
44F0219	0.5 to 0.75	1.98	1.41	1.19
44F0220	1.0 to 5.0	1.19	.99	.81
44F0221	10 to 300	.88	.88	.88
44F0241	2K	1.98	1.51	1.59
44F0241	3.9K	1.74	1.49	1.32
44F0222	10K	2.51	1.71	1.49

40J SERIES 10 WATT
Dimensions: 1.843"L x .406"D

Stock No.	Ohms	1-24	25-99	100-249
44F0223	0.1 to 0.2	2.78	2.37	1.98
44F0225	0.3	2.58	1.78	1.58
44F0224	0.5 to 0.75	2.58	1.78	1.58
44F0226	1.0 to 5.0	1.58	1.19	.88
44F0240	10 to 300	1.19	1.09	.88
44F0242	1K	1.71	1.48	1.58
44F0243	2K	1.71	1.48	1.58
44F0244	3K to 3.9K	2.58	1.77	1.48
44F0213	10K	2.41	2.09	1.88

50 SERIES SILICONE-CERAMIC-CONFORMAL

All welded construction designed for high stability precision power applications. Low temperature coefficient: 20 ppm/°C for 10 ohms and above. Dielectric withstanding voltage: 1000VAC. ±1% tolerance standard. Meet MIL-R-26 requirements.

STANDARD RESISTANCE - OHMS

.1	1.0	15.0	100.0	1K	24.9
.15	1.5	20.0	121.0	3.01K	30.1K
.20	2.0	24.9	200.0	3.32K	40.2K
.300	4.02	56.2	301.0	4.02K	49.9K
.499	4.99	61.9	499	5.11K	
.75	10.0	75.0	909.0	10K	

50F SERIES 5 WATT
Dimensions: .937"L x .219"D

Stock No.	Ohms	1-9	10-49	50-99
44F0227	01. to 0.2	2.49	2.35	2.41
44F0227	.75	2.05	2.05	1.73
44F0227	1.0	1.99	1.99	.99
44F0228	4.99	1.99	1.99	.99
44F0228	15 to 499	1.99	.99	.79
44F0228	909	1.57	1.16	.98
44F0228	1K	1.99	1.34	1.16
44F0228	3.01K to 4.02K	1.78	1.99	1.53
44F0228	5.11K to 10K	2.84	1.74	1.43

50F SERIES 5 WATT
Dimensions: .937"L x .343"D

Stock No.	Ohms	1-9	10-49	50-99
44F0229	0.1 to 0.2	4.19	3.48	2.97
44F0229	.75	2.84	2.51	1.98
44F0229	1.0	1.84	1.51	1.98
44F0229	2.0 to 4.99	1.84	1.51	1.98
44F0229	10.0 to 499	1.98	1.98	.98
44F0229	1K	1.98	1.98	1.32
44F0229	3.01K to 4.02K	1.98	1.98	1.32
44F0229	10K	2.18	1.78	1.57
44F0229	24.9K to 30.1K	2.98	2.57	1.71
44F0229	40.2K to 49.9K	2.98	2.57	1.71

50F SERIES 10 WATT
Dimensions: 1.843"L x .406"D

Stock No.	Ohms	1-9	10-49	50-99
44F0231	0.1 to 0.2	4.98	4.19	3.36
44F0231	.75	2.84	2.51	1.98
44F0232	1.0 to 4.99	1.84	1.98	1.98
44F0232	10.0 to 301	1.98	1.98	1.98
44F0232	1.0K	2.98	1.51	1.98
44F0232	3.01K to 4.02K	2.98	2.14	1.98
44F0232	5.11K to 10K	2.98	2.98	1.98
44F0234	24.9K to 30.1K	2.98	2.98	2.98
44F0230	40.2K to 49.9K	2.98	2.48	2.98

:OSTATS

r RHEOSTATS

E Series Rheostat

...ric cooling. Operate at maximum hot ...t of 40°C. 300° rotation (±6°). .519" ...t with ¼–32 bushing and hex nut on ...diameter ¾". *Order knob type 5161.*

Amps	1-4	5-9
.55 .39 .27 .12	45.17	36.33

VCLOSED MINIATURE ...ATS

...aximum hot spot temperature rise of ...r operating at high military ambients. ...sistance. Noise reference element ...over ...1-ceramic mounting, 200° ro... ...tion. ±10%. Mount with ¼–32 ...hardware. ...diameter ¾". ...KB7 separately below.

Amps	1-9	10-24
3.630 2.800 2.040 1.440 1.230	22.30	16.61
1.120 0.910 0.710 0.600 0.500	22.30 22.30 19.82 19.82 19.82	16.61 16.61 15.48 15.48 15.48
0.410 0.380 0.330 0.270 0.220	19.82	15.48
0.190 0.160 0.130 0.100 0.080 0.070	19.82 19.82 19.82 20.80 20.80 20.80	15.48 15.48 15.48 14.84 14.84 14.84
0.060 0.050 0.041 0.036 0.029	21.20 21.20 22.50 22.50 22.50	15.88 15.88 16.89 16.89 16.89

DIAL PLATES

...s are etched on a black background. ...e approximate percentage of rheo... ...in of ... Use type 5001 with knob ... 5 ... pe 5000 with knob type ... with ... type 5161.

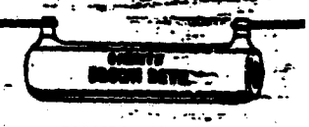
	Type	Each
	5001	5.08
	5000	1.34
	5007	1.34

WIREWOUND RESISTORS

OHMITE

200 SERIES "BROWN DEVIL" VITREOUS ENAMEL WIREWOUND POWER RESISTORS

High quality, compact wirewound resistors for voltage-dropping, bias units, blowers, etc. Extra-sturdy all-ceramic vitreous enameled. Protected against shock, vibration, heat and humidity. Have standard tolerance of ±5%. 5-watt units have 1¼" overall width, 1¼" high; 10-watt units, 2¼" overall. All have 1¼" leads.



13F150 B5J SERIES —5 WATT RESISTORS

Type	Ohms	Price Code	Type	Ohms	Price Code	Type	Ohms	Price Code	Type	Ohms	Price Code
B5J1R0	1	A	B5J50R	30	B	B5J500	500	B	B5J4K0	4,000	C
B5J1R5	1.5	A	B5J40R	40	B	B5J600	600	B	B5J5K0	5,000	C
B5J2R0	2	A	B5J50R	50	B	B5J1K0	1,000	B	B5J6K0	6,000	D
B5J3R0	3	A	B5J75R	75	B	B5J1K2	1,200	C	B5J7K0	7,000	D
B5J4R0	4	A	B5J100	100	B	B5J1K25	1,250	C	B5J7K5	7,500	D
B5J5R0	5	A	B5J125	125	B	B5J1K5	1,500	C	B5J8K0	8,000	D
B5J7R5	7.5	A	B5J150	150	B	B5J2K0	2,000	C	B5J10K	10,000	D
B5J10R	10	B	B5J200	200	B	B5J2K25	2,250	C	B5J12K5	12,500	D
B5J12R	12	B	B5J250	250	B	B5J2K5	2,500	C	B5J15K	15,500	D
B5J15R	15	B	B5J300	300	B	B5J3K0	3,000	C	B5J20K	20,000	E
B5J20R	20	B	B5J350	350	B	B5J3K5	3,500	C	B5J25K	25,000	E
B5J25R	25	B	B5J400	400	B						

B5J SERIES PRICE CODE

Price Code	1-24	25-99	100-249
A	1.05	1.00	1.32
B	1.55	1.32	1.05
C	1.63	1.56	1.14
D	1.99	1.53	1.29
E	2.11	1.75	1.48

13F155 B12J SERIES —12 WATT RESISTORS

Type	Ohms	Price Code	Type	Ohms	Price Code	Type	Ohms	Price Code	Type	Ohms	Price Code
B12J1R0	1	F	B12J40R	40	A	B12J700	700	A	B12J4K5	4,500	F
B12J1R5	1.5	F	B12J50R	50	A	B12J750	750	A	B12J5K0	5,000	F
B12J2R0	2	F	B12J75R	75	A	B12J900	900	A	B12J6K0	6,000	G
B12J3R0	3	F	B12J100	100	A	B12J1K0	1,000	A	B12J7K0	7,000	G
B12J4R0	4	F	B12J125	125	A	B12J1K2	1,200	F	B12J7K5	7,500	G
B12J5R0	5	F	B12J150	150	A	B12J1K25	1,250	F	B12J8K0	8,000	G
B12J7R5	7.5	F	B12J200	200	A	B12J1K5	1,500	F	B12J10K	10,000	G
B12J10R	10	F	B12J250	250	A	B12J1K75	1,750	F	B12J15K	15,000	G
B12J12R	12	F	B12J300	300	A	B12J2K0	2,000	F	B12J20K	20,000	H
B12J15R	15	F	B12J400	400	A	B12J2K25	2,250	F	B12J25K	25,000	H
B12J20R	20	F	B12J500	500	A	B12J2K5	2,500	F	B12J30K	30,000	I
B12J25R	25	F	B12J600	600	A	B12J3K0	3,000	F	B12J40K	40,000	I
B12J30R	30	F				B12J3K5	3,500	F	B12J45K	45,000	I
B12J35R	35	F				B12J4K0	4,000	F	B12J50K	50,000	I

B12J SERIES PRICE CODE

Price Code	1-24	25-99	100-249
A	1.05	1.00	1.32
F	2.19	1.89	1.63
G	2.44	2.09	1.80
H	2.61	2.19	1.89
I	3.09	2.69	2.19

13F160 B20J SERIES —20 WATT RESISTORS

Type	Ohms	Price Code	Type	Ohms	Price Code	Type	Ohms	Price Code	Type	Ohms	Price Code
B20J1R0	1	AA	B20J75R	75	BB	B20J500	500	BB	B20J5K0	5,000	CC
B20J2R0	2	AA	B20J100	100	BB	B20J600	600	BB	B20J10K	10,000	DD
B20J3R0	3	AA	B20J150	150	BB	B20J750	750	BB	B20J15K	15,000	FF
B20J4R0	4	AA	B20J200	200	BB	B20J1K0	1,000	BB	B20J20K	20,000	FF
B20J5R0	5	AA	B20J250	250	BB	B20J1K2	1,200	CC	B20J25K	25,000	FF
B20J10R	10	BB	B20J300	300	BB	B20J1K5	1,500	CC	B20J50K	50,000	FF
B20J25R	25	BB	B20J350	350	BB	B20J2K0	2,000	CC	B20J100K	100,000	GG
B20J50R	50	BB	B20J400	400	BB	B20J3K0	3,000	CC			

B20J SERIES PRICE CODE

Price Code	1-24	25-99	100-249
AA	2.04	2.00	2.12
BB	2.32	1.98	1.84
CC	2.84	2.19	1.79
DD	3.57	2.49	2.09
FF	5.15	3.89	3.19
GG	5.85	3.95	3.79

99 SERIES MOLDED VITREOUS ENAMEL WIREWOUND RESISTORS ±5% TOLERANCE

99J SERIES 5% WATT

Stock No.	Ohms	Dimensions	1-24	25-99	100-249
13F145	1.0 to 7.5 10.0 to 1k 1.2k to 6k 6.8k to 10k	.565"Lx.234"D	1.56 1.14 1.68 2.34	1.16 .97 1.41 1.99	.96 .80 1.17 1.64

99J SERIES 8 WATT

Stock No.	Ohms	Dimensions	1-24	25-99	100-249
13F146	1.0 to 7.5 10.0 to 1k 1.2k to 6k 6.8k to 10k 12K to 20K 22K to 25K	.865"Lx.234"D	1.82 1.48 2.62 2.57 2.57 2.42	1.37 1.22 1.72 2.44 2.44 2.91	1.13 1.01 1.41 2.01 2.01 2.40

99J SERIES 11 WATT

Stock No.	Ohms	Dimensions	1-24	25-99	100-249
13F147	1.0 to 7.5 10.0 to 1k 1.2k to 6k 6.8k to 10k 12k to 20k 22k to 40k 47k to 50k	1.796"Lx.343"D	2.49 2.19 2.78 3.59 3.59 4.31 4.31	2.09 1.85 2.33 3.09 3.09 3.57 3.57	1.86 1.53 1.93 2.54 2.54 2.94 2.94

Axial-lead resistors with all-welded, molded construction and flame resistant vitreous enamel coating. Markings resist high temperatures, solvents and abrasions. Standard resistance tolerance: ±5%. Consistent term perature measuring in clips with heat sink benefits up to 100%. Meets MIL-R-26 requirements for "insulated" resistors. Dielectric withstanding voltage: 1000 VAC: 2, 3, 5, 7 & 10 watt rating; 500 VAC: 1 watt rating. *Mounting hardware which exceeds power ratings available separately below.*

OHMIC VALUES FOR 99 SERIES RESISTORS

1.0	10	30	68	200	400	750	2200	4700	12000
1.5	12	33	75	220	470	800	2500	5000	15000
2.0	15	39	82	250	500	1000	2700	5600	18000
3.0	20	40	100	270	560	1200	3000	6000	20000
4.0	22	47	120	300	600	1500	3300	6800	22000
5.0	25	50	150	330	680	1800	3600	7500	25000
7.5	27	56	180	390	700	2000	4000	10000	50000

99J SERIES 1% WATT

Stock No.	Ohms	Dimensions	1-24	25-99	100-249
13F160	1.0 to 7.5 10.0 to 1k 1.2k to 3k	.437"Lx.140"D	3.35 2.64 2.57	2.35 2.35 2.53	2.34 1.88 2.09

99 SERIES RESISTOR MOUNTING HARDWARE CLIPS NEW

Stock No.	Type	Watts	Dimensions	Holes	1-9	10-99	100-249
99F7926	9900	1¼	.40"Lx.160"W	1	.94	.71	.56
99F7929	9904	3¼	.50"Lx.257"W	2	.94	.71	.56
99F7927	9906	5	.50"Lx.257"W	2	.94	.71	.56
99F7794	9908	11	1.75"Lx.333"W	2	.94	.71	.56

GENERAL SPECIFICATIONS

ELECTRICAL	
Resistance range	10Ω~2MΩ
Resistance tolerance	±20%
Power ratings	0.25W (70°C), 0W (125°C)
Max. input voltage	200V
Max. wiper current	100mA
Electrical angle, nominal	210°
End resistance, max.	1% of resistance value or 2Ω, whichever is greater
Contact resistance variation	1% of resistance value or 3Ω, whichever is greater
Operating temperature	−55°C~125°C
Temp. coefficient, max.	10Ω to 50Ω: ±250ppm/°C, 100Ω~2MΩ: ±100ppm/°C
Insulation resistance, min.	1,000MΩ (DC500V)
Dielectric strength	500Vrms (1 minute)
MECHANICAL	
Mechanical angle, nominal	240°
Operating torque, max.	150gcm
Stop strength, min.	350gcm

ENVIRONMENTAL (MIL-R-22097/MIL-STD-202)

Item	Test conditions	ΔR/R	S.S.
Thermal shock	−65°C~125°C, 5 cycles	±2%	±1%
Humidity	80~98%RH, 10 cycles, 240 hrs.	±2%	–
Shock	100G, 6 directions, 3 times	±1%	±1%
Vibration	20G, 10~2,000Hz, 12 hrs.	±1%	±1%
Soldering heat	260°C 10 sec./or 215° 3 min.	±1%	–
Load life	70°C, rated power, 1,000 hrs.	±3%	±1%
Low temp. operation	−55°C, 2 hrs.	±2%	±2%
High temp. exposure	125°C, 250 hrs.	±3%	±2%
Rotational life	100 cycles	±(2Ω +3%)	–

ΔR/R: Change in total resistance, S.S.: setting stability

PACKAGING

ST-4A & B 1. Reeled tape (500 pcs per reel) tape width:12mm part pitch : 8mm
2. Plastic magazine (50 pcs per stick)
3. Vinyl bag (in bulk) (100 pcs per bag)

ST-4C 1. Plastic magazine (50 pcs per stick)
2. Vinyl bag (in bulk) (100 pcs per bag)

STANDARD RESISTANCE VALUES

CODE	OHMS	CODE	OHMS	CODE	OHMS	CODE	OHMS	CODE	OHMS	CODE	OHMS
100	10	101	100	102	1K	103	10K	104	100K	105	1M
200	20	201	200	202	2K	203	20K	204	200K	205	2M
500	50	501	500	502	5K	503	50K	504	500K		

APPLICATION NOTES

Parts will meet original specifications after exposure to:

Reflow Soldering: . . . . . Peak Temperature=240°C
Solder Melting Point Temp.=180°C
Time @ Melting Point=10 seconds maximum

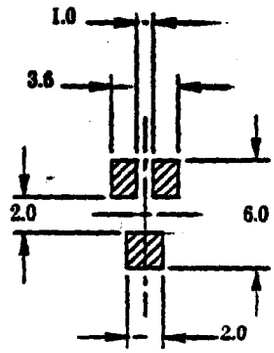
Vapor Phase Soldering: . . . 180 seconds maximum @ 215°C or
10 seconds maximum @ 260°C

Wave Soldering: . . . . . Peak Temperature=260°C
Time in Solder Wave=4 seconds maximum

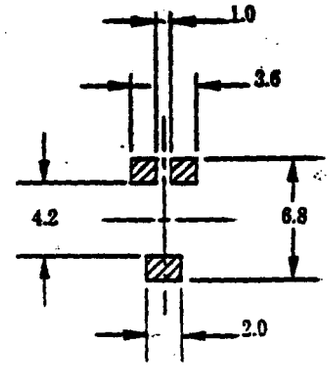
Manual Soldering: . . . . . Soldering Iron Tip Temp.=350°C maximum
Soldering Time=3 seconds maximum
Soldering Iron Wattage=40 watts maximum

P.C. BOARD PAD OUTLINE

ST-4A



ST-4B

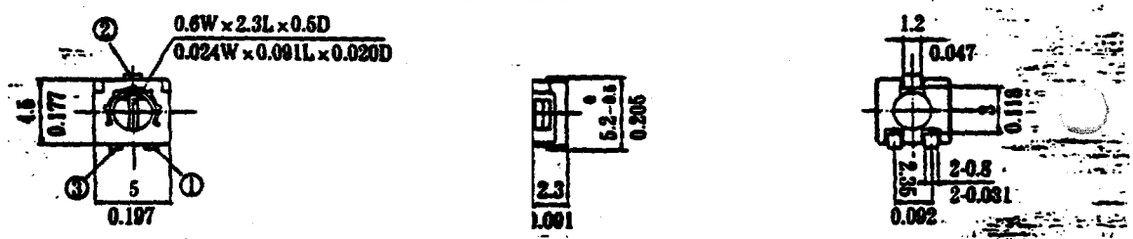


METRIC/ENGLISH CONVERSION TABLE	
mm	inches
0.3	.012
1.0	.039
2.0	.079
3.6	.142
4.2	.165
6.0	.236
6.8	.268

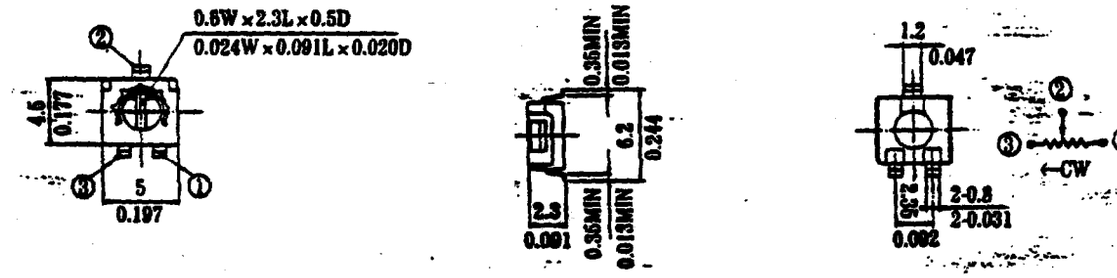
NOTES:
1. ALL DIMENSIONS ARE IN MILLIMETERS.
2. DIMENSIONAL TOLERANCES ARE ±0.3 UNLESS OTHERWI SPECIFIED.

DIMENSIONS (IN MM/INCH)

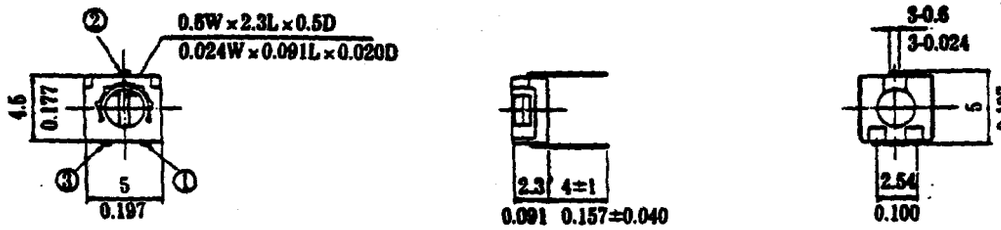
ST-4A
Top adjustment
J-Lead



ST-4B
Top adjustment
Gull Wing



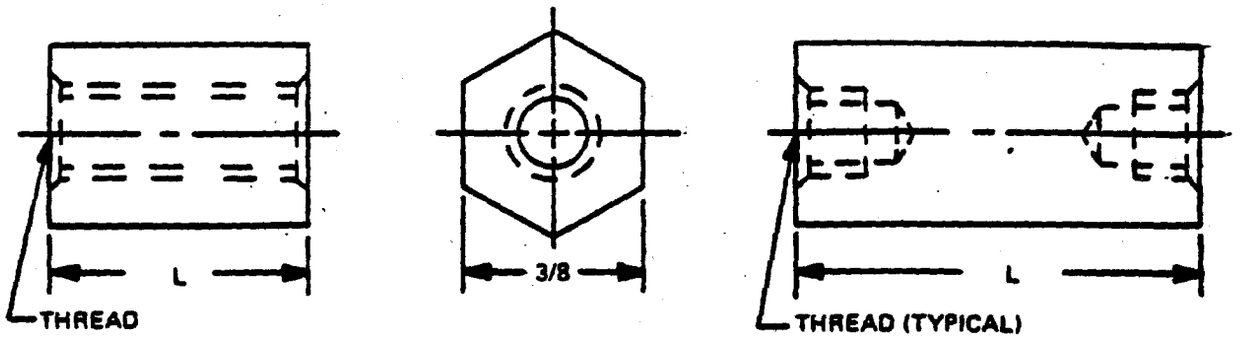
ST-4C
Top adjustment
Through Hole



Tolerances are ±0.3 mm and ±.012 inches

(63)

3/8 HEX FEMALE STANDOFFS



ORDERING CODE

XXXX XXXX X X

LENGTH | THREAD | MATERIAL | FINISH

See page [7] for thread depth
See page [6] for finish code
For parts not listed contact Sales Office

LENGTH	PART NO.
1/8	2232
3/16	2233
1/4	2234
5/16	2235
3/8	2236
7/16	2237
1/2	2238
9/16	2239
5/8	2240
11/16	2241
3/4	2242
13/16	2243
7/8	2244
15/16	2245
1"	2246
1-1/16	2247
1-1/8	2248
1-3/16	2249
1-1/4	2250
1-5/16	2251
1-3/8	2252
1-7/16	2253
1-1/2	2254
1-9/16	2255

LENGTH	PART NO.
1-5/8	2256
1-11/16	2257
1-3/4	2258
1-13/16	2259
1-7/8	2260
1-15/16	2261
2"	2262
2-1/8	2263
2-1/4	2264
2-3/8	2265
2-1/2	2266
2-5/8	2267
2-3/4	2268
2-7/8	2269
3"	2270
3-1/4	2271
3-1/2	2272
3-3/4	2273
4"	2274
4-1/4	2275
4-1/2	2276
4-3/4	2277
5"	2278
5-1/4	2279

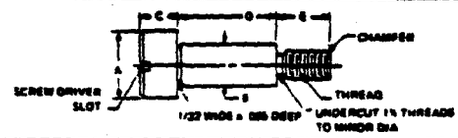
LENGTH	PART NO.
5-1/2	2280
5-3/4	2281
6"	2282
6-1/4	2283
6-1/2	2284
6-3/4	2285
7"	2286
7-1/4	2287
7-1/2	2288
7-3/4	2289
8"	2290
8-1/4	2291
8-1/2	2292
8-3/4	2293
9"	2294
9-1/4	2295
9-1/2	2296
9-3/4	2297
10"	2298

THREAD	CODE
6-32	632
8-32	832
10-32	1032
12-24	1224
1/4-20	2520

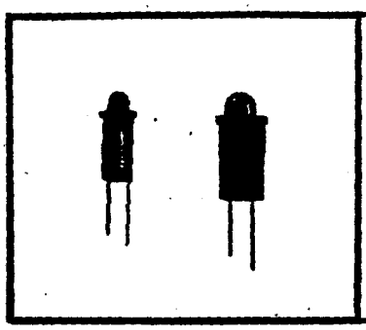
MATERIAL	CODE
Aluminum	A
Brass	B
Stainless Steel	SS
Steel	S
Nylon	N

PRECISION SHOULDER SCREWS—SLOTTED HEAD

See page [6] for finish code
For parts not listed
contact Sales Office



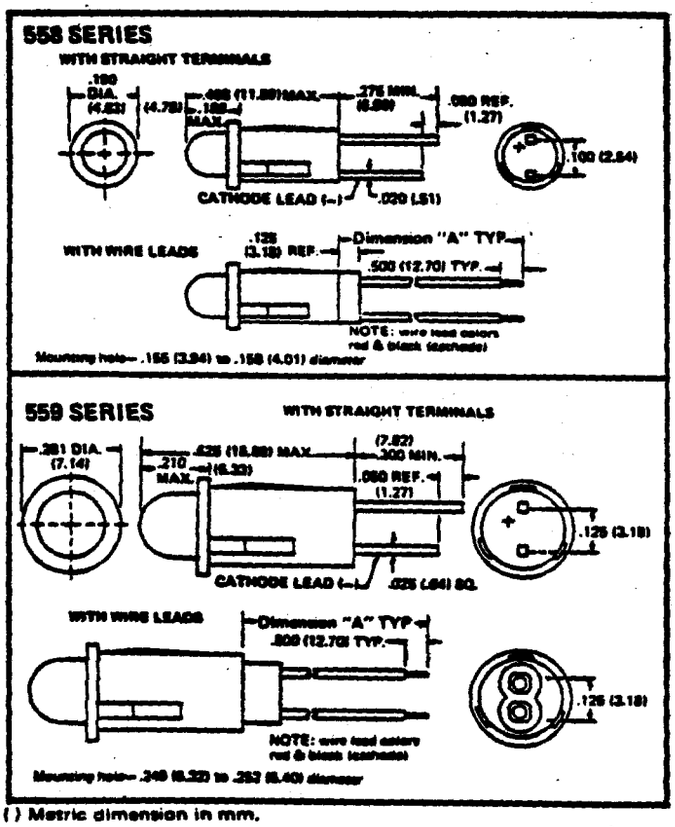
PART NO.	THREAD	A ±.004	B +.000 −.001	C ±.010	D +.002 −.000	E ±.005
7035	10-32	3/8	.2495	3/16	.1250	1/4
7036	10-32	3/8	.2495	3/16	.1875	1/4
7037	10-32	3/8	.2495	3/16	.2500	1/4
7038	10-32	3/8	.2495	3/16	.3125	1/4
7039	10-32	3/8	.2495	3/16	.3750	1/4
7040	10-32	3/8	.2495	3/16	.4375	1/4
7041	10-32	3/8	.2495	3/16	.5000	1/4
7042	10-32	3/8	.2495	3/16	.5625	1/4
7043	10-32	3/8	.2495	3/16	.6250	1/4
7044	10-32	3/8	.2495	3/16	.6875	1/4
7045	10-32	3/8	.2495	3/16	.7500	1/4
7046	10-32	3/8	.2495	3/16	.8750	1/4
7047	10-32	3/8	.2495	3/16	1.000	1/4
7048	10-32	3/8	.2495	3/16	1.250	1/4
7049	10-32	3/8	.2495	3/16	1.375	1/4
7050	10-32	3/8	.2495	3/16	1.500	1/4
7051	1/4-20	7/16	.3120	7/32	.2500	3/8
7052	1/4-20	7/16	.3120	7/32	.3125	3/8
7053	1/4-20	7/16	.3120	7/32	.3750	3/8
7054	1/4-20	7/16	.3120	7/32	.5000	3/8
7055	1/4-20	7/16	.3120	7/32	.6250	3/8
7056	1/4-20	7/16	.3120	7/32	.7500	3/8
7057	1/4-20	7/16	.3120	7/32	.8750	3/8
7058	1/4-20	7/16	.3120	7/32	1.000	3/8
7059	1/4-20	1/2	.3745	7/32	.2500	3/8
7060	1/4-20	1/2	.3745	7/32	.3125	3/8
7061	1/4-20	1/2	.3745	7/32	.3750	3/8
7062	1/4-20	1/2	.3745	7/32	.5000	3/8
7063	1/4-20	1/2	.3745	7/32	.6250	3/8
7064	1/4-20	1/2	.3745	7/32	.7500	3/8
7065	1/4-20	1/2	.3745	7/32	.8750	3/8
7066	1/4-20	1/2	.3745	7/32	1.000	3/8
7067	5/16-18	1/2	.3745	7/32	.2500	7/16
7068	5/16-18	1/2	.3745	7/32	.3125	7/16
7069	5/16-18	1/2	.3745	7/32	.3750	7/16
7070	5/16-18	1/2	.3745	7/32	.5000	7/16
7071	5/16-18	1/2	.3745	7/32	.6250	7/16
7072	5/16-18	1/2	.3745	7/32	.7500	7/16
7073	5/16-18	1/2	.3745	7/32	.8750	7/16
7074	5/16-18	1/2	.3745	7/32	1.000	7/16
7075	3/8-16	5/8	.4995	1/4	.2500	1/2
7076	3/8-16	5/8	.4995	1/4	.3125	1/2
7077	3/8-16	5/8	.4995	1/4	.3750	1/2
7078	3/8-16	5/8	.4995	1/4	.5000	1/2
7079	3/8-16	5/8	.4995	1/4	.6250	1/2
7080	3/8-16	5/8	.4995	1/4	.7500	1/2
7081	3/8-16	5/8	.4995	1/4	.8750	1/2
7082	3/8-16	5/8	.4995	1/4	1.000	1/2



Snap-in LED Indicators

.156" and .250" mounting hole

558 & 559 series



( ) Metric dimension in mm.

558 Series: Mount in .156" hole
559 Series: Mount in .250" hole

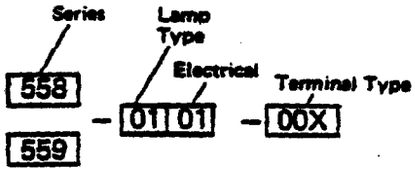
Features

- Snap-in mounting requires no additional hardware
- Available with Red, Green or Yellow LEDs, with and without integral current limiting resistors
- Compact design allows high density packaging:
  558 Series mounts on 0.2" centers
  559 Series mounts on 0.3" centers
- Straight terminals suitable for wire-wrapping and/or soldering, or wire leads
- Designed for quick positive insertion in panels from .031" through .062"
- Black housing enhances contrast ratio
- Polarity identified
- High Brightness LEDs
- Wide Angle visibility
- Low power requirements-10-20 mA
- Solid State reliability
- IC compatible
- Vibration and shock resistant

Applications

- Computers
- Process controllers
- Instrumentation
- Point of sale
- Communications equipment
- Medical equipment
- Home entertainment equipment
- Vending machines

Ordering Information



Lamp type (LED color)

Standard Efficiency	
01	Red diffused
02	Green diffused
03	Yellow diffused

High Efficiency*	
21	Red diffused
22	Green diffused
23	Yellow diffused

* 559 only

Electrical

01	Requires external resistor
02	5V, 15mA (01, 02, 03 lamp only)
03	12V, 15mA (01 lamp only-559 only)

Terminal type

001	Straight Terminals
Dimension "A" wire leads	
003	6.000 (152.40) Typ.
004	8.000 (203.20) Typ.
005	10.000 (254.00) Typ.
006	12.000 (304.80) Typ.
007	14.000 (355.60) Typ.

Standard Efficiency snap-in LED indicators 558 and 559 Series

Operating Characteristics — 558 and 559 Series without Integral Resistor

Symbol	Characteristics	RED			GREEN			YELLOW			Units		TEST CONDITIONS		
		Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.			RED	GREEN	YELLOW
I_V	Luminous Intensity: 558 Series	.3	2.4		.8	2.0		1.0	2.5		mcd	I_F	20mA	20mA	10mA
	559 Series	.8	1.6		1.6	3.2		1.6	3.2				10mA		20mA
λ_{pk}	Wavelength: 558 Series		655			585			590		nm				
	559 Series		700												
V_F	Forward Voltage: 558 Series		1.6	2.0		2.4	3.0		2.2	3.0	V	I_F	20mA	20mA	10mA
	559 Series		1.9	2.4									5mA		20mA
BV_R	Reverse breakdown voltage	3.0			5.0			3.0			V	I_R	100µA* 10µA	10µA	10µA

*558-0101-XXX

Maximum Ratings — 558 and 559 Series without Integral Resistor

Characteristics	RED				GREEN		YELLOW		Units	TEST CONDITIONS
	558 Series		559 Series		Min.	Max.	Min.	Max.		
	Min.	Max.	Min.	Max.						
Power Dissipation		100		70		120		120	mW	derate from 25° C 1.62 mW/° C
Forward DC Current		50		30		30		558 20 559 30	mA	
Peak Forward Current		1				1		1	A	1µs pulse 300pps
Peak Reverse Voltage		5				5		5	V	
Operating temperature	-55	+100	-25	+80	-55	+100	-55	+100	°C	
Storage temperature			-30	+85						
Lead Soldering temperature		260				260		260	°C	

Operating Characteristics — 558 and 559 Series WITH INTEGRAL RESISTOR (5V and 12V*)

Symbol	Characteristics	RED			GREEN			YELLOW			Units	TEST CONDITIONS		
		Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.			Red	Gr., Yel.
I_V	Luminous Intensity 558 series	.3	.8		.5	2.0		.5	2.0		mcd	V_F	5.0V	5.0V
	558-0X02	.6			.6	2.0		.6	2.0		mcd	V_F	5.0V	5.0V
	558-0103	.6									mcd	V_F	12.0V	
λ_{pk}	Wavelength		655			585			590		nm			
I_F	Forward Current		15	20		15	20		15	20	mA	V_F	5.0V	5.0V
			12	20							mA	V_F	12.0V	
BV_R	Reverse Breakdown Voltage	3			3			3			V	I_R	10µA	10µA

Maximum Ratings — 558 and 559 Series WITH INTEGRAL RESISTOR (5V and 12V*)

Characteristics			RED		YELLOW		GREEN		Units	Test Conditions
			Min.	Max.	Min.	Max.	Min.	Max.		
DC Forward Voltage	558 & 559-0202, -0302			7.5		7.5		7.5	V	Derate to 5V at 100°C
	558-0102, -0103			14						Derate to 12V at 70° C
Reverse Voltage				7.0		7.0		7.0		
Operating & Storage temperature		(5V)	-55	+100					°C	
		(12V)	-55	+70					°C	
Lead Soldering Temp.				230		230		230	°C	Maximum 7 Seconds

*12V available in RED only.

(40, 71) 38

High Efficiency
Snap-in LED Indicators
559 series

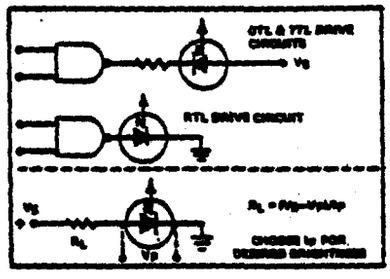
Absolute Maximum Ratings

Parameter	Red	Yellow	Green	Units
Power Dissipation (derate linearly from 25° C at 1.14mW/° C)	105	105	105	mW
Average Forward Current	35	35	35	mA
Peak Operating Forward Current (1μsec pulse width, 3% duty cycle)	1	1	1	A
Operating and Storage Temperature Range	-50° C to +100° C			
Lead Solder Temperature (1/16 inch from case)	230° C for 7 seconds			

Operating Characteristics at T_A= 25° C

Symbol	Parameter	Description	Min.	Typ.	Max.	Units	Test Conditions
V_F	Forward Voltage	Red & Yellow Green		2.2 2.2	3.0 3.0	V V	I_F= 10mA I_F= 20mA
BV_R	Reverse Voltage	All	5.0			V	I_R= 100μA
λpk	Peak Wavelength	Red Yellow Green		635 585 565		nm nm nm	Measurement at Peak Measurement at Peak Measurement at Peak
I_V	Luminous Intensity	Red Yellow Green	3.0 2.2 2.2	4.0 3.0 3.0		mcd mcd mcd	I_F= 10mA I_F= 10mA I_F= 20mA
t_S	Rise and Fall Time	Red & Yellow Green		90 200		ns ns	
C	Capacitance	All		45		pF	V_F= 0, f=1MHz

Typical Drive Circuits



Typical Resistor Values

DC Voltage	Current @ 10 mA R_L	Current @ 20 mA R_L
3.6	150 Ω	75 Ω
5	330 Ω	150 Ω
6	390 Ω	180 Ω
10	820 Ω	390 Ω
14	1200 Ω	620 Ω
28	2700 Ω	1300 Ω

70, 71

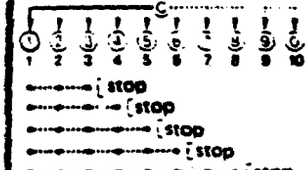
- Rating resistive circuit : 12 V 0,5 A – 24 V 0,3 A
- Minimum contact rating : 1 mA – 10 mV (See page 7)
- Contact resistance (nominal) : < 20 mΩ
- Dielectric strength
  between contacts : 1 kV eff. 50 Hz
  between contacts and body : 1 kV eff. 50 Hz
- Capacitance
  between contacts and shell : 2 pF
  between contacts : 1 pF
- Insulation resistance at ambient under 500 V = : 10 000 MΩ
- Non shorting contacts
- Life : 100 000 contacts

- Switch body : cadmiated bichromated
- Contacts : 2 μ gold on 2 μ nickel (multicon
- Termination spills : 2 μ gold on 2 μ nickel
- Insulator standard version : glass filled poly
  tropical version : P.B.T.P.
- Solder time : 5 seconds at 250°C
- Vibration : normal operation 10 – 500
  1,5 mm amplitude
- Operating temperature range : – 40
- Damp heat : 10 days to spec NFC 20
- Saline atmosphere : 96 hours to spec NFC 2

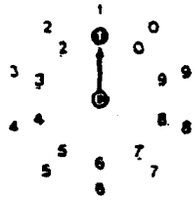
conformity to specifications and list

♦ Specification NFC 93413
✚ Included in NATO list.

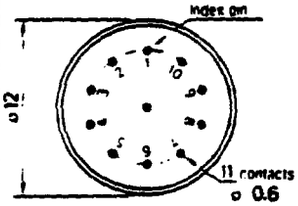
Electrical Schematic
For all models
Non - shorting contacts.



For shorting contacts consult SECME



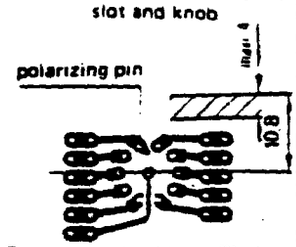
REAR VIEW



REAR VIEW
OF SWITCH BODY

Pad layout
track side of PCB

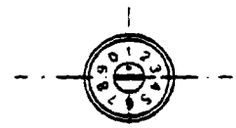
For "Leim's" with screwdriver slot and knob

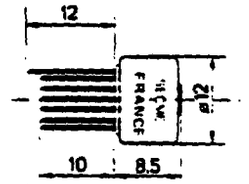


For thumbwheel type "Leim"

single pole positions

	PART NUMBER		Weight : 3,5 g Packaged in boxes of 10
	STANDARD	TROPICAL	SPECIFICATIONS
3	26 11003 23	27 11003 23	♦HK 30 CCQ
4	26 11004 23	27 11004 23	♦HK 30 CCQ
5	26 11005 23	27 11005 23	♦HK 30 CCQ
6	26 11006 23	27 11006 23	✚ ♦HK 30 CCQ
8	26 11008 23	27 11008 23	✚ ♦HK 30 CCQ
10 Without stop	26 11000 23	27 11000 23	✚ ♦HK 30 CCQ



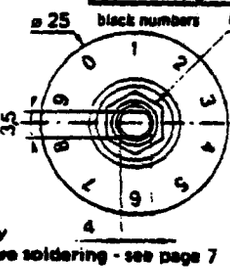


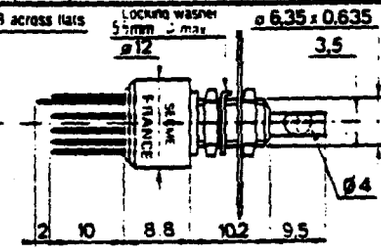
TROPICAL VERSION : designed for wave soldering use a self adhesive removable pad.- see page 7

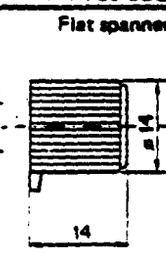
single pole positions

	PART NUMBER		Weight : 14,5 g Packaged in boxes of 10
	STANDARD	SEALED TO 1 BAR	SPECIFICATIONS
3	26 31003 23	27 31003 23	✚ ♦HK 30 CCQ
4	26 31004 23	27 31004 23	♦HK 30 CCQ
5	26 31005 23	27 31005 23	✚ ♦HK 30 CCQ
6	26 31006 23	27 31006 23	♦HK 30 CCQ
8	26 31008 23	27 31008 23	♦HK 30 CCQ
10 Without stop	26 31000 23	27 31000 23	♦HK 30 CCQ

Delivered with knurled knob and aluminium disc.







Flat spanner : see page 71

SEALED : Version specially designed for wave soldering - see page 7

Knurled knob only : 20 00111 01

mounting dimens

FOR TO : consult SEC

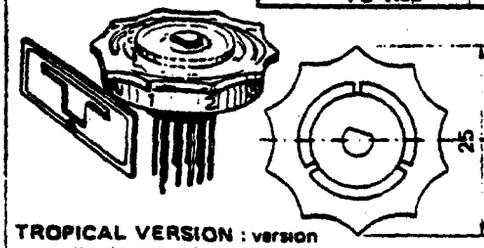
single pole positions

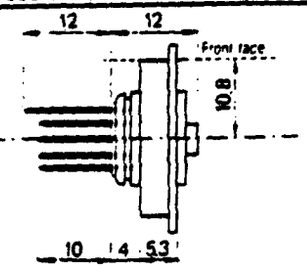
	PART NUMBER		Weight : 5 g Packaged in boxes of 10
	STANDARD	TROPICAL	
3	26 41003 23	27 41003 23	
4	26 41004 23	27 41004 23	
5	26 41005 23	27 41005 23	
6	26 41006 23	27 41006 23	
8	26 41008 23	27 41008 23	
10 Without stop	26 41000 23	27 41000 23	

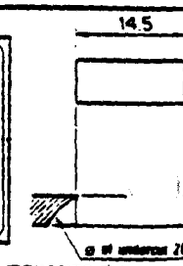
Delivered with aluminium strip havi black figures on a self-adhesive aluminium mask.

On request : aluminium disc with black letters 26 40998 81

For vertical impression : consult us







TROPICAL VERSION : version specially designed for wave soldering - see page 7

mounting dimensions

BELDEN

.050 Pitch Rainbow Cable

Computer and Instrumentation
28 Gage
Stranded Conductors (7x36)

Product Description
Tinned copper (.050 conductor spacing). PVC preinsulated-laminated to a clear PVC film. Color code: Brown, Red, Orange, Yellow, Green, Blue, Violet, Gray, White, Black.

For additional gage size and centering options or C.S.A. approval, contact your nearest Belden Distributor or Belden Sales Representative.

Will terminate to any standard IDC connector.

Standard Specifications

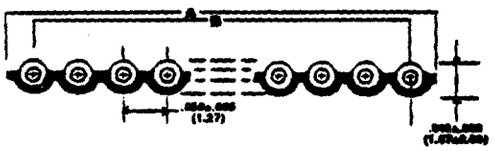
Voltage Rating	300V rms
Dielectric Withstand Voltage	2000V rms
Propagation Delay	GSG 1.4 ns/ft. (4.6 ns/M) nom.
Insulation Resistance	10^3MΩ (10 ft. sample)
Characteristic Impedance	150Ω GS, 105Ω GSG
Nominal Capacitance @ 1 MHz	10 pF/ft. (33 pF/m) GS 15 pF/ft. (49 pF/m) GSG
Inductance @ 1.MHz	0.29 µH/ft. (.95 µH/m) GS 0.20 µH/ft. (.66 µH/m) GSG
Standard Put-Up Length	100 ft. (30.4 m)
Cable Configuration Tested	GS=Ground–Signal GSG=Ground–Signal–Ground
Temperature Rating	–70°C to 150°C

Description U.L. Style	Trade No.	No. of Cond.	Nominal Width A inch	A mm	B inch	B mm
	9R28010†	10	.50	12.70	.45	11.4
	[illegible]	[illegible]	.70	17.78	.65	16.5
	9R28016†	16	.80	20.32	.75	19.1
	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	9R28024†	24	1.20	30.48	1.15	29.2
	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	9R28028†	26	1.30	33.02	1.25	31.8
	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	9R28036†	36	1.80	45.70	1.75	44.5
	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	9R28040†	40	2.00	50.80	1.95	49.5
	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	9R28060†	60	3.00	76.20	2.95	74.9
	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

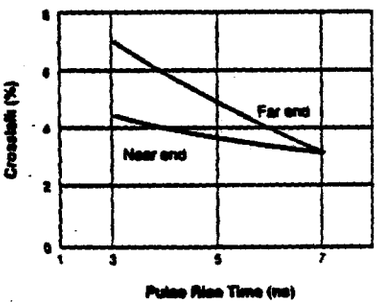
[illegible] 2884
300V 105°C

† Passes VW-1 Vertical Wire Flame Test.

Dimensions inch (mm)

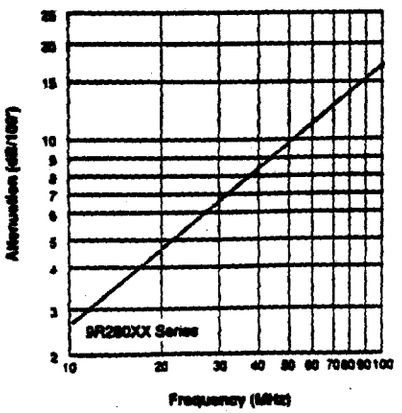


Unbalanced Crosstalk

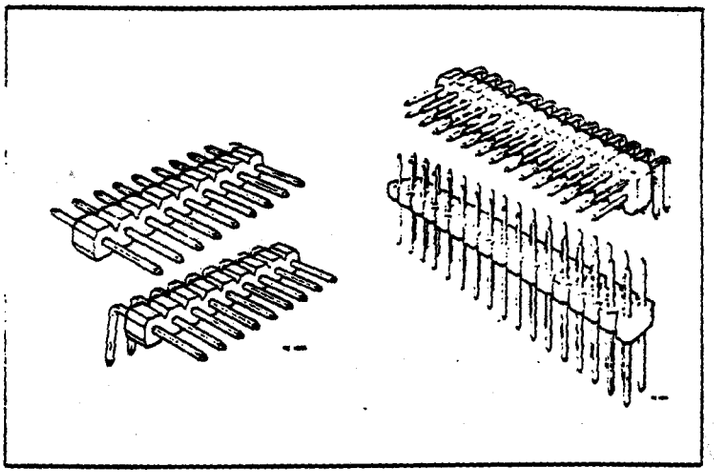


*GSG

Attenuation*

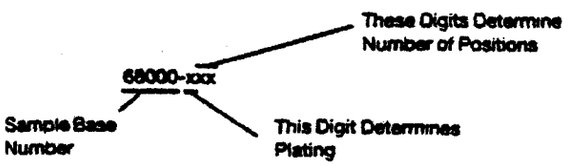


- **Can be manually broken to the desired length.**
- **Drawn 0,64 mm (0.025 in.) square wire presents 4 quality surfaces suitable for wire wrapping.**
- **Standoffs allow cleaning to eliminate soldering contaminates.**
- **Optional retention tail provides 0.5 lb minimum retention prior to soldering on 0.062 in. thick pc boards.**



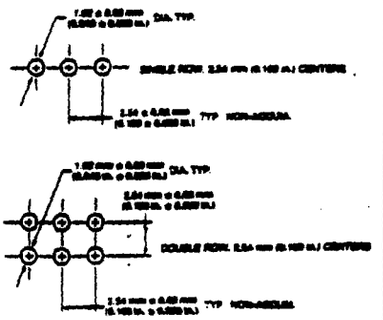
Spacing		2,54 mm (0.100 in.) and 2,54 × 2,54 mm (0.100 × 0.100 in.)
Polarization		By omitting a pin
Body	**Material**	Glass-filled nylon (UL 94 V-0). Color black
	Insulation Resistance	5000 MΩ min.
	Temperature Range	−65° C to +105° C
	Withstanding Voltage	1500 V rms (sea level)
Pin	**Material**	Phosphor bronze
	Finish	Gold, tin-lead or GXT™ (see "Ordering Data")
	Current Rating	5A max.
	Retention Force to Body	8.9 N (2 lb.) in either direction

Note on Part Numbers
All parts in this section have an eight-digit part number: the first five digits are the base number; the last three digits constitute the dash number. In some of the following tables; dash numbers will be provided. In others, dash numbers will be indicated by -xxx; in this case, you must provide the dash number based on your particular plating and position requirements.



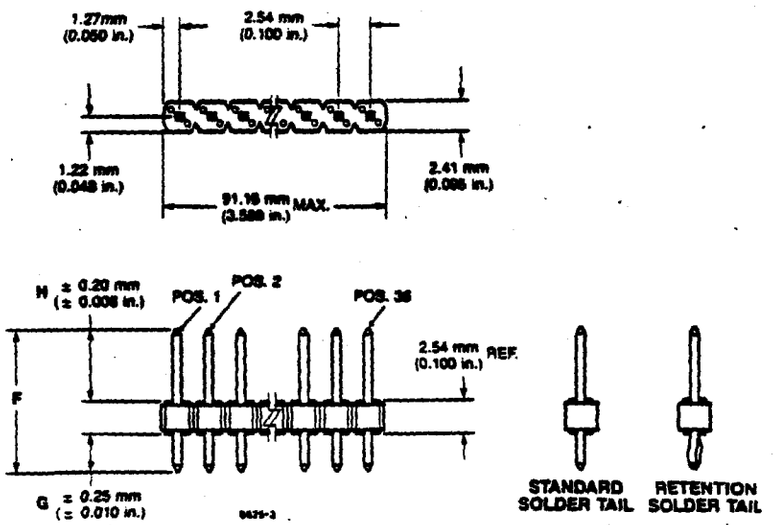
The number codes for available plating options are provided at the end of each part number list. The number of positions available are listed in their own column.

Recommended P.C.B. Hole Patterns



79

Straight Single Row
BergStik® II Headers



(Part number series preceded by * are not listed).

H Length Above Plastic		G Length Below Plastic		F Overall Length		Part Numbers Bold lettering indicates standard product	Number of	Plating*
mm	in.	mm	in.	mm	in.			
5,08	0.200	2,41	0.095	10,03	0.395	68705-xxx	1-36	1,2,3,4,5 or 6
5,84	0.230	2,41	0.095	10,80	0.425	68000-xxx	1-36	1,2,3,4,5 or 6
5,84	0.230	3,05	0.120	11,43	0.450	68001-xxx	1-36	1,2,3,4,5 or 6
5,84	0.230	3,05	0.120	11,43	0.450	68180-xxx**	1-36	1,2,3,4,5 or 6
5,84	0.230	3,81	0.150	12,19	0.480	68002-xxx	1-36	1,2,3,4,5 or 6
5,84	0.230	4,65	0.183	13,03	0.513	68031-xxx	1-36	1,2,3,4,5 or 6
5,84	0.230	7,33	0.285	15,62	0.615	68717-xxx	1-36	1,2,3,4,5 or 6
5,84	0.230	9,86	0.388	18,24	0.718	68479-xxx	1-36	1,2,3,4,5 or 6
5,84	0.230	10,16	0.400	18,54	0.730	68418-xxx	1-36	1,2,3,4,5 or 6
5,84	0.230	12,27	0.483	20,65	0.813	68416-xxx	1-36	1,2,3,4,5 or 6
5,84	0.230	15,10	0.595	23,50	0.925	68465-xxx	1-36	1,2,3,4,5 or 6
5,84	0.230	17,65	0.695	26,04	1.025	68490-xxx	1-36	1,2,3,4,5 or 6
6,86	0.270	2,54	0.100	11,94	0.470	68771-xxx	1-36	1,2,3,4,5 or 6
7,11	0.280	17,02	0.670	26,67	1.050	68654-xxx	1-36	1,2,3,4,5 or 6
8,08	0.318	3,05	0.120	13,67	0.538	*78511-xxx	1-36	1,2,3,4,5 or 6
8,08	0.318	3,05	0.120	13,67	0.538	*78229-xxx**	1-36	1,2,3,4,5 or 6
8,08	0.318	3,81	0.150	14,43	0.568	68024-xxx	1-36	1,2,3,4,5 or 6
8,08	0.318	5,72	0.225	16,33	0.643	68633-xxx	1-36	1,2,3,4,5 or 6
8,08	0.318	7,37	0.290	17,98	0.708	68655-xxx	1-36	1,2,3,4,5 or 6
8,08	0.318	10,59	0.417	21,21	0.835	68483-xxx	1-36	1,2,3,4,5 or 6
8,59	0.338	1,57	0.062	12,70	0.500	68795-xxx	1-36	1,2,3,4,5 or 6
8,64	0.340	11,18	0.440	22,35	0.880	68731-xxx	1-36	1,2,3,4,5 or 6
9,65	0.380	3,05	0.120	15,24	0.600	68604-xxx	1-36	1,2,3,4,5 or 6
10,16	0.400	2,92	0.115	15,62	0.615	68415-xxx	1-36	1,2,3,4,5 or 6
10,16	0.400	5,16	0.203	17,86	0.703	68631-xxx	1-36	1,2,3,4,5 or 6
10,16	0.400	10,49	0.413	23,19	0.913	68472-xxx	1-36	1,2,3,4,5 or 6
10,16	0.400	15,57	0.613	28,27	1.113	68656-xxx	1-36	1,2,3,4,5 or 6
11,63	0.458	11,63	0.458	25,81	1.016	68686-xxx	1-36	1,2,3,4,5 or 6
12,70	0.500	3,05	0.120	18,29	0.720	69152-xxx	1-36	1,2,3,4,5 or 6
13,54	0.533	13,58	0.533	29,62	1.166	68755-xxx	1-36	1,2,3,4,5 or 6
13,72	0.540	2,41	0.095	18,67	0.735	68653-xxx	1-36	1,2,3,4,5 or 6
14,99	0.590	3,30	0.130	20,83	0.820	68658-xxx	1-36	1,2,3,4,5 or 6
15,24	0.600	5,41	0.213	15,24	0.600	68689-xxx	1-36	1,2,3,4,5 or 6
15,49	0.610	2,92	0.115	20,96	0.825	68456-xxx	1-36	1,2,3,4,5 or 6
17,70	0.697	3,25	0.128	23,50	0.925	68466-xxx	1-36	1,2,3,4,5 or 6
17,78	0.700	17,02	0.670	37,34	1.470	68652-xxx	1-36	1,2,3,4,5 or 6

*See plating designations on page 113.
**Retention solder tail part number.

Draft Specification

Postamp / Comparator (Discriminator)

David Christian & Merle Haldeman

30-MAR-89

Version 2.3

Table of Contents

1.0 General Information

1.1 Overview

The purpose of the **Postamp / Comparator** (P / C) module is to compare the voltage amplitude of the pulses, at the input of this module, to a reference voltage (the threshold voltage) and to latch a logic level output for each pulse greater than that programmable reference voltage. The latching of the logic levels is synchronized with a 53 MHz clock provided by a **sequencer** module. Differential input signals are received from Silicon Strip Detector **(SSD) amplifiers** through four 64-conductor ribbon cable connectors mounted on the front portion of the P/C module. ECL level output signals are sent, via a custom auxiliary backplane, to a **delay / encoder** module located in an adjacent slot in the FASTBUS crate. Eight fast analog sums and eight analog encoded digital sums, intended for use by the prompt trigger logic, are constructed and output through FASTBUS auxiliary cards.These signals also provide a means of debugging a PC module with an oscilloscope without the need of an extender card.

1.11 Standard Bus System Used

The Postamp/Comparator module is a FASTBUS slave module designed to be used together with a delay / encoder module in a FASTBUS crate equipped with an SSD Readout auxiliary backplane.

1.12 Number of Channels

The Postamp/Comparator module is a 128 channel module.

1.2 Application

This module was designed for use in the fast Silicon Strip Detector (SSD) readout system developed at FNAL for use by E-771 and E-789. It is directly usable only in the context of this readout system.

1.3 Packaging

This board is a standard single width FASTBUS module (see ANSI / IEEE Std 960-1986).

1.31 Physical Size

The maximum board dimensions are 14.437 inches high by 15.878 inches deep. The board thickness is between 0.086 inches and 0.100 inches.

1.32 Pinout

See Appendix 6.

1.33 Front Panel Displays

The following front panel display LED's are implemented:

NAME	COLOR	ENERGIZED MEANING
Mode	**Red**	module is in the **INITIALIZE (TEST)** mode.
Mode	**Green**	module is in the **RUN** mode.
Clock	**Yellow**	**CLOCK 1** (53 Mhz) is **present.**
Select	**Yellow**	Module is being **accessed via FASTBUS**
Temp.	**Red**	Module temperature is **> 50 degrees C.**

1.4 Power Requirements

The Postamp/Comparator module uses the following voltages which are distributed on the FASTBUS backplane:

+5.0 V	6.6 Amps
-5.2 V	7.9 Amps
-2.0 V	0.16 Amps

Five additional voltages are derived from these and used by the ASIC's.They are:

+3.5 V	4.0 Amps
-1.0 V	0.16 Amps
-3.5 V	4.4 Amps

1.41 Control and Monitoring Requirements

All power supply voltages distributed on the FASTBUS backplane will be monitored by a CAMAC based system. This system will be read out into EPICURE.

1.5 Cooling Requirements

The total power dissipated on the board will not exceed 75 watts.

2.0 Theory of Operation and Operating Modes

2.0 Theory of Operation and Operating Modes

2.1 Basic Operation

2.11 Inputs

128, SSD preamplifier signals are received at the Postamp/Comparator module via four, 64-conductor ribbon cables. The input signal order does not correspond to the order of the silicon strips in the detector for reasons having to do with signal density on both the detector itself and the amplifier cards connected to the detector. The input signal traces on the Postamp/Comparator module are routed to restore the monotonic strip order of the detector. These differential signals are coupled through .1μF capacitors to the inputs of the IC-01 ASIC's, and both sides terminated through 50 Ω to ground. These termination components are housed in a custom 10-lead sip each of which will terminate two channels (4 wires) of the 128 channels; thus 64 sips per module.

2.12 IC-01: SSD Two Channel Sum, Discriminator and Latch

IC-01 is a bipolar ASIC made using a Tektronix Quickchip 2K-130 linear array. It contains two linear summing amplifiers, four time-over-threshold comparators (with variable threshold setting capability and hysteresis), and four latches (see Appendicies). Each IC-01 (except for the units with the lowest and highest numbered channels on the board) receives signals from three consecutive silicon strips. The first and third signals are also input to neighboring IC-01 chips. Two linear sums are made; the first by adding the first and second inputs, and the second by adding the second and third inputs. The output of each of the two linear sums is input to a comparator. The output of the second sum is also sent to IC-04. The second and third inputs are connected to the remaining two comparators. The result is two pairs of comparators, one pair discriminating individual strip signals (inputs two and three), and the other pair discriminating the two sum outputs. Each discriminator has a separate threshold setting input. The output of each of the four comparators is sent directly to a transparent latch (one per comparator) which is controlled by a differential, 2.5 ns wide LATCH input signal. This LATCH signal, common to all four latches, controls the function of the latches, which can be thought of as digital sample-and-holds. When the LATCH control lines are in the "transparent" state, the LATCH output follows the comparator output. When the LATCH is in the "latched" state, the output of the LATCH remains unchanged. In normal operation, an ASIC (IC-02) located within three inches of each IC-01 provides this 2.5 ns wide pulse. This LATCH pulse is synchronized to the 53 MHz CLK1 signal received from a sequencer. This pulse switches the LATCH input from "latched" mode to "transparent" mode for a time equal to the width of the pulse, and then returns it to the "latched" mode. The result is that the state of the comparator is latched and held for one RF cycle less the width of the LATCH pulse. Two more control signals, called INDIVIDUAL CHANNEL FORCE ZERO and SUMMED CHANNEL FORCE ZERO, also effect the operation of the latches. INDIVIDUAL CHANNEL FORCE ZERO is

connected to the two individual strip latches and SUMMED CHANNEL FORCE ZERO is connected to the sum latches. When FORCE ZERO is HI, the effected latches are forced to logical zero. This function is used to insure that no signal is received from an IC-01 chip while the discriminator board is in the INITIALIZE / TEST mode (see below). It also makes it possible to disable either bank of comparators while leaving the other bank operational.

2.13 IC-02 - 3-Channel Logic,Quad Analog Sum & Latch Driver

The IC-01 latched outputs are input to two other bipolar ASIC's (IC-02 and IC-04) made using Tektronix Quickchip 2K-130 linear arrays (see figure 2). The combination of these two chips accept inputs from four IC-01's (8 SSD channels), and produces eight ECL outputs which are sent, via the auxiliary backplane, to the **delay / encoder** board in the adjacent slot to the right (viewed from the front of the FASTBUS crate). The logic producing these eight ECL signals performs in the following way: any given output channel N is set HIGH if the corresponding single strip (N) latch is set, or if one of the summed-channel latches including strip N is set, but neither of the individual channel latches for the two channels contributing to the sum are set. These signals are OR'ed with test inputs which sit at logical low levels while the module is in the **RUN** mode. Test signals are sent to these test inputs in the **TEST** mode during which time the latch outputs are forced to a logical zero.

IC-04 is also used to produce a signal known as **NHIT** or **DIGITAL SUM** which consists of a current, the magnitude of which is proportional (100 uA / strip hit) to the number of hit strips in the group of eight. The NHIT inputs are constructed by both IC-02 and IC-04, forming a logical OR of a single strip signal and one of the summed channel signals which has that strip as one of it's inputs. The result is that the output of NHIT will register a ONE if a hit is shared by two adjacent strips even though two bits (for two adjacent strips) are set in the **delay / encoder**.

IC-02, similar to IC-04, produces an **ANALOG SUM** of the same eight strips used in producing NHit. The difference being that this output current is truely a linear sum of the eight individual strip signals. IC-02 also has the circuitry to accept a narrow pulse (2.5 ns wide) and fan out a differential version of this pulse to the LATCH inputs of four **IC-01 Sum, Discriminator and Latch** chips.

2.14 IC-03 - DAC / ADC

Voltage levels are provided for the threshold voltage (Vth) inputs of the **IC-01 Sum, Discriminator and Latch** chips by a CMOS ASIC produced by United Silicon Structures. This chip contains four 8 bit D/A converters and one 8 bit A/D converter. This A/D converter requires an on board 100 kHz clock which can be turned on or off via CSR0. Up to 64 DAC chips may be mounted, providing individual control of the 256 discriminators on a Postamp/Comparator module. The DAC chips are loaded under FASTBUS control and the analog outputs (Vth) are measured by the integral A/D and read back via FASTBUS. The IC-03 chips

share 8 data lines and 4 control lines. Each chip has a separate select line. The DAC outputs may be jumpered together to allow the use of as few as one DAC chip on the Postamp/Comparator module. The output range is from 0 V to the reference 2.55 V. The output impedance of these DACs varies between 10 and 30 Ω. Each chip requires power supply voltages of -1, +5 and a reference voltage of +2.55.

2.15 IC-04 - SSD 5-Channel Logic and Octal NHit

See section 2.13

2.16 Test Circuit

A presetable 8 bit counter is included on the Postamp/Comparator module to provide a means of sending test data to the **delay / encoder** module associated with the Postamp/Comparator module. This counter is controlled through FASTBUS. The counter's outputs are reset (forced to zero) when the discrimintor board is in the RUN mode. When the board is in the **INITIALIZE (TEST)** mode, the counter may be set in the count or hold mode. In the hold mode, the counter output pattern can be altered by presetting the counter via FASTBUS. In the count mode, the counter is incremented by the CLK1 signal. The connection of the test counter to IC-02 and IC-04 is illustrated in figure 1. This means that 256 unique patterns may be sent to the delay / encoder board. The test pattern may be static i.e., altered by presetting the counter via FASTBUS or may change every 18.9 ns by being incremented by CLK 1.

2.17 FASTBUS Interface

The discriminator is a FASTBUS slave. It responds to geographic addressing in control space only. It allows both single and block transfers and contains the following registers:

CSR0 This is the main control and status register. It is a selective set and reset register with the following bit assignments:

BIT 02 -- **RUN / HALT** — Writing a "1" to this bit puts the module in the **RUN** mode. A "0" must be simultaneously written to BIT 18. This bit is cleared by writing a "1" to BIT 18 (**HALT**).

BIT 06 -- **EN_IC** — Writing a "1" to this bit ENABLES the individual strip latches. Has the opposite effect as that of BIT 06.

BIT 07 -- EN_SC — Writing this bit ENABLES the summed channel latches = clears bit 07 (FORCE ZERO=OFF).

BIT 08 -- CCLK__EN — When HALTED, setting this bit enables the clock to the 8-bit test counter. The clock is disabled while the module is in the RUN mode.

BIT 09 -- CLK1__ON — This bit will READ ONE whenever CLK1 (53 MHz) is present.

BIT 10 -- **ADCCLK__EN** — Writing a "1" to this bit ENABLES the 100 kHz clock which is used by the ADC portion of the DAC / ADC units.

BIT 18 -- **HALT** — Writing a "1" to this bit places the module in the **INITIALIZE / TEST** mode (In this mode all latch outputs are FORCED to ZERO). A "0" must be simultaneously written to BIT 02.

BIT 22 -- **FZ__IC** — Writing a "1" to this bit **FORCEs** the **individual channel** latch outputs to **ZERO.** A "0" must be simultaneously written to BIT 22. This bit is cleared by writing a "1" to bit 22. This bit will READ ONE if the individual strip latches are FORCED to ZERO.

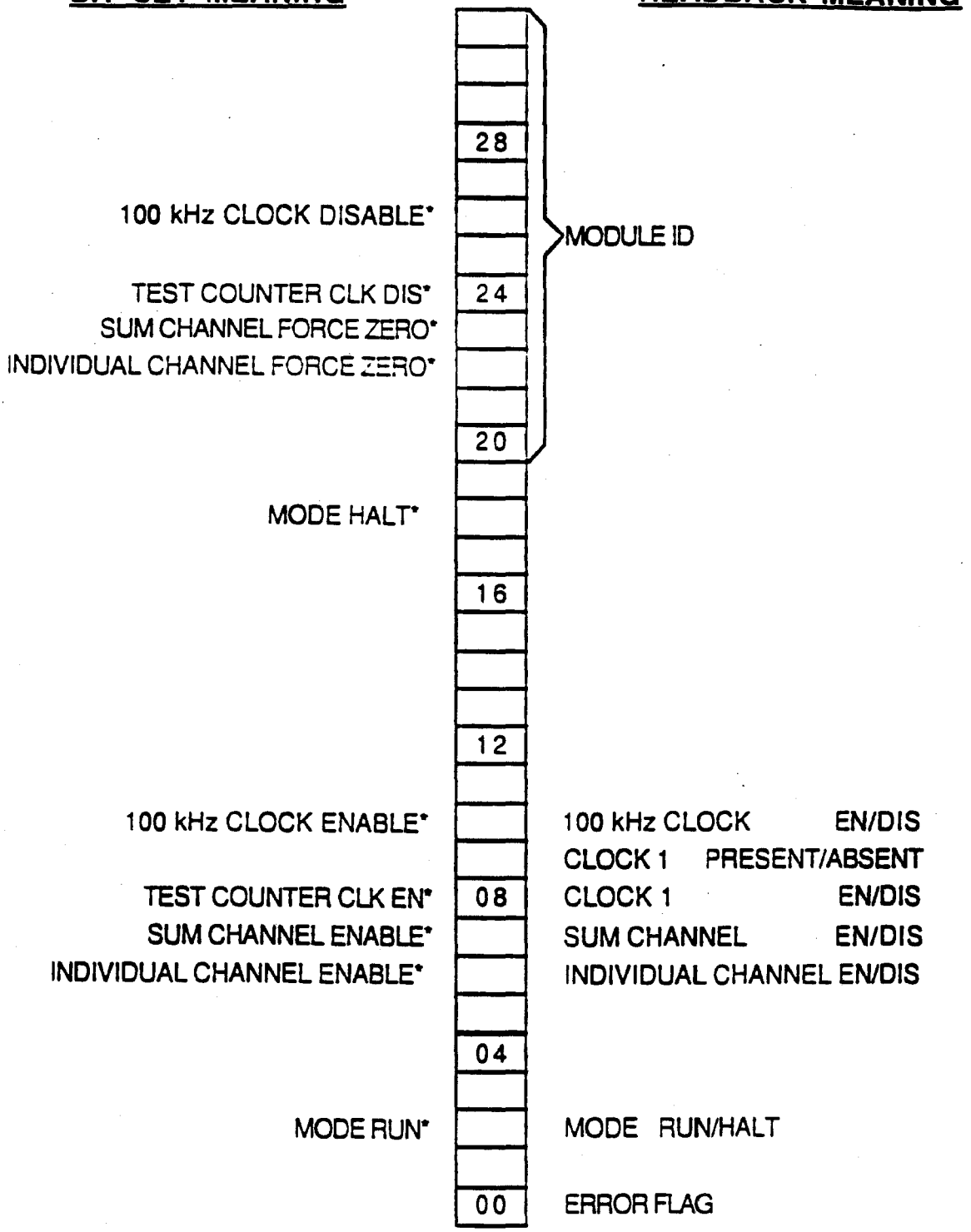
BIT 23 -- **FZ__SC** — Writing a "1" to this bit **FORCEs** the **summed Channel** latch outputs to **ZERO.** A "0" must be simultaneously written to BIT 23. This bit is cleared by writing a "1" to bit 23. This bit will READ ONE if the summed channel latches are FORCED to ZERO.

BIT 24 -- CCLK__DIS — Writing to this bit disables the clock to the 8-bit test counter.

BIT 20-31 -- ID — The device ID is read back on these bits.

BIT 26 -- ADCCLK_DIS — Disables the100 kHz clock enabled by bit 10

Preamp/Comparator (Discriminator) Module



CSR0

* The corresponding bit displaced 16 bits away must have a zero written simultaneously.

CSR1 This register is used to load and read the 8-bit test counter. The counter is disabled when the module is in the RUN mode.

BIT 00-07 -- DATA Data to be loaded to / read back from the 8-bit test counter.

Preamp/Comparator (Discriminator) Module

TEST COUNTER CONTROL AND STATUS

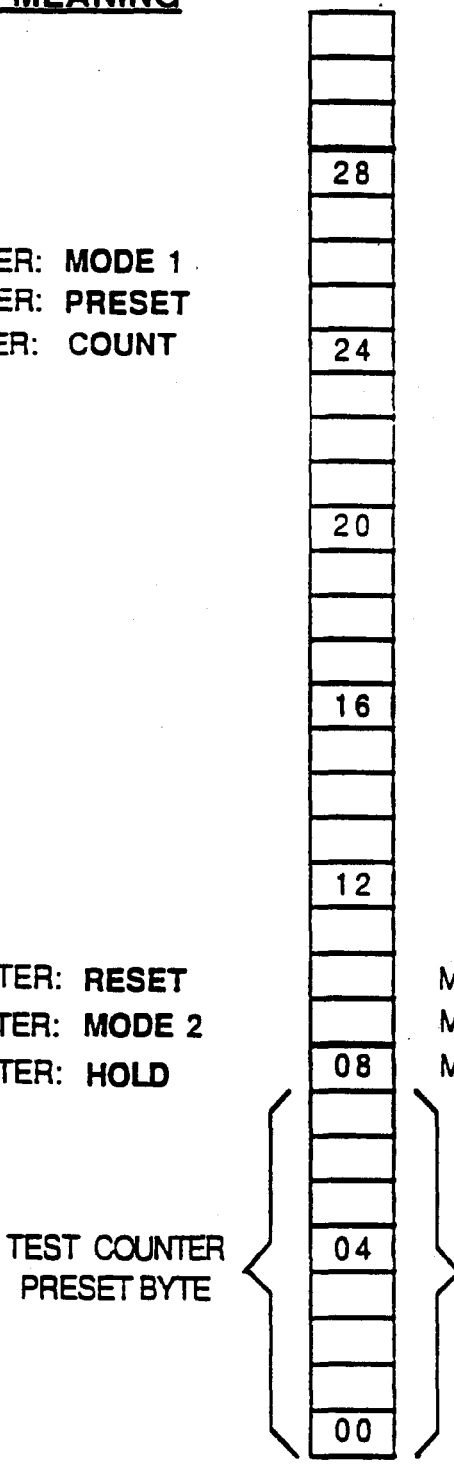
BIT SET MEANING

* TEST COUNTER: **MODE 1**
* TEST COUNTER: **PRESET**
* TEST COUNTER: **COUNT**

* TEST COUNTER: **RESET**
* TEST COUNTER: **MODE 2**
* TEST COUNTER: **HOLD**

READBACK MEANING

MODULE STATUS: : RESET/MODE 1
MODE 1 STATUS: MODE 2/PRESET
MODE 2 STATUS: HOLD/COUNT



CSR1

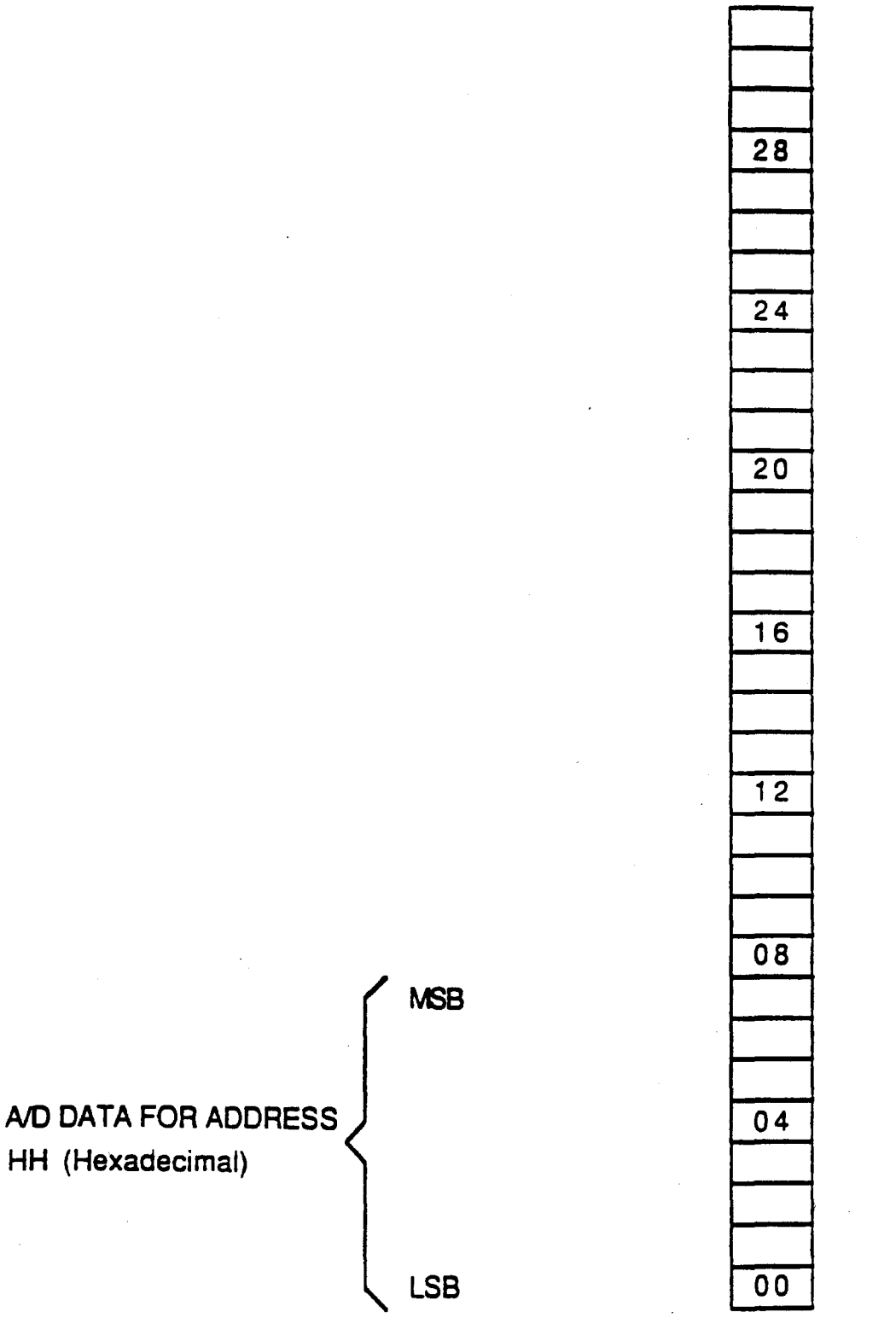
* The corresponding bit displaced 16 bits away must have a zero written simultaneously.

CSR C000__0000 - C000__00FF

These registers are used to load and read the threshold DAC's. They can be written only when the module is HALTed (in the INITIALIZE / TEST mode). The module returns an SS=6 if a write is attempted when in the RUN mode. This address range is programmable to match the number of DAC's installed on the board.

BIT 00-07 -- DATA Data to be loaded to / read back from DAC's.

Preamp/Comparator (Discriminator) Module



CSR COOO OOHH

2.2 Addressing Modes

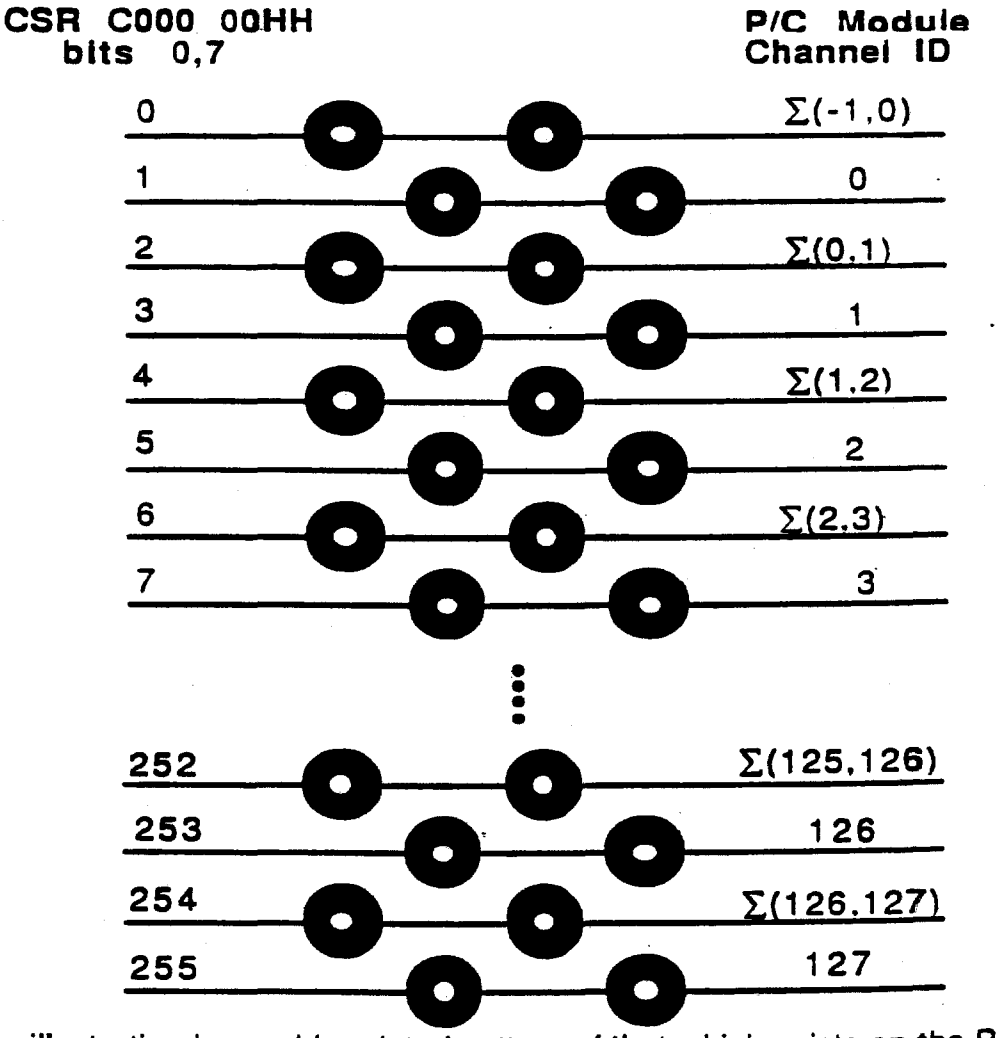
2.21 Error Responses

The module asserts a FASTBUS error (SS=6) if an attempt is made to write to a threshold DAC while the board is in the RUN mode.2.2 Addressing Modes

2.3 Hardware Jumpers and Switches

2.31 DAC / ADC Jumpers

As was stated in section 2.14, the DAC outputs can be jumpered to allow a given output to be connected to one or more of any of the 256 discriminator threshold inputs.

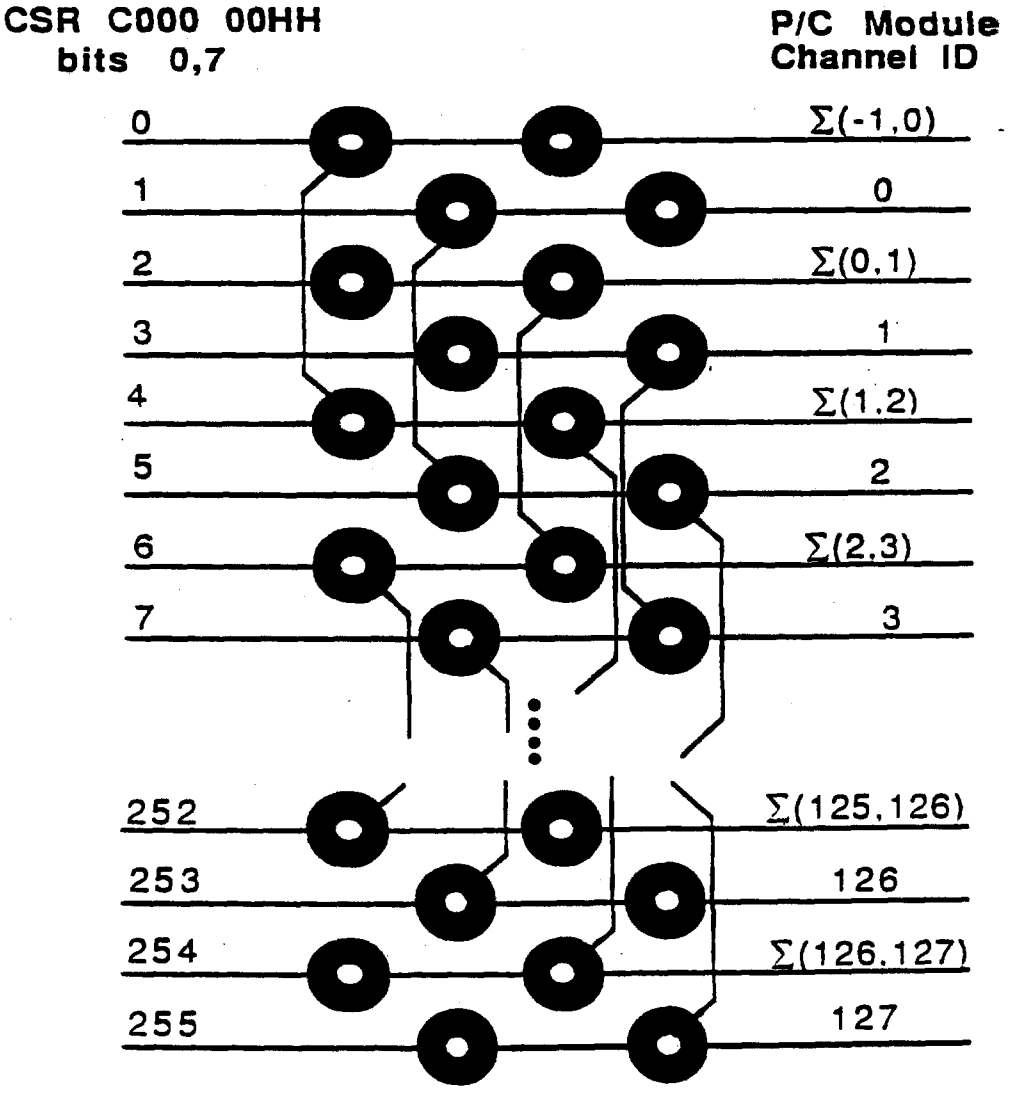


The above illustration is an abbreviated pattern of that which exists on the P/C Module. The lines on the left are connected to the DAC / ADC outputs and the lines on the right are connected to the threshold setting inputs.

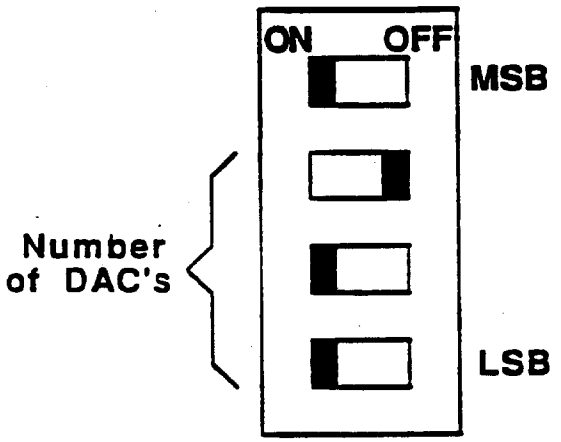
There are 256 DAC/ ADC's which are addressed via bits 0 through 7 of CSR C000 00HH. DAC address 00 (Hex), will always set the threshold voltage on the uppermost

trace which is the threshold for the discriminator connected to the output of the summed channel which takes the sum of channel -1 and channel 0 (Σ-1,0). Likewise DAC address 01 (Hex), will always set the threshold voltage on the trace immediately below the the uppermost trace which is the threshold for the discriminator connected directly to the channel 0 input. DAC address 02 (Hex), will always set the threshold voltage for the discriminator connected to the output of the summed channel which takes the sum of channel 0 and channel 1 (Σ0,1). DAC address 03 (Hex), will always set the threshold voltage for the discriminator connected directly to the channel 1 input. This patttern continues for the 256 DAC /ADC's and the 256 discriminators.

It should be obvious that if all 256 DAC 's are employed, no jumpers are required. However, in the event that only 4 of the 256 DAC's are mounted on the P/C Module, the jumpers would probably be applied as illustrated below. In this case DAC 00 (hex) would drive every other sum channel. i.e., Σ(-1,0), Σ(1,2), Σ(3,4), Σ(5,6) etc. DAC 01 (hex) would drive the even individual channels; DAC 02 (hex) would drive sum channels Σ(0,1), Σ(2,3), Σ(4,5), Σ(6,7) etc., and DAC 03 (hex) would drive the odd, individual channels.



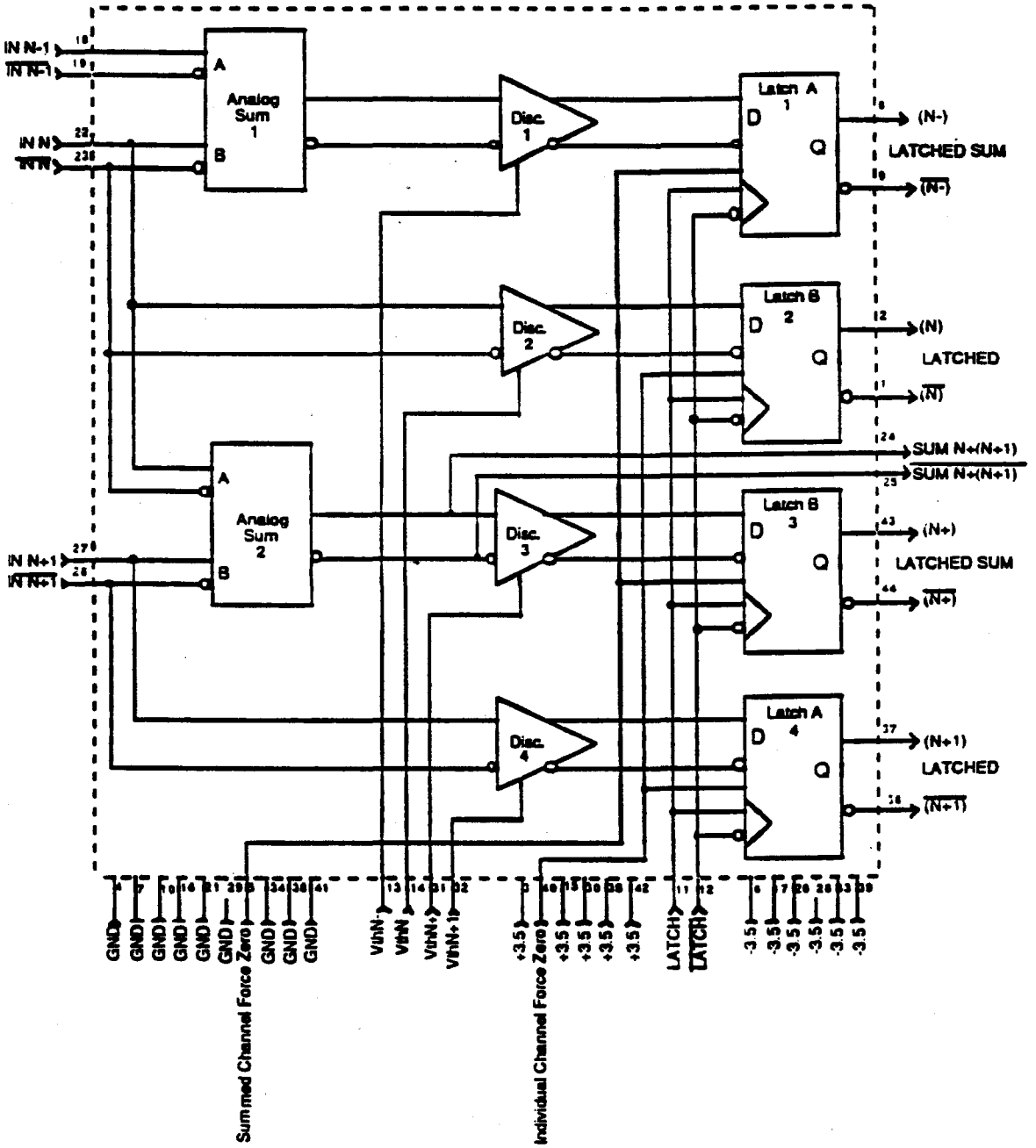
When less than 256 DAC's are utilized, the P/C module will be populated from th lowest address onward without skipping addresses. An on board,4 bit dip switch (illustrated below) utilizes 3 bits to indicate the number of DAC chips mounted. The choices are 0,4,8,16,32,64,128 or 256 which corresponds to 0,1,2,4,8,16,32 and 64 packages. It seems reasonable to assume that for four DAC's every fourth threshold lead would be connected togather and so on. However, any and all possibilities exist for connecting DAC outputs to threshold inputs and the methode used may be dictated by the detector geometry.



APPENDICIES

1. IC-01
2. IC-02
3. IC-03
4. IC-04

IC-01: SSD TWO CHANNEL SUM, DISCRIMINATOR AND LATCH
Cavity Down



Part Number IC-01

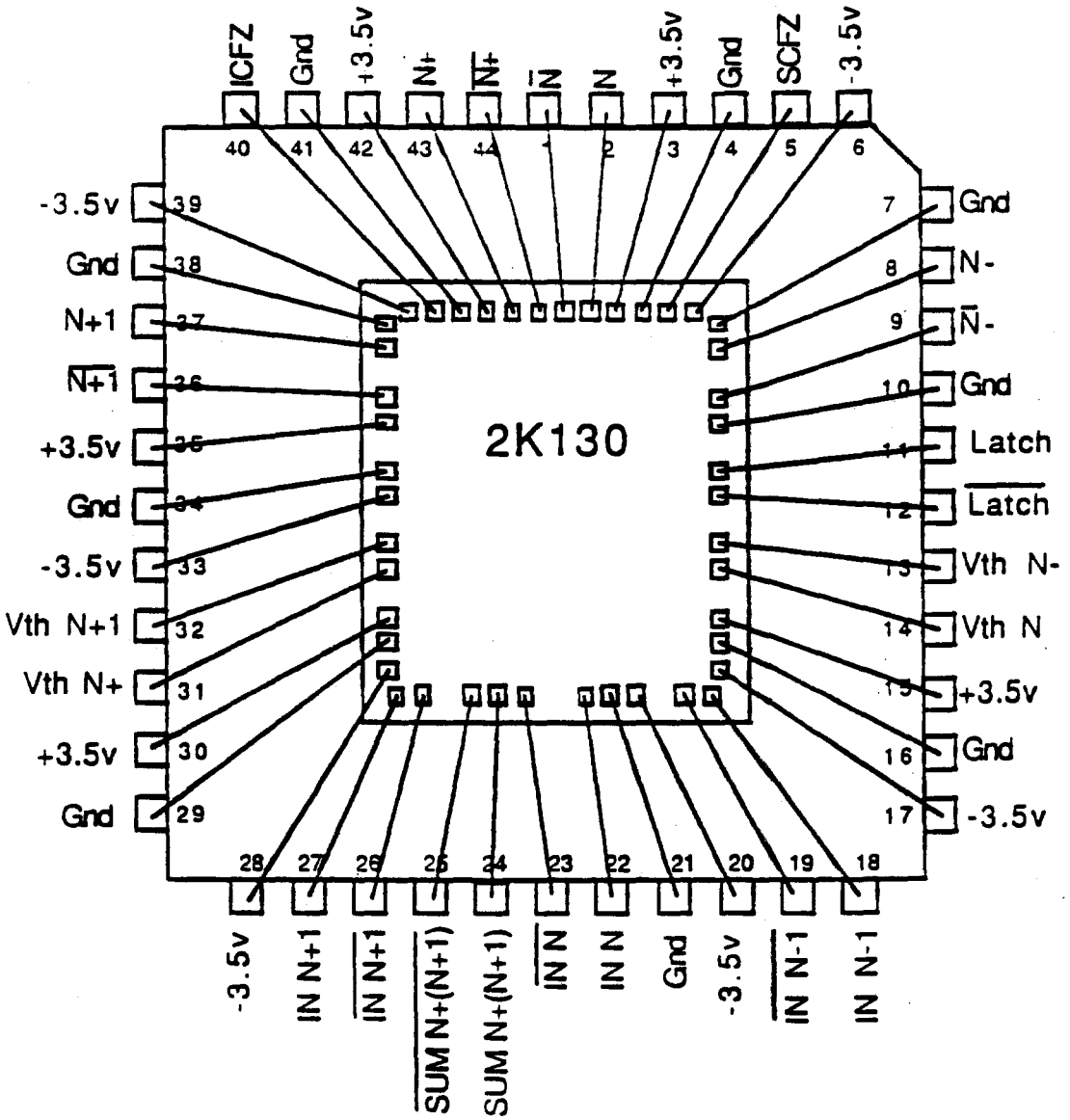
NOTES:

1. All input and output signals are full differential except power, threshold (Vth#) and Force Zero.
2. The discriminator compares the input amplitude with the threshold setting. If the input exceeds the setting, an output exists for the time over threshold.
3. The "LATCH" signal must be at least 1 Nanosecond wide.
4. Power dissipation is typically 406 mW : 504 mW max.
5. Power supply currents: +3.5 V @typically 46 mA : 56 mA max.
   -3.5 V @typically 68 mA : 88 mA max.

Created 1-24-89
Revised 3-30-90
Merle Haldeman / Scott Holm

IC-01: SSD TWO CHANNEL SUM, DISCRIMINATOR and LATCH
Cavity Down
Thermally Enhanced, 44 Lead Plastic Chip Carrier

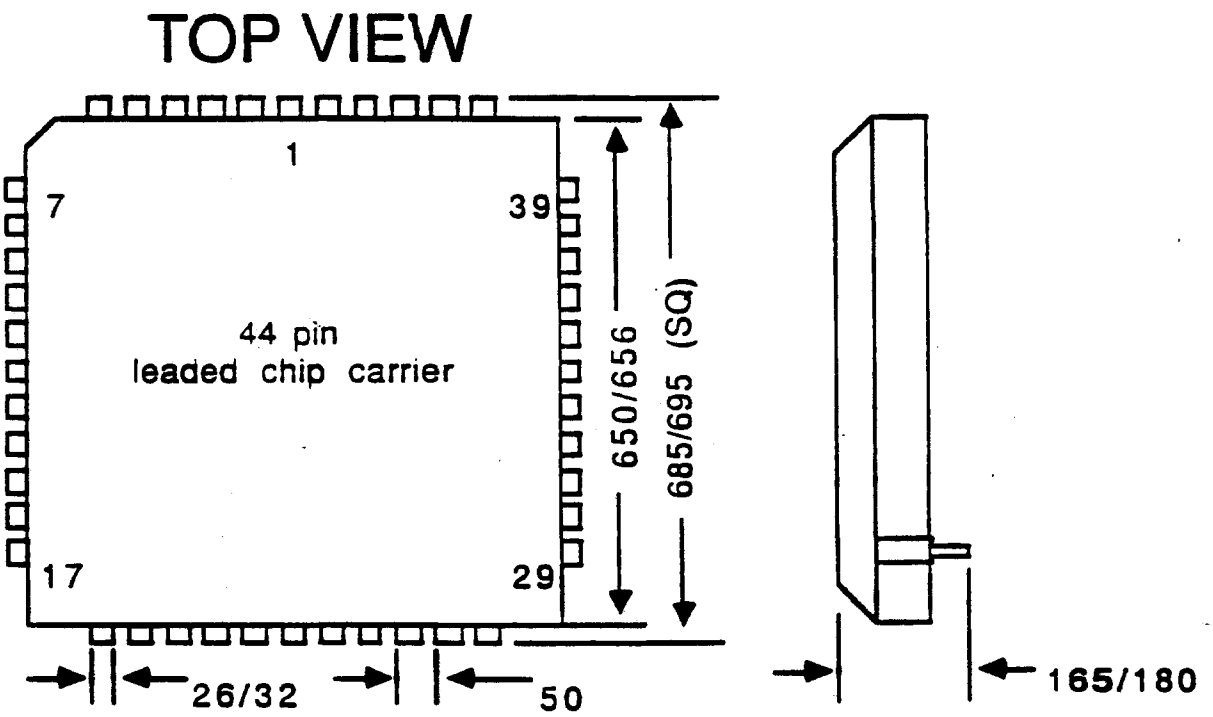
Part Number IC-01



BOTTOM VIEW

Created 8-28-89
Revised
Merle Haldeman / Scott Holm

IC-01: SSD TWO CHANNEL SUM, DISCRIMINATOR and LATCH
Cavity Down



IC-01 Package Dimension Diagram Notes.

1. All dimensions are in mils. (Min/Max)

Created 1-24-89
Revised 6-29-89
Merle Haldeman / Scott Holm

IC-01
SSD TWO CHANNEL, DUAL SUM, QUAD DISCRIMINATOR AND QUAD LATCH

Overview

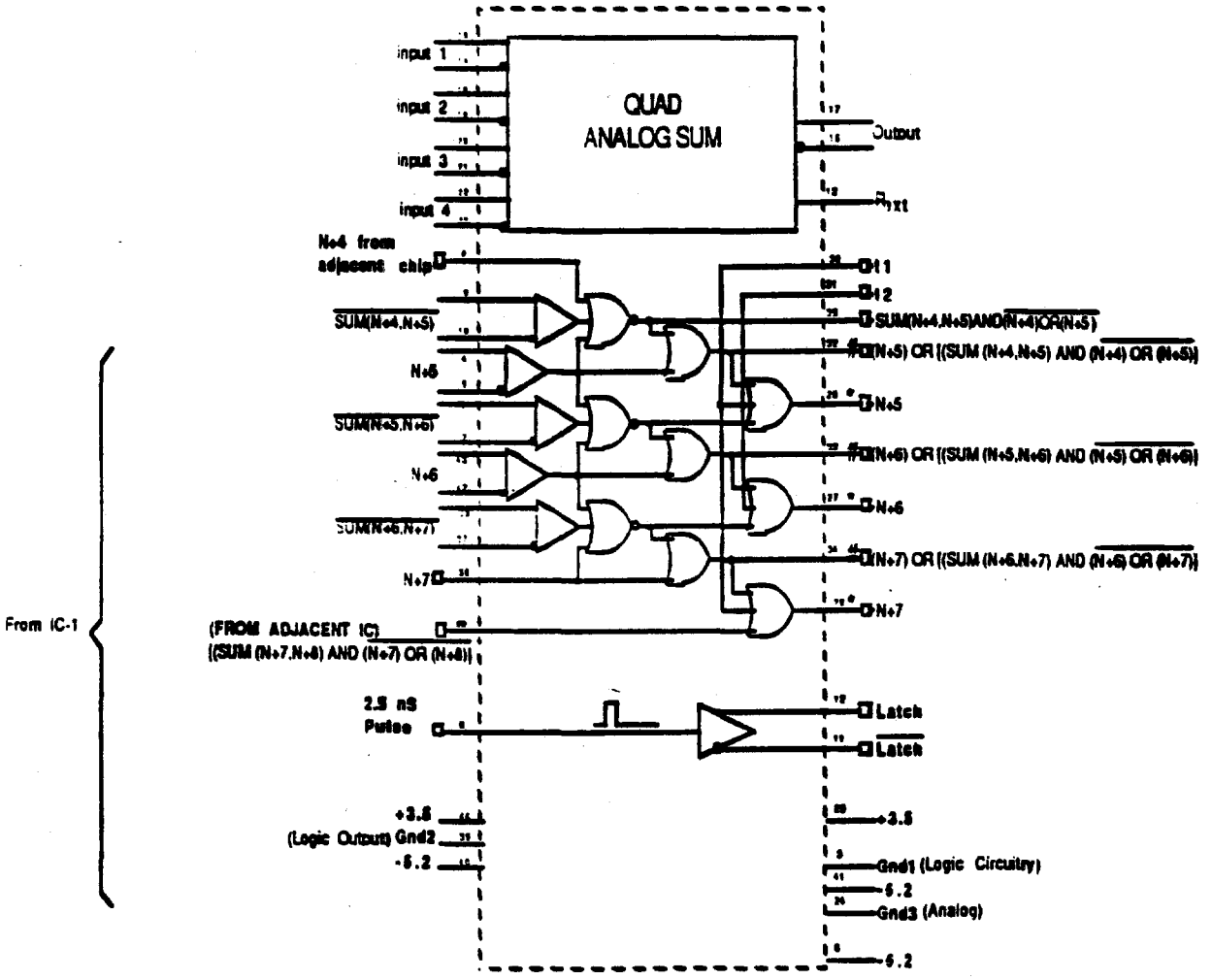
This IC is being designed for two experiments here at FERMILAB; E-771 and E-789. In E-771, a Silicon Strip Detector (SSD) produces signals which are amplified by preamplifiers and then transmitted over balanced,100 ohm impedance, transmission lines to the inputs of several of the IC-01 devices for further signal processing. These ICs will be mounted on a FASTBUS board approximately as illustrated in figure 1. These boards are then plugged into a FASTBUS Crates each of which is capable of housing 26 single width FASTBUS modules. This experiment has **24 planes with 688 strips per plane** for a **total of 16,512 strips** to be read out.

Circuit Function

As can be seen in figure 2, IC-01 consists of two sum circuits, four discriminators and four latches. The four discriminators are identical. Each of the sum circuits has a differential voltage gain of approximately one. The purpose of this summing amplifier is to provide a signal voltage, when a charged particle passes between two strips, that has essentially the same amplitude as a that produced by the same particle passing solely through one strip. The output of these sum amplifiers drive the two discriminators which look for the shared signal condition.

The discrimiantors have a built in hysteresis which is equivalent to approximately 10 millivolts referred to the input. In this area of operation , the discriminator has sufficient positive feedback to provide infinite voltage gain. This gain only exists for an input variation of about ten millivolts. This feature is provided by current source "QCS3" and current switch transistors "Q300" and "Q301". There is a variable threshold input terminal which allows the input discriminate between input voltages from 5 to 50 millivolts for a dynamic range of 10 to 1. The threshold is controlled by the application of a DC voltage to terminal "VTHR".

IC-02: SSD 3-Channel Logic, Quad Analog Sum and Latch Driver



Notes:

- * Lines to Auxiliary Backplane
- # Lines to NHit
- ECL Levels

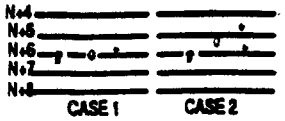
Power dissipation: 473 mW typical: 597 mW max.

Power supply currents: +3.5 @ 21mA typ.; 27 mA max.
-5.2 @ 62 mA typ.; 78 mA max.

INPUTS					OUTPUTS	
N+5	SUM(N+5,N+6)	N+6	SUM(N+6,N+7)	N+7	N+6	N+6 OR [****]
X	X	1	X	X	1	1
0	1	0	X	X	1	1
X	X	0	1	0	1	

All other input conditions produce zero outputs

X = Don't Care



o Location of particle

Signals to NHit

* Signals to Auxiliary Backplane

All 44 PADS USED

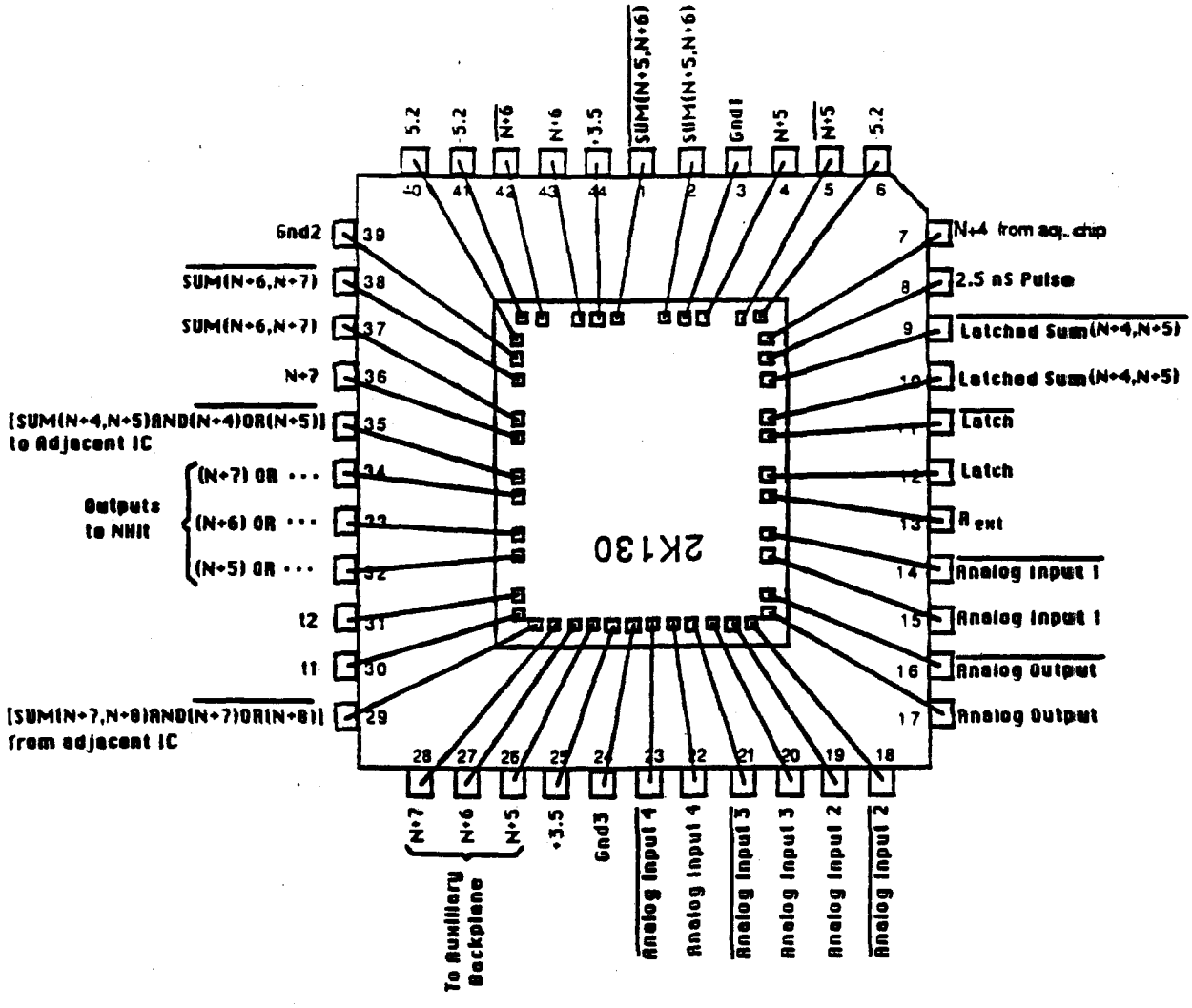
Created 4-13-88
Revised 8-25-89
Mario Holzbauer / Bruce Mandel

IC-02: SSD 3-CHANNEL LOGIC, QUAD ANALOG SUM and LATCH DRIVER

Cavity Down

Thermally Enhanced, 44 Lead Plastic Chip Carrier

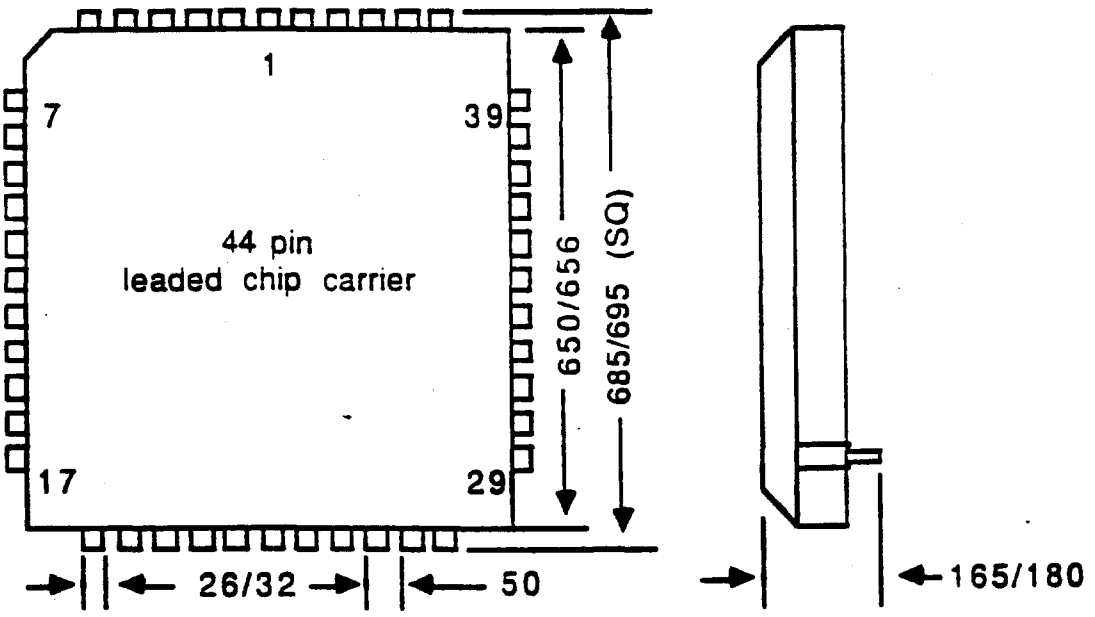
Part Number IC-02



BOTTOM VIEW

Created 8-28-89
Revised
Merle Haldeman / Bruce Merkel

IC-02: SSD 3- CHANNEL LOGIC , QUAD ANALOG SUM and LATCH DRIVER
Cavity Down



TOP VIEW

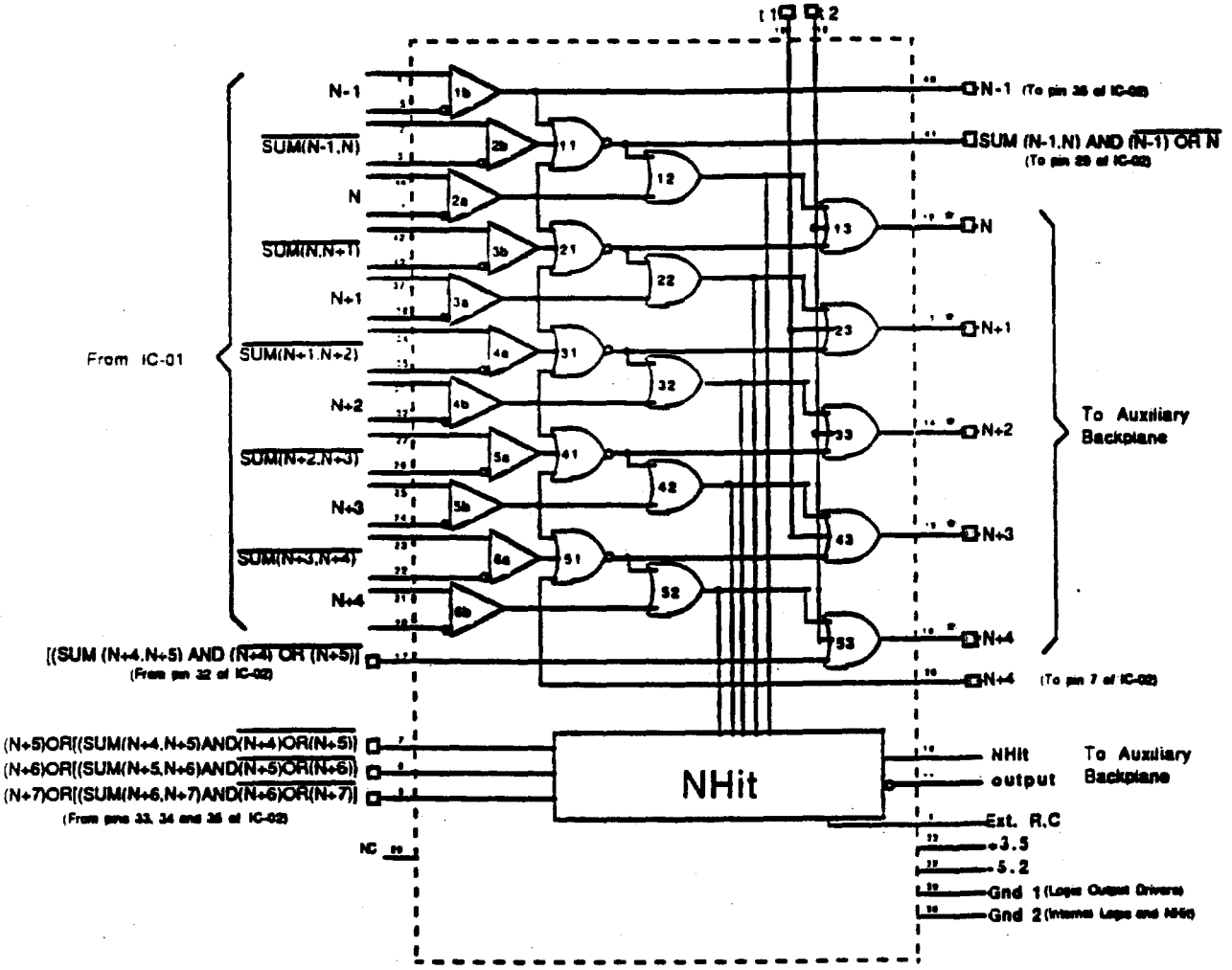
Part Number IC-02

IC-02 Package Dimension Diagram Notes.

1. All dimensions are in mils. (Min/Max)

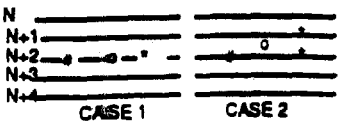
Created 5-18-89
Revised 6-30-89
Merle Haldeman /Bruce Merkel

IC-04: SSD 5-Channel Logic and Octal NHit



INPUTS					OUTPUTS	
N	SUM(N,N+1)	N+1	SUM(N+1,N+2)	N+2	N+1	N+1 OR [*****]
X	X	1	X	X	1	1
0	1	0	X	X	1	1
X	X	0	1	0	1	

All other input conditions produce zero outputs
X = Don't Care



o Location of particle in SSD
Signals to NHit
* Signals to Auxiliary Backplane
ECL Levels

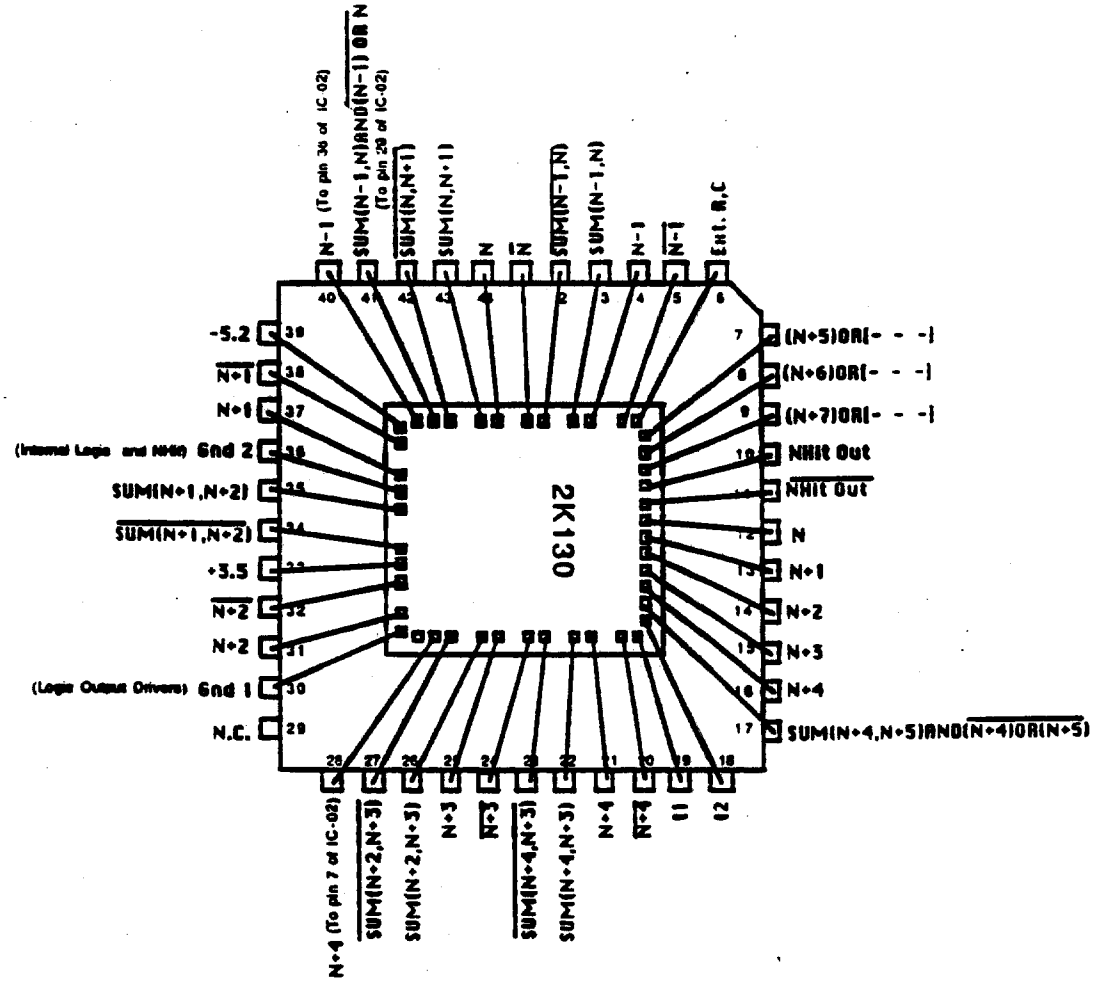
Power dissipation 350 mW Outputs unloaded.
Power dissipation 500 mW All outputs loaded with 100 ohms to -2.0 Volts.

Created 4-23-90
Revised 6-3-90

IC-04: SSD 5-CHANNEL LOGIC & OCTAL NHit Cavity Down

Thermally Enhanced, 44 Lead Plastic Chip Carrier

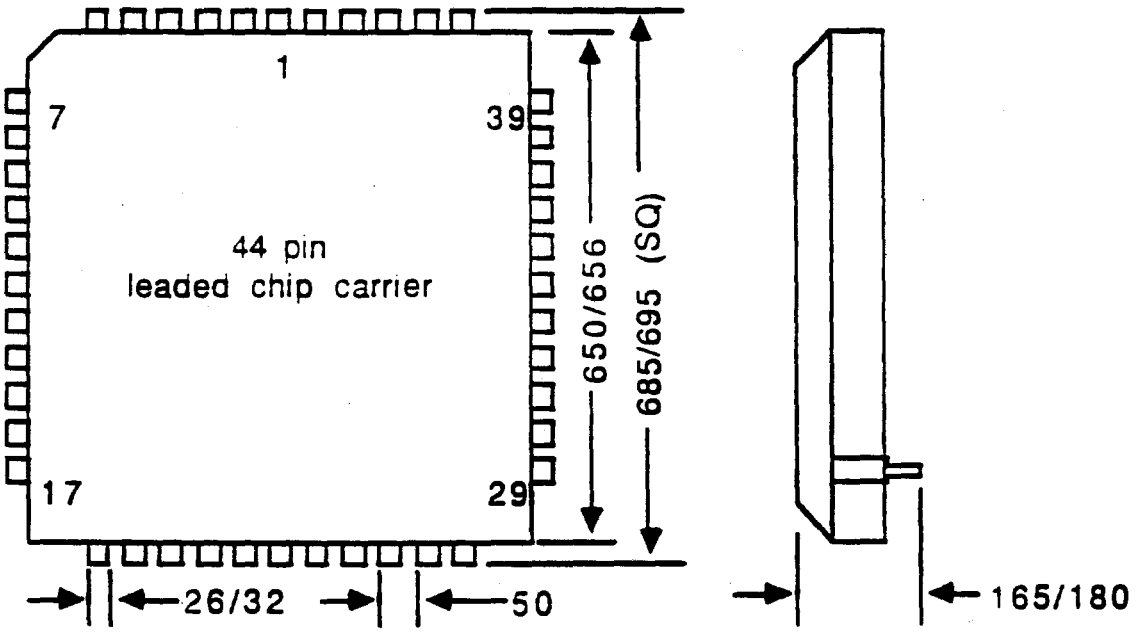
Part Number IC-04



BOTTOM VIEW

Created 8-28-89
Revised 8-29-89
Merle Haldeman / Jim Hoff

IC-04: SSD 5-CHANNEL LOGIC and OCTAL NHit
Cavity Down



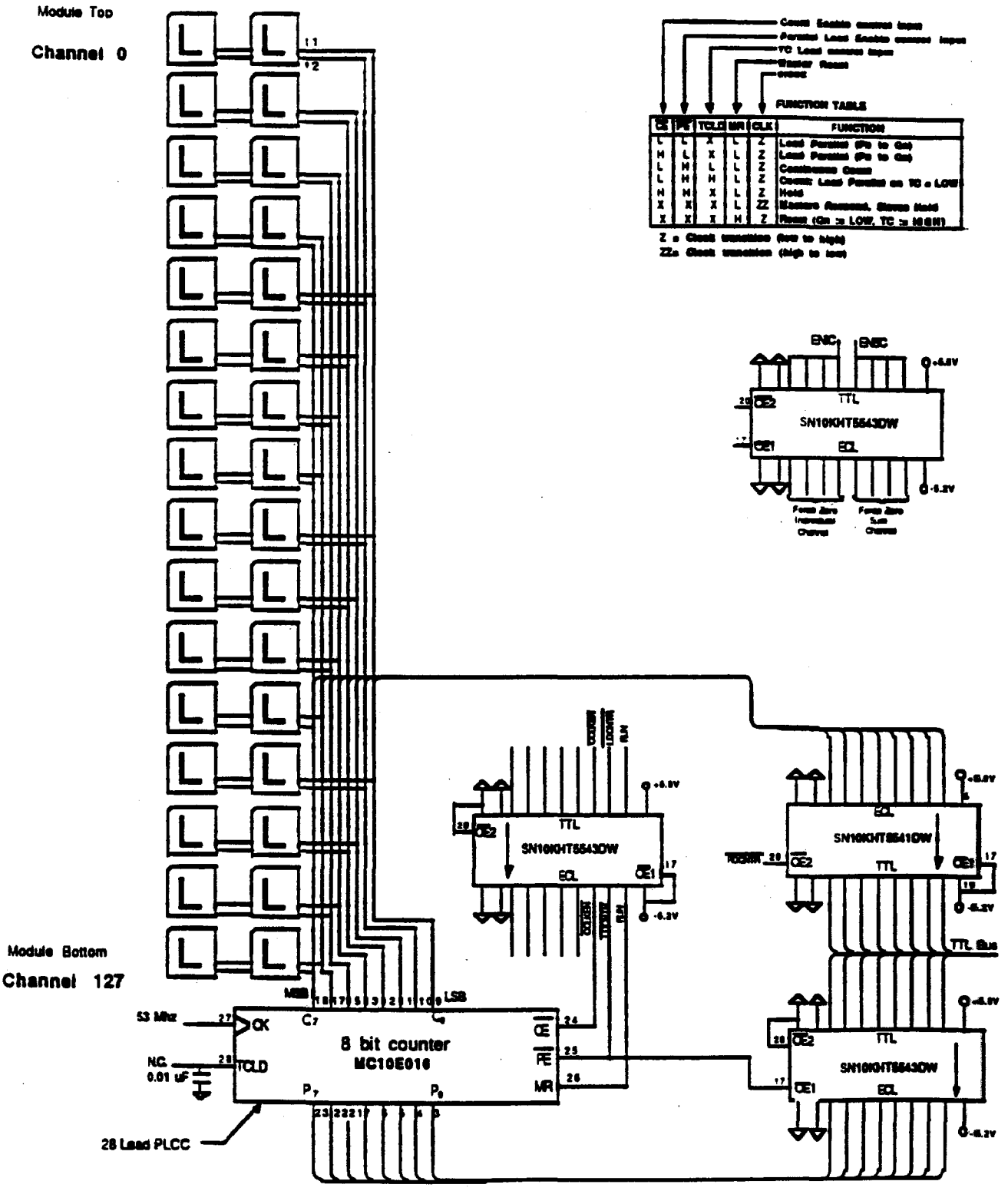
TOP VIEW

Part Number IC-04

IC-04 Package Dimension Diagram Notes.

1. All dimensions are in mils. (Min/Max)

Created 5-18-89
Revised 6-30-89
Merle Haldeman / Jim Hoff



Postamp / Comparator (Discriminator) Module Test Generation

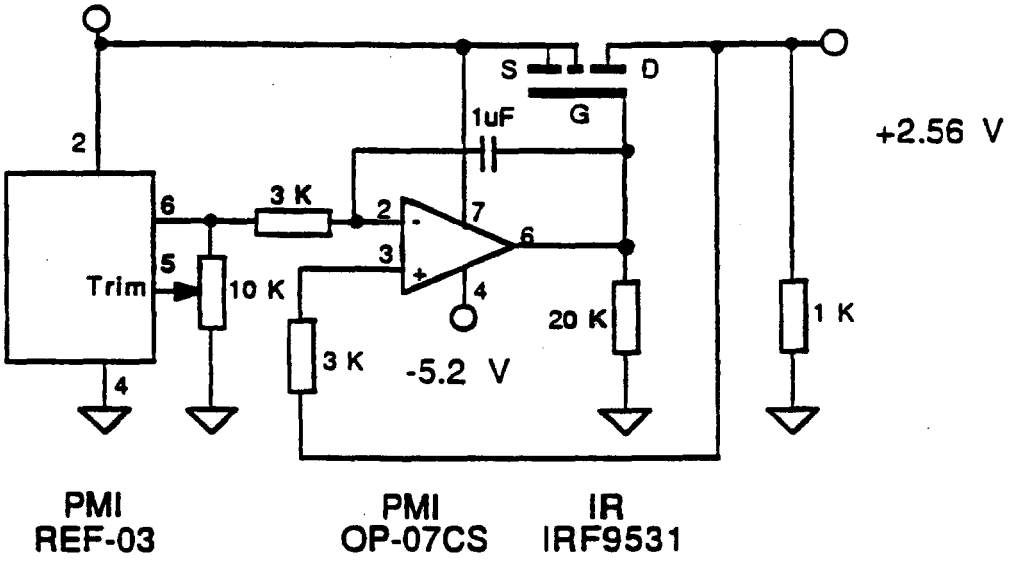
Merle Haldeman 3-27-89
Revised 9-19-89

SSD: POSTAMP/COMPARATOR Module

2.5 Volt Reference Voltage for IC-03 A/D Chip

Merle Haldeman
9/22/89

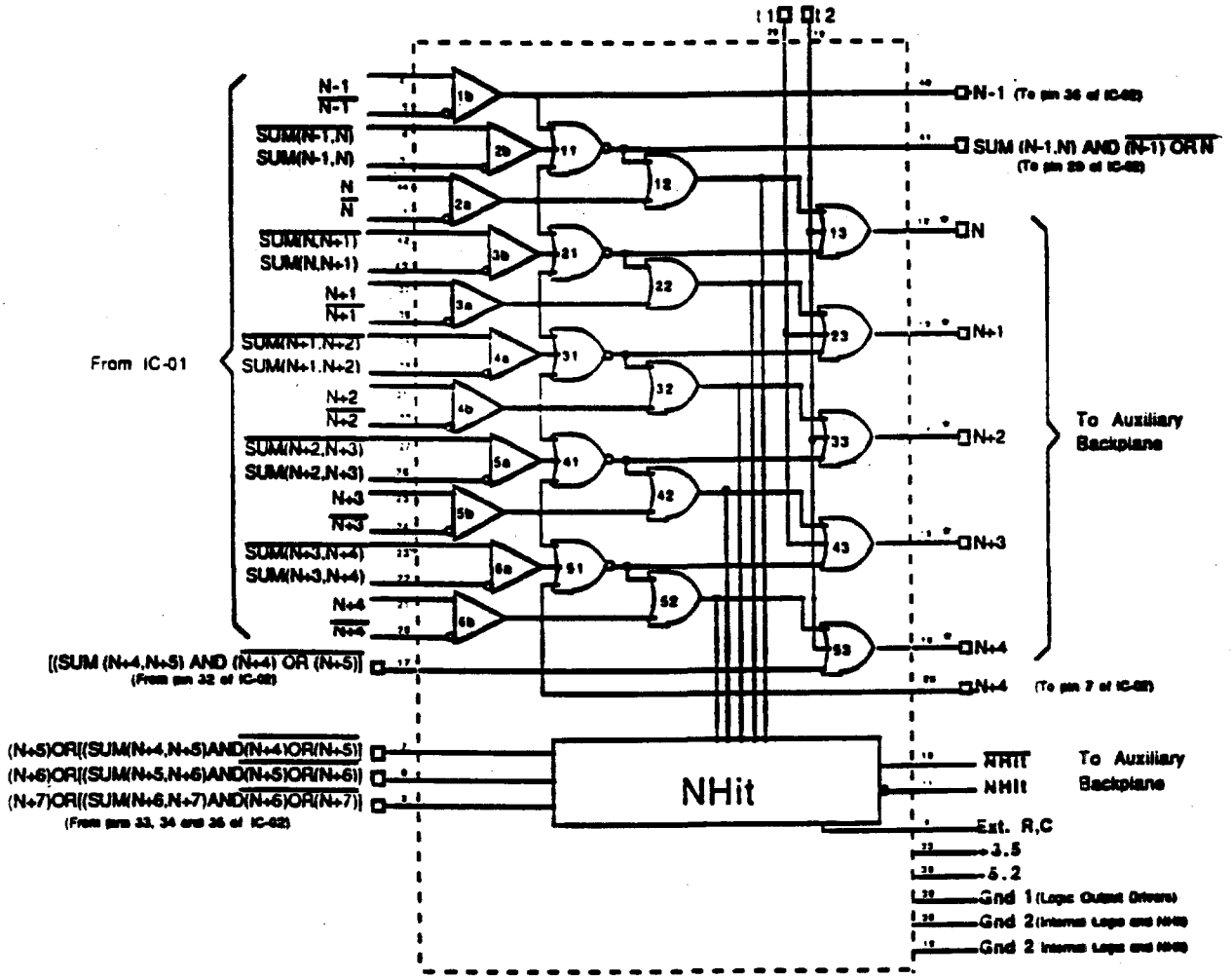
The IC-03 D/A Chip utilized for setting threshold voltages on the P/C Module requires a 2.5 volt reference voltage capable of delivering 3.5 mA per package. With the maximum requirement of 64 packages (256 D/A's) the load on the reference could be as much as 224 mA. One possible design utilizes a voltage reference as the basic stabilizing element, as illustrated below, followed by an opamp and MOSFET for buffering the output.



This basic circuit, with proper heat sinking, should be capable of handling loads up to 6 amperes. In our application, the dissipation should not exceed 600 mW during normal operation.

IC-04: SSD 5-Channel Logic and Octal NHit

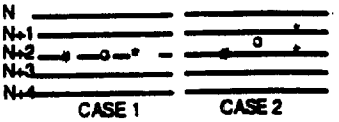
Part Number IC-04 V2.0



☐ ECL Levels

INPUTS					OUTPUTS	
N	SUM(N,N+1)	N+1	SUM(N+1,N+2)	N+2	N+1	N+1 OR [***]
X	X	1	X	X	1	1
0	1	0	X	X	1	1
X	X	0	1	0	1	

All other input conditions produce zero outputs
X = Don't Care



o Location of particle in SSD
Signals to NHit
* Signals to Auxiliary Backplane

Power dissipation 350 mW Outputs unloaded.
Power dissipation 500 mW All outputs loaded with 100 ohms to -2.0 Volts.

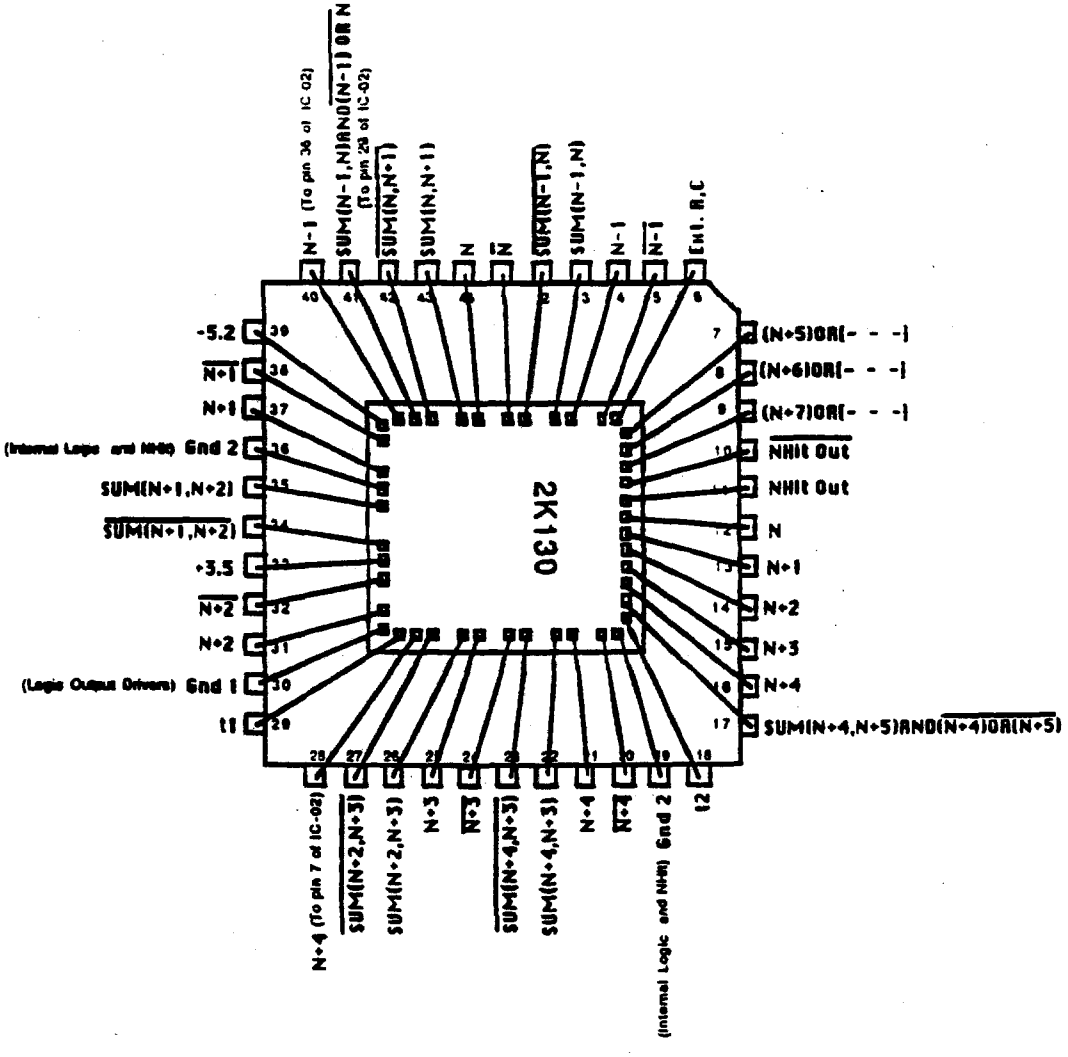
43 Pads Used

Created 4-23-89
Revised 10-3-89
Mario Haldeman / Jim Hoff

IC-04: SSD 5-CHANNEL LOGIC & OCTAL NHit Cavity Dowm

Thermally Enhanced, 44 Lead Plastic Chip Carrier

Part Number IC-04, V2.0



BOTTOM VIEW

Created 8-28-89
Revised 10-3-89
Merle Haldeman / Jim Hoff

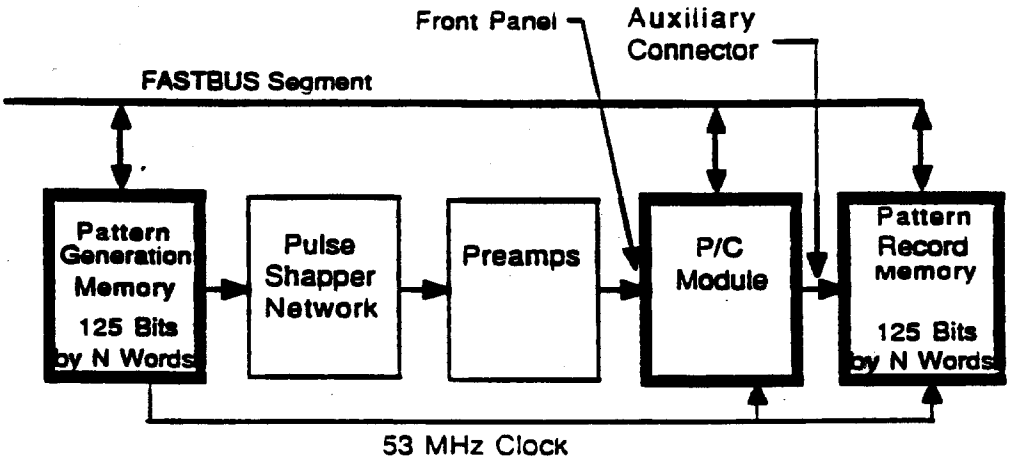
Module Pinout

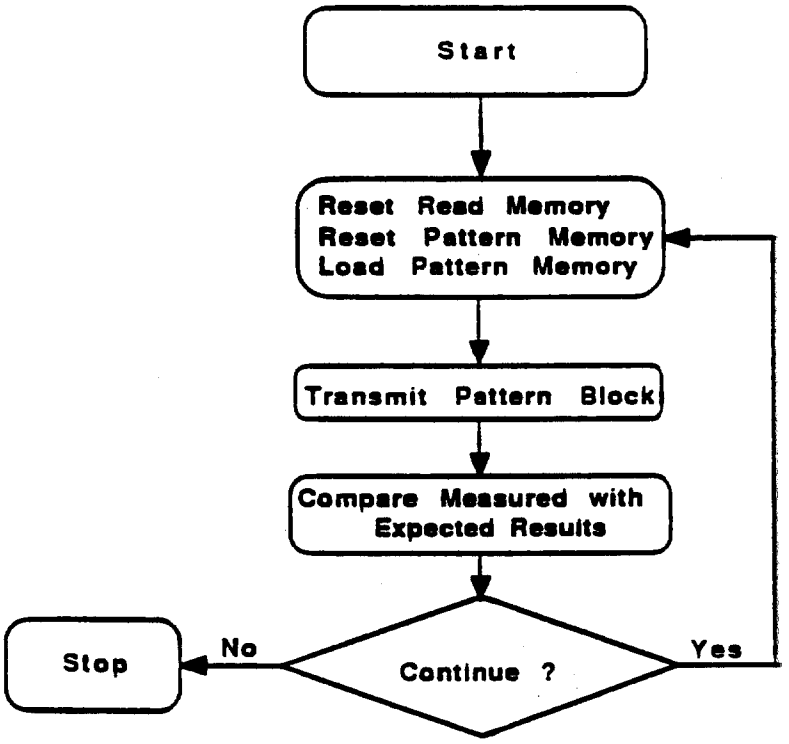
(Viewed from Front of FASTBUS Crate)

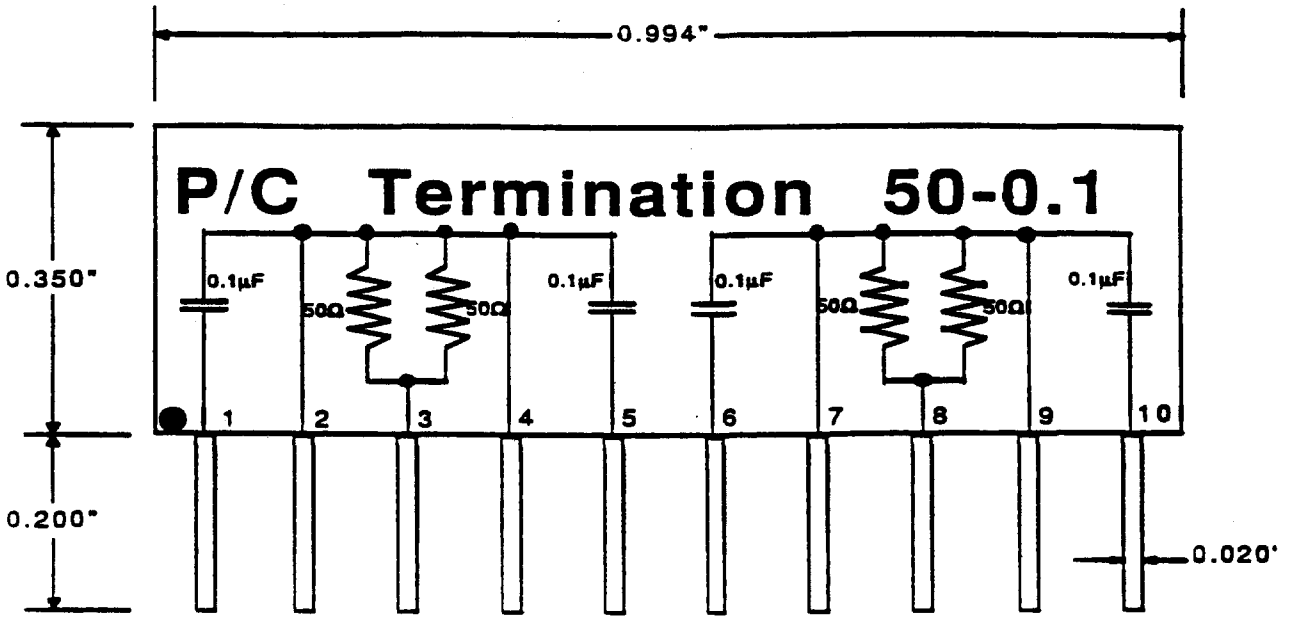
C01	53 MHz. 01 Clock	B01	Post / Comp Ch-01	A01	Post / Comp Ch-00
C02	GND	B02	Post / Comp Ch-03	A02	Post / Comp Ch-02
C03	PA / E Spare 0	B03	Post / Comp Ch-05	A03	Post / Comp Ch-04
C04	PA / E Spare 0	B04	Post / Comp Ch-07	A04	Post / Comp Ch-06
C05	PA / E Spare 0	B05	Post / Comp Ch-09	A05	Post / Comp Ch-08
C06	PA / E Spare 0	B06	Post / Comp Ch-11	A06	Post / Comp Ch-10
C07	PA / E Spare 0	B07	Post / Comp Ch-13	A07	Post / Comp Ch-12
C08	Reset	B08	Post / Comp Ch-15	A08	Post / Comp Ch-14
C09		B09	Post / Comp Ch-17	A09	Post / Comp Ch-16
C10		B10	Post / Comp Ch-19	A10	Post / Comp Ch-18
C11	GND	B11	Post / Comp Ch-21	A11	Post / Comp Ch-20
C12	Analog Sum 0-	B12	Post / Comp Ch-23	A12	Post / Comp Ch-22
C13	Analog Sum 0+	B13	Post / Comp Ch-25	A13	Post / Comp Ch-24
C14	Digital Sum 0-	B14	Post / Comp Ch-27	A14	Post / Comp Ch-26
C15	Digital Sum 0+	B15	Post / Comp Ch-29	A15	Post / Comp Ch-28
C16	Analog Sum 1-	B16	Post / Comp Ch-31	A16	Post / Comp Ch-30
C17	Analog Sum 1+	B17	Post / Comp Ch-33	A17	Post / Comp Ch-32
C18	Digital Sum 1-	B18	Post / Comp Ch-35	A18	Post / Comp Ch-34
C19	Digital Sum 1+	B19	Post / Comp Ch-37	A19	Post / Comp Ch-36
C20	GND	B20	Post / Comp Ch-39	A20	Post / Comp Ch-38
C21	Analog Sum 2-	B21	Post / Comp Ch-41	A21	Post / Comp Ch-40
C22	Analog Sum 2+	B22	Post / Comp Ch-43	A22	Post / Comp Ch-42
C23	Digital Sum 2-	B23	Post / Comp Ch-45	A23	Post / Comp Ch-44
C24	Digital Sum 2+	B24	Post / Comp Ch-47	A24	Post / Comp Ch-46
C25	Analog Sum 3-	B25	Post / Comp Ch-49	A25	Post / Comp Ch-48
C26	Analog Sum 3+	B26	Post / Comp Ch-51	A26	Post / Comp Ch-50
C27	Digital Sum 3-	B27	Post / Comp Ch-53	A27	Post / Comp Ch-52
C28	Digital Sum 3+	B28	Post / Comp Ch-55	A28	Post / Comp Ch-54
C29	GND	B29	Post / Comp Ch-57	A29	Post / Comp Ch-56
C30		B30	Post / Comp Ch-59	A30	Post / Comp Ch-58
C31	Analog Sum 4-	B31	Post / Comp Ch-61	A31	Post / Comp Ch-60
C32	Analog Sum 4+	B32	Post / Comp Ch-63	A32	Post / Comp Ch-62
C33	Digital Sum 4-	B33	Post / Comp Ch-65	A33	Post / Comp Ch-64
C34	Digital Sum 4+	B34	Post / Comp Ch-67	A34	Post / Comp Ch-66
C35	Analog Sum 5-	B35	Post / Comp Ch-69	A35	Post / Comp Ch-68
C36	Analog Sum 5+	B36	Post / Comp Ch-71	A36	Post / Comp Ch-70
C37	Digital Sum 5-	B37	Post / Comp Ch-73	A37	Post / Comp Ch-72
C38	Digital Sum 5+	B38	Post / Comp Ch-75	A38	Post / Comp Ch-74
C39	GND	B39	Post / Comp Ch-77	A39	Post / Comp Ch-76
C40	Analog Sum 6-	B40	Post / Comp Ch-79	A40	Post / Comp Ch-78
C41	Analog Sum 6+	B41	Post / Comp Ch-81	A41	Post / Comp Ch-80
C42	Digital Sum 6-	B42	Post / Comp Ch-83	A42	Post / Comp Ch-82
C43	Digital Sum 6+	B43	Post / Comp Ch-85	A43	Post / Comp Ch-84
C44	Analog Sum 7-	B44	Post / Comp Ch-87	A44	Post / Comp Ch-86
C45	Analog Sum 7+	B45	Post / Comp Ch-89	A45	Post / Comp Ch-88
C46	Digital Sum 7-	B46	Post / Comp Ch-91	A46	Post / Comp Ch-90
C47	Digital Sum 7+	B47	Post / Comp Ch-93	A47	Post / Comp Ch-92
C48	GND	B48	Post / Comp Ch-95	A48	Post / Comp Ch-94
C49		B49	Post / Comp Ch-97	A49	Post / Comp Ch-96

C50	GND	B50	Post / Comp Ch-99	A50	Post / Comp Ch-98
C51		B51	Post / Comp Ch-101	A51	Post / Comp Ch-100
C52		B52	Post / Comp Ch-103	A52	Post / Comp Ch-102
C53		B53	Post / Comp Ch-105	A53	Post / Comp Ch-104
C54		B54	Post / Comp Ch-107	A54	Post / Comp Ch-106
C55		B55	Post / Comp Ch-109	A55	Post / Comp Ch-108
C56		B56	Post / Comp Ch-111	A56	Post / Comp Ch-110
C57		B57	Post / Comp Ch-113	A57	Post / Comp Ch-112
C58		B58	Post / Comp Ch-115	A58	Post / Comp Ch-114
C59		B59	Post / Comp Ch-117	A59	Post / Comp Ch-116
C60		B60	Post / Comp Ch-119	A60	Post / Comp Ch-118
C61	GND	B61	Post / Comp Ch-121	A61	Post / Comp Ch-120
C62		B62	Post / Comp Ch-123	A62	Post / Comp Ch-122
C63	GND	B63	Post / Comp Ch-125	A63	Post / Comp Ch-124
C64	VCC, +5.0 Volts	B64	Post / Comp Ch-127	A64	Post / Comp Ch-126
C65	VEE, - 5.2 Volts	B65	GND	A65	VEE, - 5.2 Volts

P/C Module Test Scheme







Specifications:

Physical Requirements

Maximum Package Height: 0.350"
Maximum Package Length: 1.000"
Maximum Package Thickness: 0.100"
Package Marking: "P/C TERMINATION 50-0.1"

Electrical Requirements

Resistors
Value: 50Ω, +/- 2%
Power: 100 mW @ 70 C
TCR: +/-100PPM
Ratio Match between all four resistors +/-1%

Capacitors
Dielectric Type: Z5U
Value: 0.1µF, +80,-20%
Working Voltage: 25 Volts

Fermi National Accelerator Laboratory

January 4, 1991

TO: Distribution
FROM: Carl Swoboda
SUBJECT: [redacted]ardware Description

The attached document is the "as built" hardware description for the SSC DeLay/Encoder module. Please add this document to your SSD Readout System binder.

Distribution:
David Christian
Brad Cox (E771)
Peter Garbincius
Franco Grancagnolo (E771,MS219)
Hector Gonzalez
Bill Haynes
Rick Kwarciany
Wolfgang Kowald (E771)
Mark Larwill
Cash McManus (E771)
Garry Moore
Tom Nash
John Peoples
Ruth Pordes
Matteo Racagni (E771)
Robert Trendler
Ken Treptow

HN 101

Fermi National Accelerator Laboratory

DELAY/ENCODER

HARDWARE DESCRIPTION

H.L. Gonzalez, J. Chramowicz, R. Reitz

November, 1990

Table of Contents

1. General Information

1.1. Purpose

The front end readout electronics for the Silicon Strip Detector is designed to process data at the RF bucket frequency, 53MHz. The Delay/Encoder(DE) module has been specified to accept data at 53MHz, provide a delay mechanism while a trigger decision is made, and generate an address hit list upon a Level 1 accept signal. A simplified block diagram is provided in Figure 2.

The delay element continuously stores data while the level 1 system is processing data corresponding to previously stored events. The delay is implemented in RAM and it is required for the control system to map the level 1 decisions into an eight bit address. The current implementation assumes that processing of an event takes about 1μsecond from the time that it is loaded into the DE. It is mandatory that the event acceptance be time ordered and the decision time be fixed with respect to the event occurrence. The address of accepted events is broadcasted by the Master Timing Controller to all Sequencer modules in the system and each Sequencer addresses the DE in its crate. The addressing mechanism triggers the DE to read the event from memory and transfer it to the encoder section.

The data encoding scheme uses the trigger bucket and the previous bucket simultaneously to generate an address hit list. A flag is asserted whenever the previous bucket has the bit set for the address being output. The address hit list is transmitted synchronously to a crate Sequencer module which serves as a crate controller and event builder for two planes of silicon strip data. The Sequencer is capable of transmitting hit data over fiber optic at 40Mbytes/sec or read out by a Fastbus master.

This document includes figures and timing diagrams intended to simplify the specifications. In some cases, specifically the timing diagrams, the information is an attempt to specify the module and its interface with other system components.

1.1.1. Silicon Strip Readout System

This section presents a simplified block diagram of the silicon strip readout system. Figure 1 shows the interconnection of all the modules that are referenced in this document. A brief description of each module follows.

PC - PostAmp/Comparator board, 12 per crate. Processes 128 pre-amp silicon strip signals, outputs discriminated data to the Delay/Encoder and outputs analog and digital sums to Level 1.

DE - Delay/Encoder board, 12 per crate. Provides event buffering for the PC discriminated data and for level 1 accepted event transmits a hit list to the Sequencer.

SEQ - Sequencer board, 1 per crate. Fans out system clock to PC and DE, initiates the encoding of a event, stores, pipelines and transmits encoded events to the next level. The events can be readout by a FASTBUS master.

FSCC- Fastbus Smart Crate Controller board, 1 per crate. Initializes the crate by exercising control over the SEQ, runs local diagnostics and provides an alternate data path to readout events.

MTC- Master Timing Control board, 1 per silicon strip readout system. The MTC synchronizes the 12 SEQ in the system by providing timing and control. Some of the functions that it performs are listed below:

- Distributes the RF clock to all SEQs.
- Maps a level 1 accept signal into an address of the DE memory and transmits addresses to all SEQs.
- Queue level 1 accepted events.
- Controls the write enable signal for the DE.
- Responds to READY and ERROR condition from the SEQs.
- Interface with the overall experiment controller.
- Synchronize System.

The readout system consists of 12 readout crates and a control crate that contains the MTC and other special modules. Each of the readout crates processes two planes of silicon strip data. Data processing is done in groups of 128 strips by a PC and DE pair. The 12 DEs in a crate send data, in parallel, to the crate SEQ. For a formal description of the system refer to the 'Silicon Strip Readout Implementation Plan' document.

In the context of this specification an **event** is the output of the PC and they are generated every 18.9nanoseconds.

1.2. Application

The DE is being designed for the Silicon Strip Readout System for E771 and E789. The function of the module is hardwired and there is no other application for it beyond the ones described on this document.

1.3. Packaging

The board is a single width FASTBUS module that does not contain a FASTBUS interface.

1.4. Power Requirements

The maximum and typical current for the module are listed below.

Voltage	Current Max.	Typical Current
-5.2V	17A	14A
-2.0V	6A	6A
+5.0V	<.5A	

The typical power dissipation for the module is 85 W. For protection fuses and tranzsorbs are used for each power supply. The recommended number of fuses is: -5.2v - three 5A and one 3A , -2.0v - three 3A and 5.0v - one 1/2A.

1.5. Cooling Requirements

The module will operate at the temperature range provided by the FASTBUS cooling system.

2. Theory of Operation and Operating Modes

The DE module is a single width board packaged in FASTBUS that does not implement a FASTBUS interface. The DE accepts PC discriminated data from 128 silicon strips, provides event buffering, encodes and transmits accepted events to a crate SEQ. The module communicates with the PC and SEQ through a special FASTBUS auxiliary backplane. The auxiliary connector signals for the DE are described in Appendix A.

The module is divided in two independent functions; the Delay element and the Encoder, see Figure 2. The following sections provide a brief description for each function.

2.1. Delay

The Delay element receives 128 channels of discriminated data from the associated PC and provides buffering for 256 events (~4.8 μseconds). During data acquisition the DE continuously stores data in a FIFO like memory, while the level 1 system is making decisions for previously stored events. The DE does not implement any logic to prevent overwriting interesting events. This operation is delegated to the MTC which keeps track of the system write pointer (for DE) and the queued events. For system implementation reasons, it is mandatory that event acceptance be time ordered and the decision time be fixed with respect to the event occurrence.

The delay element control logic requires that the 53MHz input (CLK2) be a 50% duty cycle clock. The logic splits the 18.9 nanoseconds time slice of each bucket into a read and write periods for a combined bandwidth of 106MHz. The write operation uses an address counter clocked by CLK2 and a write enable signal (WRITE*, asserted low) generated by the MTC and distributed in each crate by the SEQ.

The WRITE* signal is send 128 cycles before the SYNC* pulse. The DE retime the signal with the SYNC* to synchronize the start of event acquisition, see timing 2. The DE starts writing data synchronously with SYNC*, but stops writing data asynchronously when WRITE* is deasserted. The DE will track their synchronization by checking that the write address is zero when the SYNC* signal is asserted. Note that prior to the assertion of WRITE* the DEs had been reset, which forces the write address to zero.

The readout of an event occurs when the MTC receives an accept pulse from the level 1 trigger system. The MTC maps the pulse into an address for the DEs memory and sends the address to all SEQs in the system. The address generated by the MTC shall correspond to the previous bucket location. The previous and accepted buckets are loaded into registers and the Encoder is enabled to begin encoding that particular event.

2.2. Encoder

The Encoder is a simultaneous two bucket hit-list address generator. The encoding is performed in two stages, byte and bit levels, see Figure 3. At load time, the byte encoder performs byte integration, see Figure 4. The output of this process is a 16-bit word with a bit set for bytes with hit channels. At encoding time the byte encoder sequentially selects bytes to be

processed by the bit encoder. The bit encoder (block ENCODER-8) loads the input data when ready and outputs a byte wide address stream of asserted bits. The encoded address is formed by concatenating the byte address with the bit address. In addition the bit encoder sets a flag (DATA0) whenever the previous bucket has a hit for the address being output. The hit list is generated from low to high address and no hit count is generated by the DE. An example of byte encoding is shown below:

Bit # Bucket	Previous Bucket	Trigger Bucket	Hit Type	Bit ADD (HEX)	Flag
7	1	0	A	7	1
6	0	0	-	-	-
5	0	1	B	5	0
4	1	1	C	4	1
3	0	0	-	-	-
2	0	0	-	-	-
1	1	1	C	1	1
0	0	1	B	0	0

Note: The encoder never looks at addresses without hits, this is represented with "-".

The option to remove type A hits is implemented on the prototype. There are two options for the address hit list data transfer, 53MHz or 26.5MHz, CLK2 and CLK3 respectively. These options are switch selectable for each DE. Currently only the 26.5MHz transfers have been tested and the present version of the SEQ does not support the 53MHz transfer rate.

From the encoding table shown above two modes of operation are derived: **Mode 1** in which type A, B and C hits are included; **Mode 2** where type B and C hits are included.

3. Input/Output Specifications

The DE is a two port module, PC port and a SEQ port, See Figure 2. Both of these ports are implemented on an application specific Fastbus Auxiliary backplane that pairs a PC with a DE and has separate connections between each DE and the SEQ.

3.1. Communication Interfaces

3.1.1. PostAmp/Comparator Port

The PC port is a 128 bit uni-directional single-ended ECL connection from a PC to the associated DE. Synchronization of this port is controlled by the SEQ supplied CLK1 and CLK2 clocks. CLK1 is remotely programmed through FASTBUS and the rising edge is used by the PC to latch the silicon strip discriminated data. CLK2 is referenced (delayed) to CLK1 such that the PC output data is valid while the DE is asserting the Write* signal for the memories, see Timing 1. It is required that data on this port be valid for at least 12 nanosececonds simultaneously at all DE in a crate.

3.1.2. Sequencer Port

The Sequencer port is the access port for the SEQ to readout the level 1 accepted events. This data port is a byte wide point-to-point connection designed to support a 53MHz data transfer rate. Currently this port is operated a 26.5MHz.

The port provides an address bus, a data bus and control signals. To initiate a transfer the Sequencer supplies the DE with an event address and asserts the Add_Valid signal. After a fixed delay the Encoder will assert a Data_Valid signal (if hits present) and start transmitting the address hit list to the SEQ until completion. The data transfer is synchronous with CLK2 or CLK3 depending on the user selected encoder operating frequency. A non-detailed timing diagram is provided in Timing 4.

3.1.3. Front Panel

The DE front panel is intended to provide information that will help diagnose data encoding problems, refer to Figure 5 for the layout of the front panel. The front panel provides the byte mask (B1_Byt(0:7) and B2_Byt(0:7)) which describes the bytes that have hits. Note that the byte mask provides a bit for each of the 16 bytes of the 128 channels. In addition the front panel provides the bytes that will be loaded next by the Bit_Encoder (TB(0:7) and RPB(0:7)). This information combined with the signal Clk_Byt* provides everything that is required to trace the encoding operation of the DE. Description of the front panel signals is provided on appendix A.

Other signal provided are TC*, 53M* and a sync error led.

4. Initialization

Initialization of the DEs is achieved by deasserting the Write_En* signal and pulsing the SSD Reset signal. Then the MTC initiates the enable of the write process and after the appropriate delay the enable of the Level 1 system. The DE will start writing data synchronously after receiving Sync* when the Write _En* signal is asserted.

After reset, the write counter points to location zero, the Encoder is in the ready state and all control signals driven by the DE are negated. The read counter is not initialized because it is loaded on demand when an event is accepted.

5. Module Diagnostics

5.1. Hardware Test

The DE does not implement any internal diagnostic tests or FASTBUS interface to access it's memory. The decision of excluding these features is based on timing and power considerations.

The Test Stand Module (TSM) is the test module used to debug the DE. The TSM implements a 256 x 128 memory used to emulate the PC output port and one port of the SEQ. The TSM is operated by a FASTBUS master, i.e. the FSCC. In brief the user will load the memory, initiate the data transfer from TSM to DE and then request events to be encoded by the DE (from it's memory). The encoded event is read out from the TSM and compared with the expected result. A detailed description of the DE software diagnostics is provided in the Software Document **PN434**.

5.2. System Test

For system diagnostics the PC provides a maximum of 256 data patterns generated by a counter. The counter can be programmed to count or hold the loaded value. The data is stored by the DE into it's memory an events to be encoded are requested through the SEQ. Synchronization is achieved by the initialization of the SSD system and from the match between the 256 different patterns and the 256 memory locations on the DE.

6. Appendix A

This appendix describes the DE auxiliary connector signals used on the silicon strip readout crates and the signals provided in the front panel. In addition any other signals of interest are described.

A.1 Auxiliary Connector

CLK1 - A 53MHz clock driven by the SEQ and used by the PC to latch data. This clock is remotely programmable through the FASTBUS port on the SEQ.

CLK2 - A 53MHz clock driven by the SEQ. The clock is a delayed version of CLK1 used to synchronize the write process in the DE with the output data of the PC, to generate internal timing and as a reference when transmitting data to the SEQ at 53MHz.

CLK3 - A 26.5MHz clock driven by the SEQ. The clock used to transmit data to the SEQ. The DE samples the 26.5MHz on the backplane with the internal 53MHz clock.

DI(0:127) - Input discriminated data driven by the associated PC. A 100 ohms termination is provided by the DE.

Address(0:7) - The address bus (bussed to all DE) driven by the SEQ to transfer event addresses to the DE.

Add_Valid - Signal driven by SEQ to validate ADD(0:7).

Write_En* - Write enable signal distributed on the backplane by the SEQ. The signal is controlled by the MTC or through a FASTBUS register on the SEQ.

Sync* - Synchronization signal generated by the MTC and used by the DEs to test write counter synchronization at each zero crossing. If a DE has a

write counter different from zero then it is out of synchronization.

Sync_Err - Signal asserted by a DE that is out of synchronization and received by the SEQ. The signal is wire-ored on the backplane.

Reset - Reset signal distributed on the backplane by the SEQ.

Data_X(0:7) - Encoded hit list data bus for Delay/Encoder X, where X is a hex number assigned to each DE. Terminated by the SEQ.

Data_ValidX - Data valid signal asserted by Delay/Encoder X, where X is a hex number assigned to each DE. Terminated by the SEQ.

A.2 Front Panel Signals

B1_Byt(0:7) * - This are the lower 8 bits of the byte encoding process implemented by the logic in Figure 4.

B2_Byt(0:7)* - This are the upper 8 bits of the byte encoding process implemented by the logic in Figure 4.

TC* - This signals is asserted low every time that the write counter of the DE is at FFh. This signal should remain at a fixed reference relative to the system Sync signal.

53M* - A buffered inverted sample of the 53MHz clock received by the DE from the backplane.

Error - RED LED that is on when the DE detects that is out of synchronization with respect to the Sync signal on the backplane.

Clk_Byt* - Clock pulse that signals that a new byte is being loaded into the Bit-Encoder. Also the lowest bit in

B1_Byt(0:7), B2_Byt(0:7) will be set, meaning that the corresponding byte is being processed.

TB(0:7)* - The Trigger bucket loaded into the Bit_Encoder when Clk_Byt* was pulsed.

RPB(0:7)* - The Previous bucket loaded into the Bit_Encoder when Clk_Byt* was pulsed.

7. Appendix B

This appendix covers the switches and switch setting for the DE. The switches or jumper (label as TP) points are implemented with wire-wrap pins to limit the problems of wrong set ups. For the operation mode refer to Section 2.2.

SW1 - Selects for the encoding clock (same clock used to transfer data to the SEQ). The default is pin-2 to pin-1, the 26MHz is selected

SW2 - Selects the clock that increments the write counter. The default is connect pin-2 to pin-3, the write counter is incremented with a signal derived from the write signal going to the memories.

SW3 - Selects the the select line for the address mux. The default is pin-2 to pin-3, use CLKB4 which is a 53MHz signal.

SW4 - This switch was implemented to disable the write signal during the time that an event was being read out into the registers (the write would be disable for 40nsec. approximately). The default is the open position, this is no wire.

Mode 1 Setting

TP1 and TP2 shorted, this allows both buckets to be included on the generation of the byte mask shown in Figure 4. In addition the following pairs on TP must be shorted: TP4-TP8, TP5-TP9, TP6-TP10, TP7-TP11, TP12-TP16, TP13-TP17, TP14- TP18, TP15-TP19.

Mode 2 Setting

TP1 and TP2 open, this disables the previous bucket from being included in the generation of the byte mask. In addition the set of pins mentioned in Model 1 (TP4 ..... TP19) must be open. The following pins are to be tied to an ECL high; TP4, TP5, TP6, TP7, TP12, TP13, TP14 and TP15.

8. Appendix C

Encoder Module Pin List

(Viewed From Front of Crate-10/1/90)

C01-N/C	B01-Post/Disc Ch.00	A01-Post/Disc Ch.01
C02-GND	B02-Post/Disc Ch.02	A02-Post/Disc Ch.03
C03-N/C	B03-Post/Disc Ch.04	A03-Post/Disc Ch.05
C04-GND	B04-Post/Disc Ch.06	A04-Post/Disc Ch.07
C05-GND	B05-Post/Disc Ch.08	A05-Post/Disc Ch.09
C06-GND	B06-Post/Disc Ch.10	A06-Post/Disc Ch.11
C07-GND	B07-Post/Disc Ch.12	A07-Post/Disc Ch.13
C08-Reset	B08-Post/Disc Ch.14	A08-Post/Disc Ch.15
C09-Sync	B09-Post/Disc Ch.16	A09-Post/Disc Ch.17
C10-GND	B10-Post/Disc Ch.18	A10-Post/Disc Ch.19
C11-GND	B11-Post/Disc Ch.20	A11-Post/Disc Ch.21
C12-Sync Err	B12-Post/Disc Ch.22	A12-Post/Disc Ch.23
C13-GND	B13-Post/Disc Ch.24	A13-Post/Disc Ch.25
C14-GND	B14-Post/Disc Ch.26	A14-Post/Disc Ch.27
C15-GND	B15-Post/Disc Ch.28	A15-Post/Disc Ch.29
C16-GND	B16-Post/Disc Ch.30	A16-Post/Disc Ch.31
C17-GND	B17-Post/Disc Ch.32	A17-Post/Disc Ch.33
C18-GND	B18-Post/Disc Ch.34	A18-Post/Disc Ch.35
C19-GND	B19-Post/Disc Ch.36	A19-Post/Disc Ch.37
C20-GND	B20-Post/Disc Ch.38	A20-Post/Disc Ch.39
C21-Hit Data 0	B21-Post/Disc Ch.40	A21-Post/Disc Ch.41
C22-Hit Data 1	B22-Post/Disc Ch.42	A22-Post/Disc Ch.43
C23-GND	B23-Post/Disc Ch.44	A23-Post/Disc Ch.45
C24-Hit Data 2	B24-Post/Disc Ch.46	A24-Post/Disc Ch.47
C35-Hit Data 3	B25-Post/Disc Ch.48	A25-Post/Disc Ch.49
C26-GND	B26-Post/Disc Ch.50	A26-Post/Disc Ch.51
C27-Hit Data 4	B27-Post/Disc Ch.52	A27-Post/Disc Ch.53
C28-Hit Data 5	B28-Post/Disc Ch.54	A28-Post/Disc Ch.55
C29-GND	B29-Post/Disc Ch.56	A29-Post/Disc Ch.57
C30-Hit Data 6	B30-Post/Disc Ch.58	A30-Post/Disc Ch.59
C31-GND	B31-Post/Disc Ch.60	A31-Post/Disc Ch.61
C32-Hit Data 7	B32-Post/Disc Ch.62	A32-Post/Disc Ch.63
C33-Data Valid	B33-Post/Disc Ch.64	A33-Post/Disc Ch.65
C34-GND	B34-Post/Disc Ch.66	A34-Post/Disc Ch.67
C35-26 MHz Clock	B35-Post/Disc Ch.68	A35-Post/Disc Ch.69
C36-GND	B36-Post/Disc Ch.70	A36-Post/Disc Ch.71
C37-Event Address Valid	B37-Post/Disc Ch.72	A37-Post/Disc Ch.73
C38-Event Address Wrt. En.	B38-Post/Disc Ch.74	A38-Post/Disc Ch.75
C39-GND	B39-Post/Disc Ch.76	A39-Post/Disc Ch.77
C40-Event Address 0	B40-Post/Disc Ch.78	A40-Post/Disc Ch.79
C41-Event Address 1	B41-Post/Disc Ch.80	A41-Post/Disc Ch.81
C42-GND	B42-Post/Disc Ch.82	A42-Post/Disc Ch.83
C43-Event Address 2	B43-Post/Disc Ch.84	A43-Post/Disc Ch.85
C44-Event Address 3	B44-Post/Disc Ch.86	A44-Post/Disc Ch.87
C45-GND	B45-Post/Disc Ch.88	A45-Post/Disc Ch.89
C46-Event Address 4	B46-Post/Disc Ch.90	A46-Post/Disc Ch.91
C47-Event Address 5	B47-Post/Disc Ch.92	A47-Post/Disc Ch.93
C48-GND	B48-Post/Disc Ch.94	A48-Post/Disc Ch.95
C49-Event Address 6	B49-Post/Disc Ch.96	A49-Post/Disc Ch.97
C50-Event Address 7	B50-Post/Disc Ch.98	A50-Post/Disc Ch.99
C51-GND	B51-Post/Disc Ch.100	A51-Post/Disc Ch.101
C52-GND	B52-Post/Disc Ch.102	A52-Post/Disc Ch.103
C53-GND	B53-Post/Disc Ch.104	A53-Post/Disc Ch.105

C54-GND	B54-Post/Disc Ch.106	A54-Post/Disc Ch.107
C55-GND	B55-Post/Disc Ch.108	A55-Post/Disc Ch.109
C56-GND	B56-Post/Disc Ch.110	A56-Post/Disc Ch.111
C57-GND	B57-Post/Disc Ch.112	A57-Post/Disc Ch.113
C58-GND	B58-Post/Disc Ch.114	A58-Post/Disc Ch.115
C59-GND	B59-Post/Disc Ch.116	A59-Post/Disc Ch.117
C60-GND	B60-Post/Disc Ch.118	A60-Post/Disc Ch.119
C61-GND	B61-Post/Disc Ch.120	A61-Post/Disc Ch.121
C62-H53MHZ,Ø2 Clock	B62-Post/Disc Ch.122	A62-Post/Disc Ch.123
C63-L53MHZ,Ø2 Clock	B63-Post/Disc Ch.124	A63-Post/Disc Ch.125
C64-N/C	B64-Post/Disc Ch.126	A64-Post/Disc Ch.127
C65-N/C	B65-GND	A65-N/C

9. Appendix D

Figures and Timing Diagrams

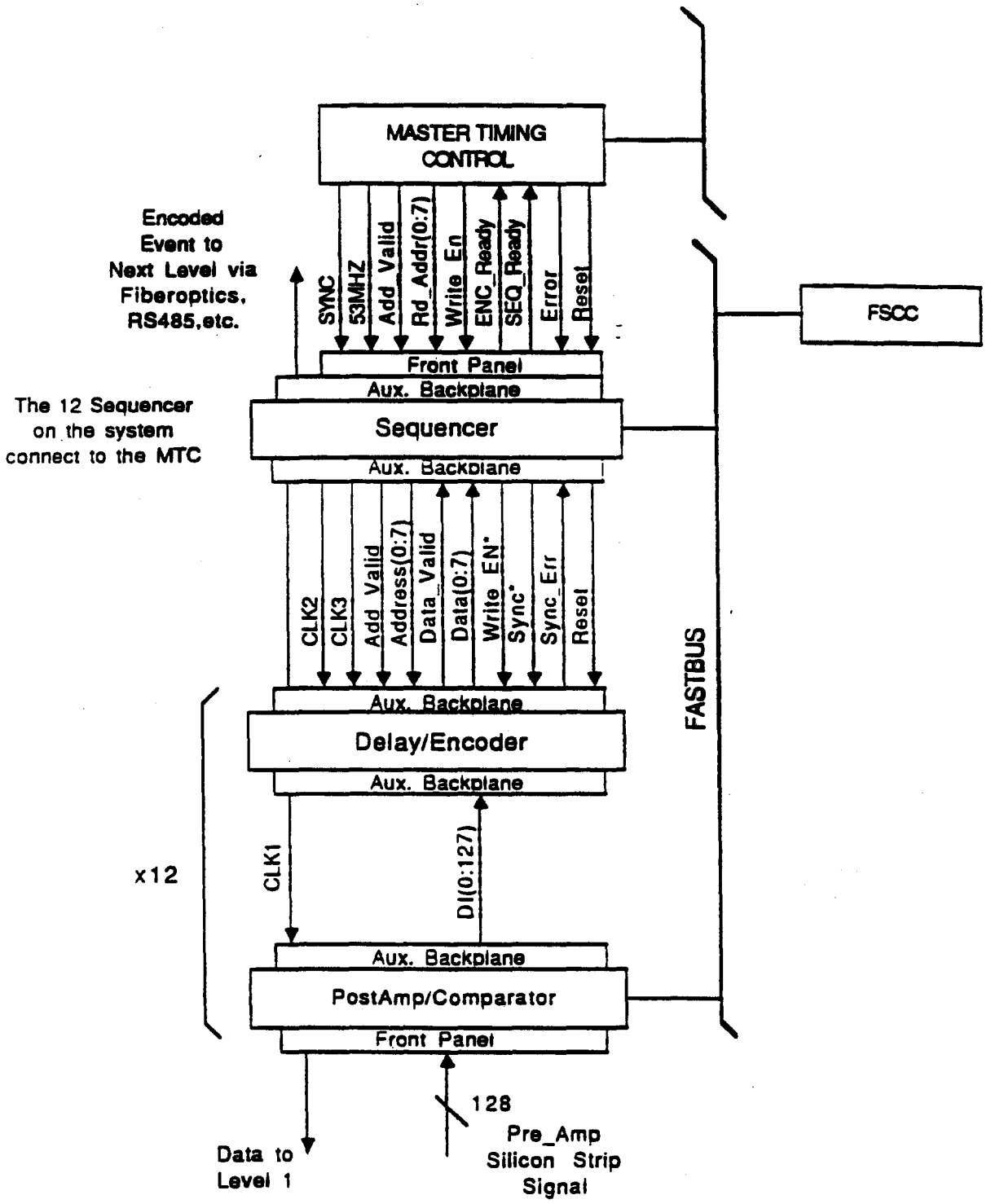
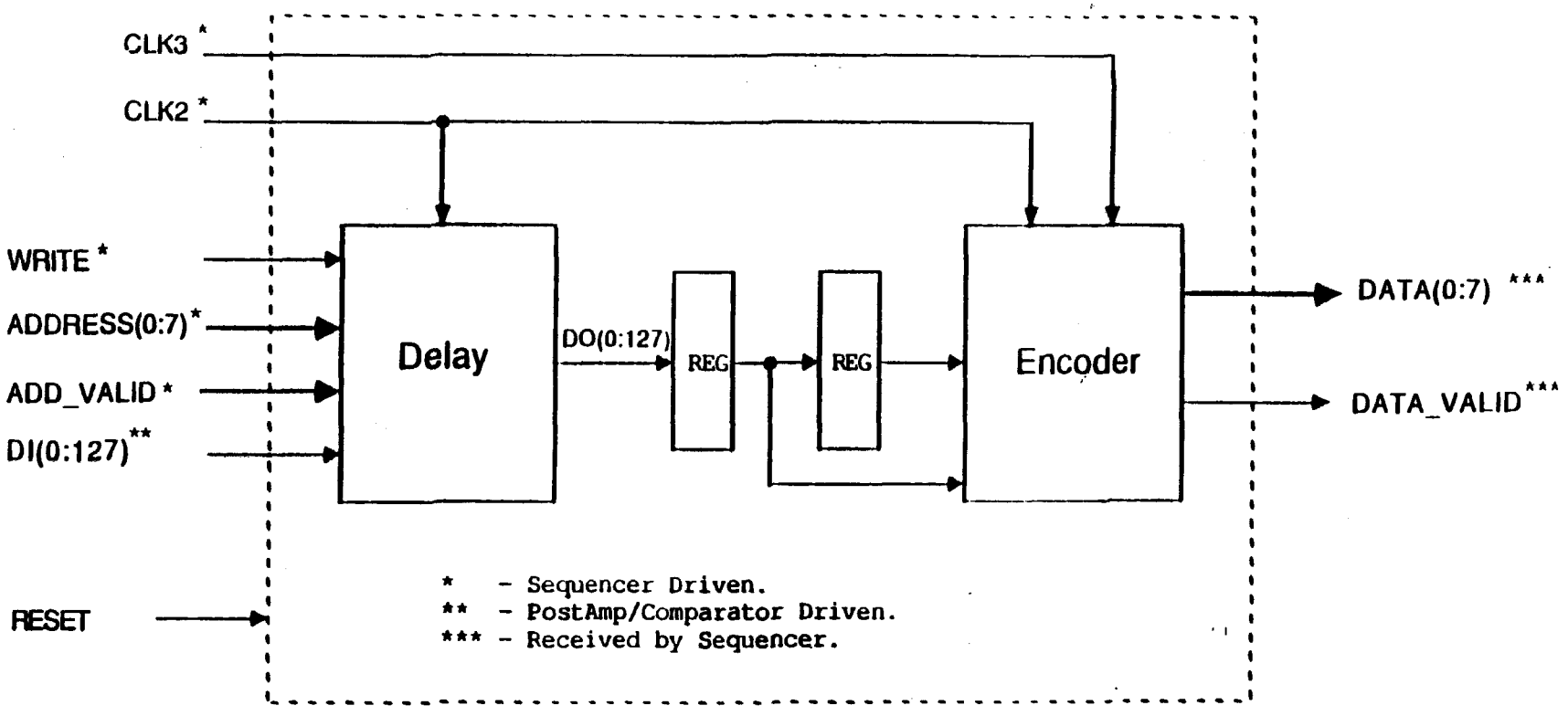


FIGURE 1: Silicon Strip Readout System



PC PORT:

DI(0:127)	- PC output data.

SEQ PORT:

WRITE	- Write enable for the delay memory.
ADDRESS(0:7)	- Accepted bucket address.
ADD_VALID	- Validates ADDRESS(0:7).
DATA(0:7)	- Channel to transmit the address hit list.
DATA_VALID	- Enable the Sequencer to clock data into its FIFO.

Figure 2: Delay/Encoder Block Diagram

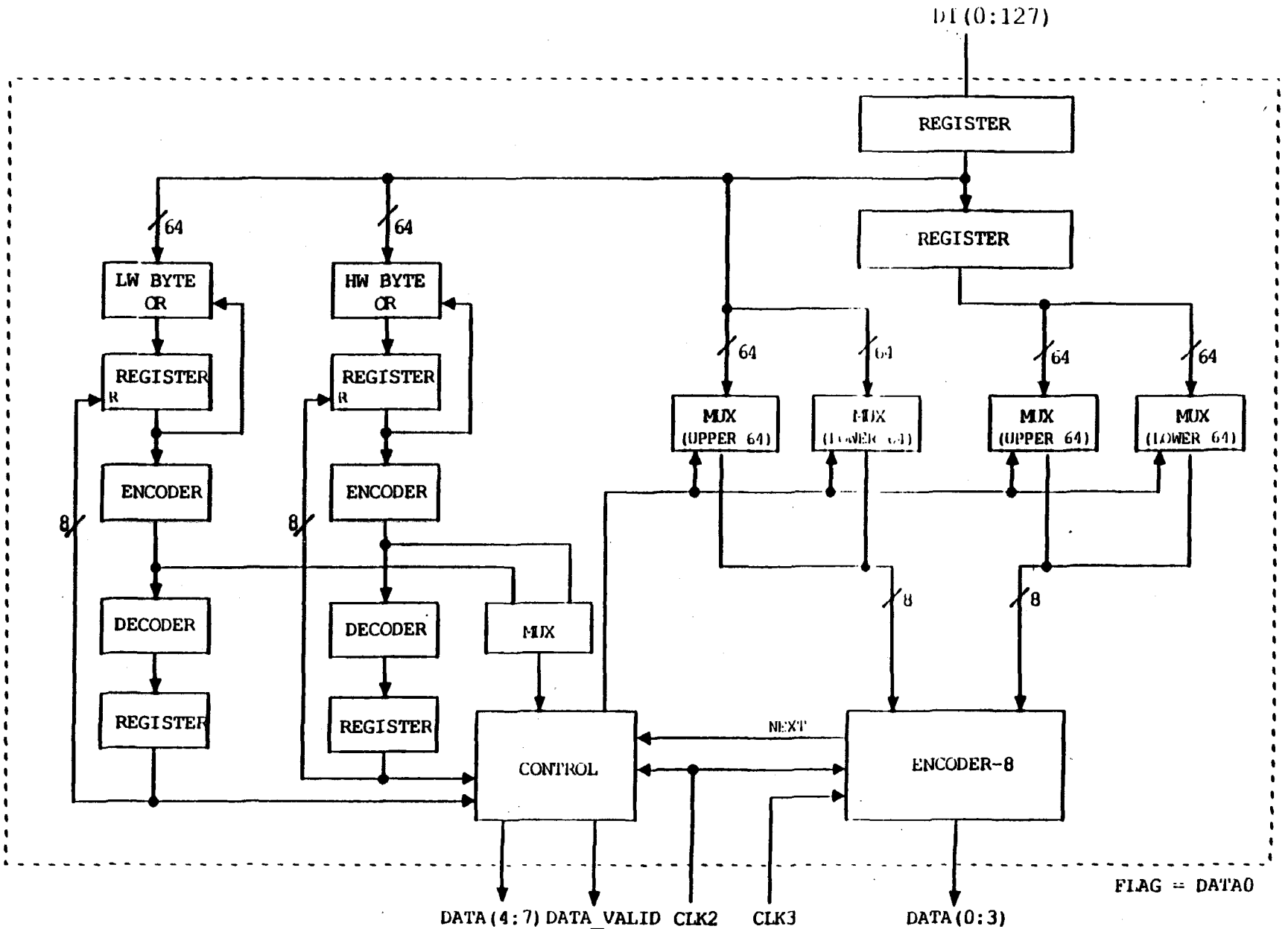


Figure 3: 128-BIT ENCODER

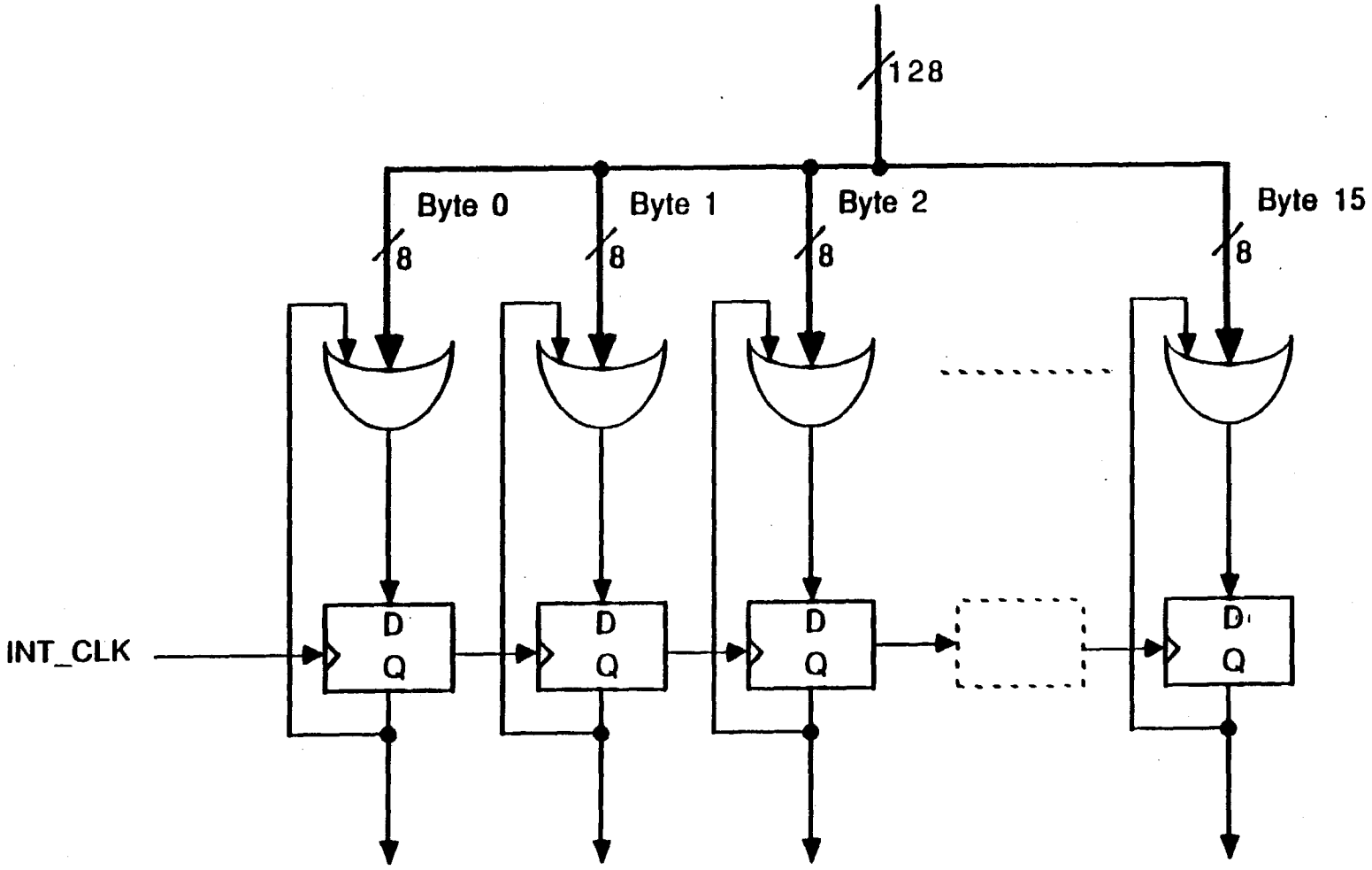


Figure 4: Byte Integration

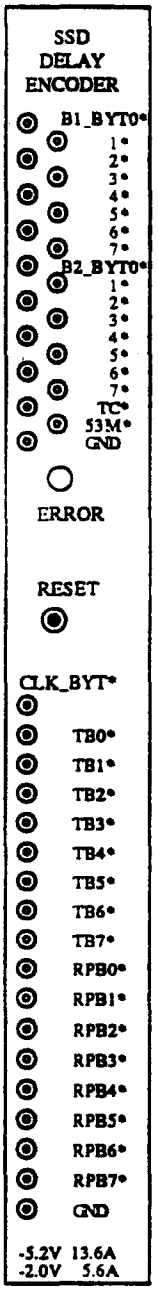
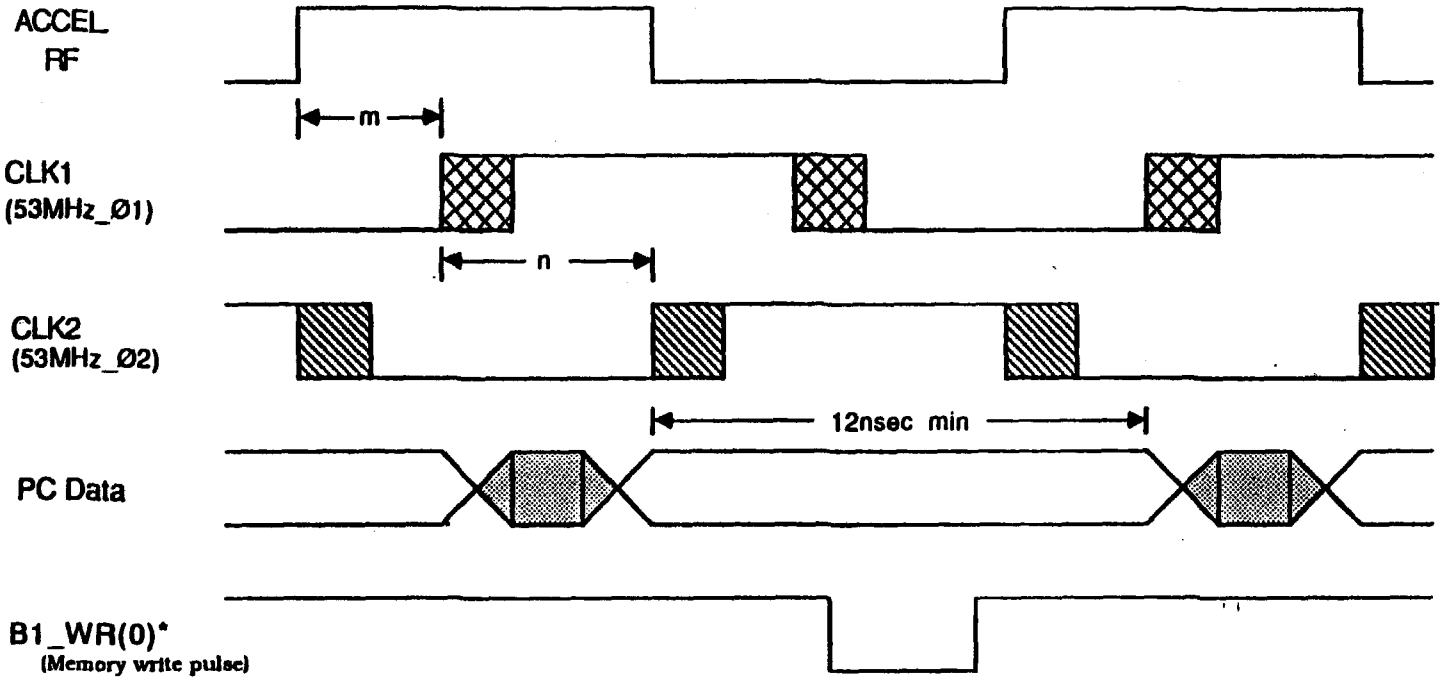
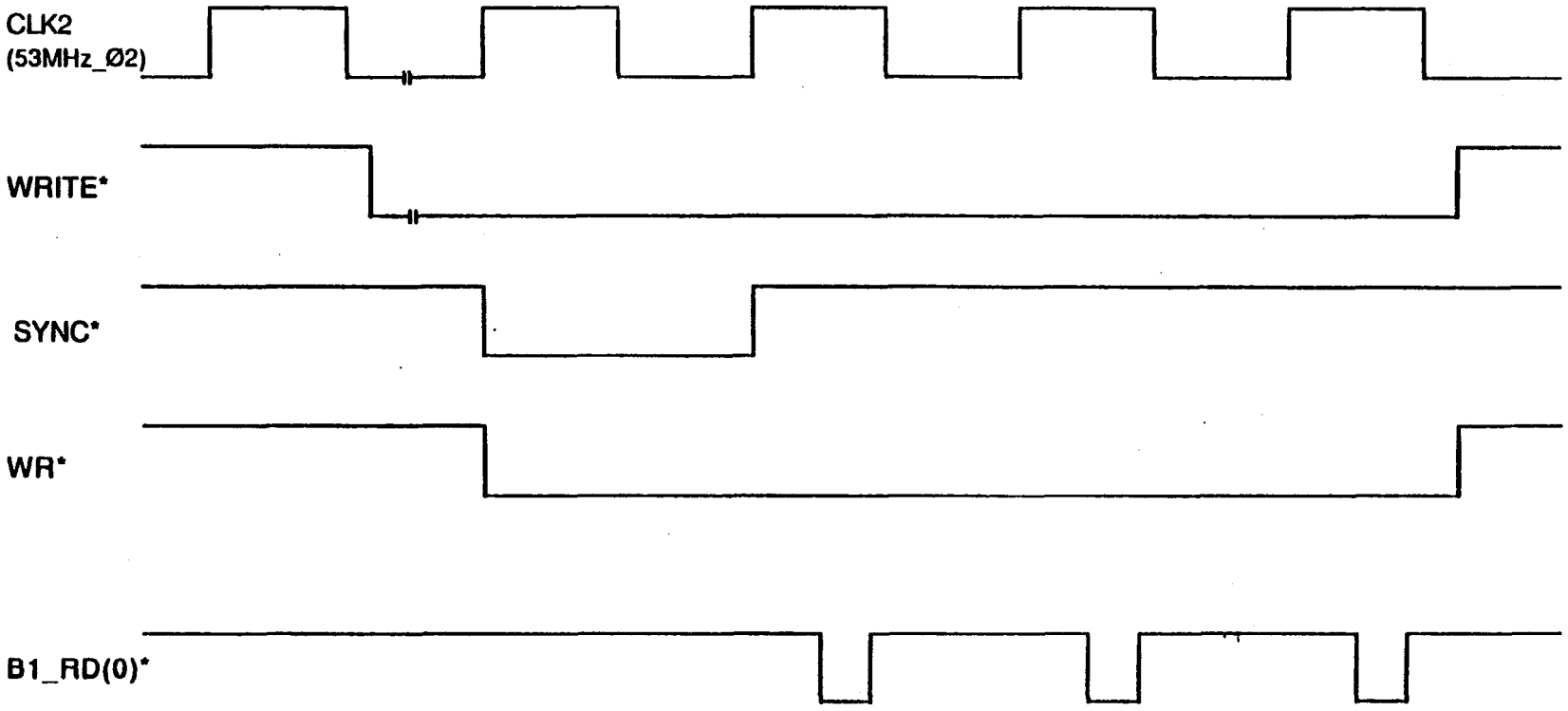


Figure 5: Front Panel Layout



m - Delay added to synchronize the PC to the accelerator RF.
n - Delay added to CLK1 clock to synchronize the DE to the PC output data.

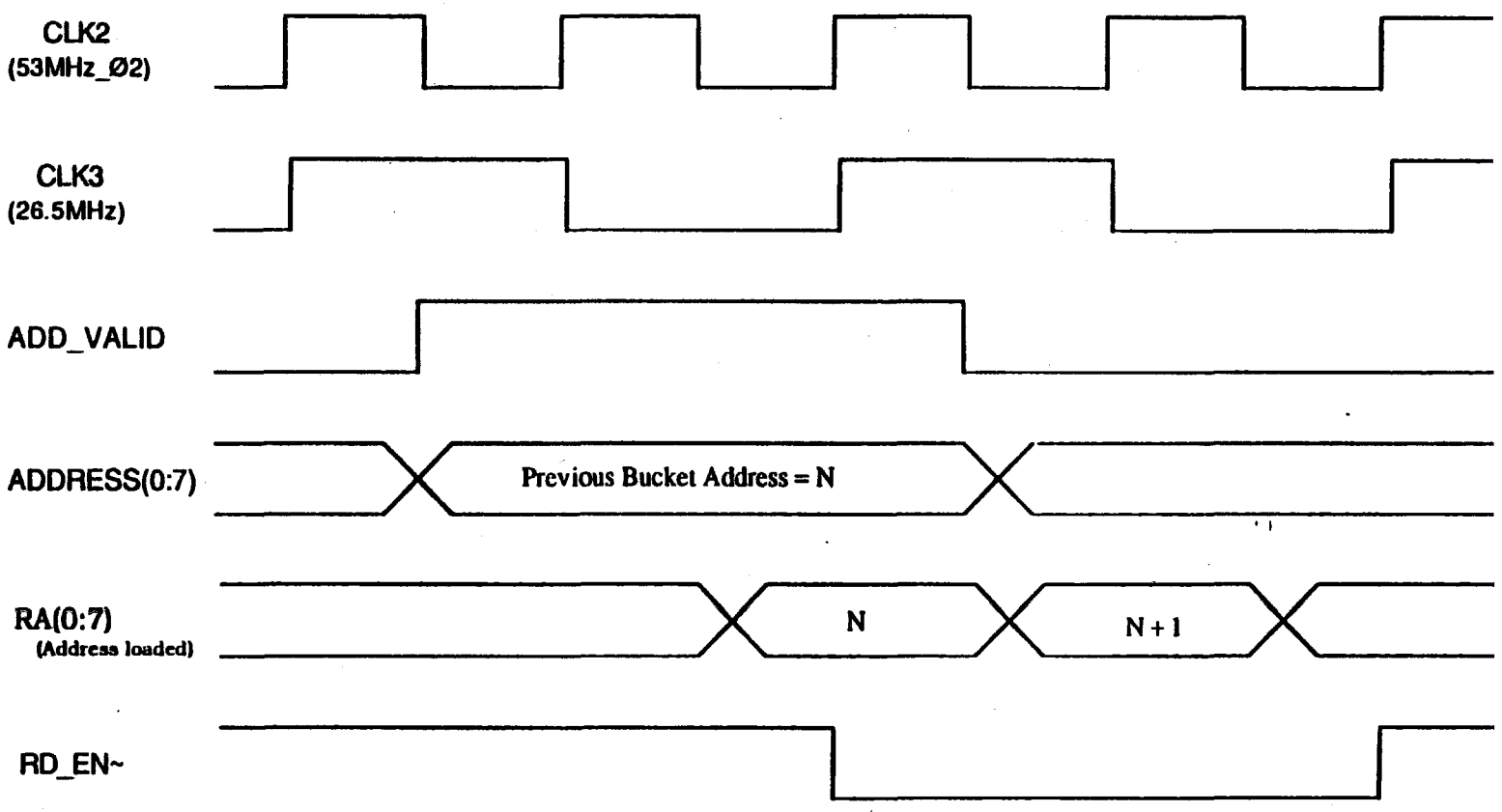
Timing 1: Synchronization of DE to PC.



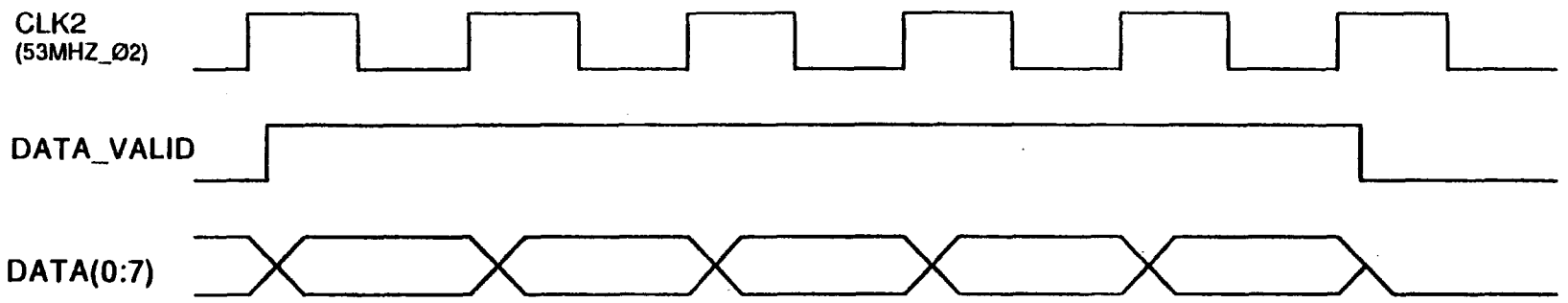
DESCRIPTION:

WRITE*	- Write enable signal on the auxiliary backplane driven by Sequencer.
SYNC*	- Synchronization signal generated by MTC and driven on the backplane by the SEQ.
WR*	- DE internally synchronized write enable signal.
B1_WR(0)*	- A typical write pulse for the memories.

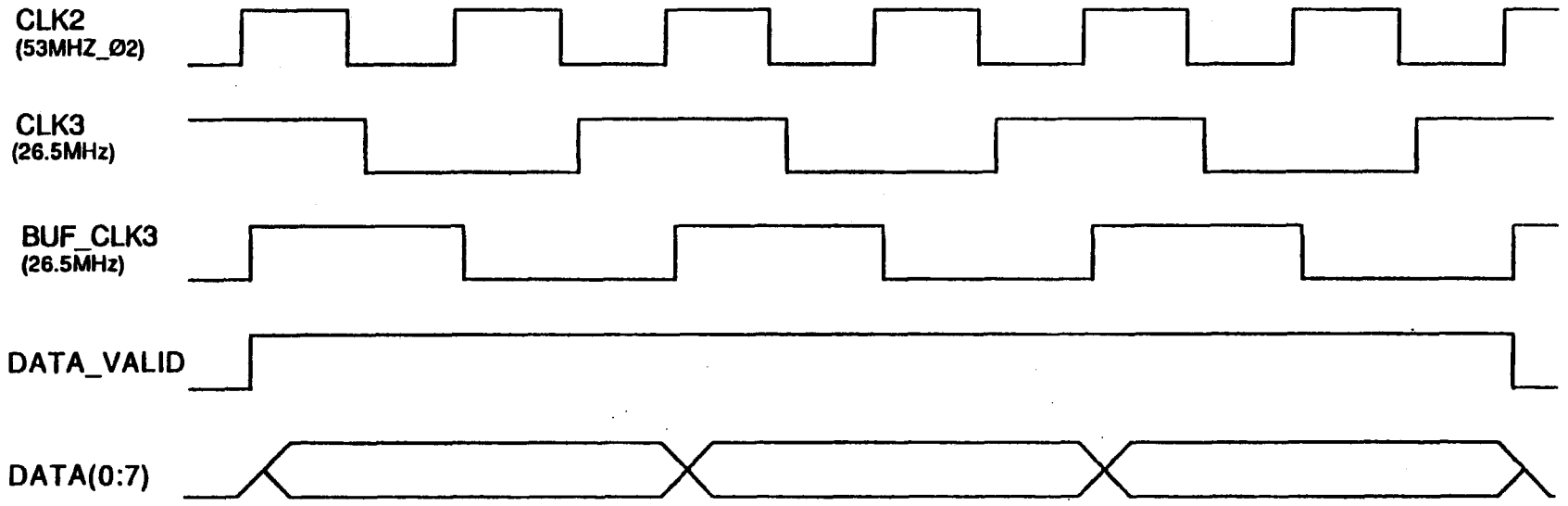
Timing 2: Write Enable Synchronization.



Timing 3: Address Transfer from SEQ to DE.



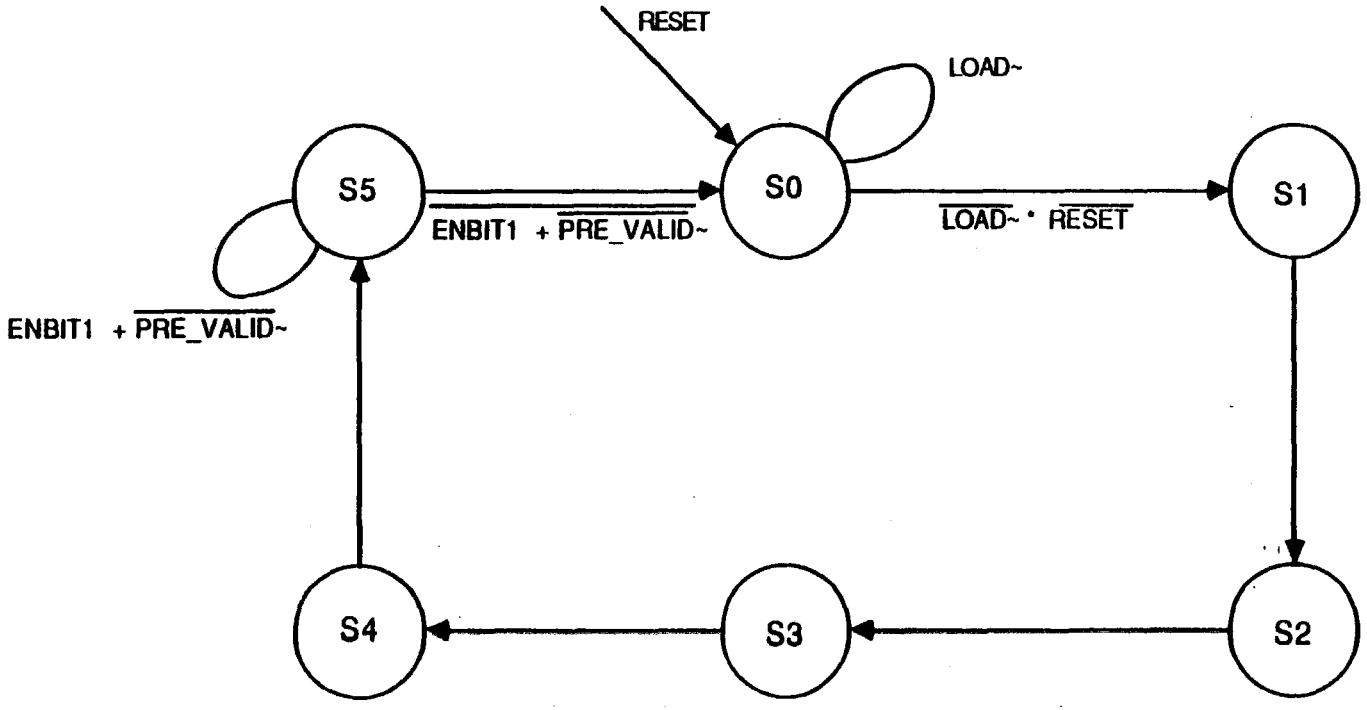
Timing 4a : 53MHz Encoded Event Transfer



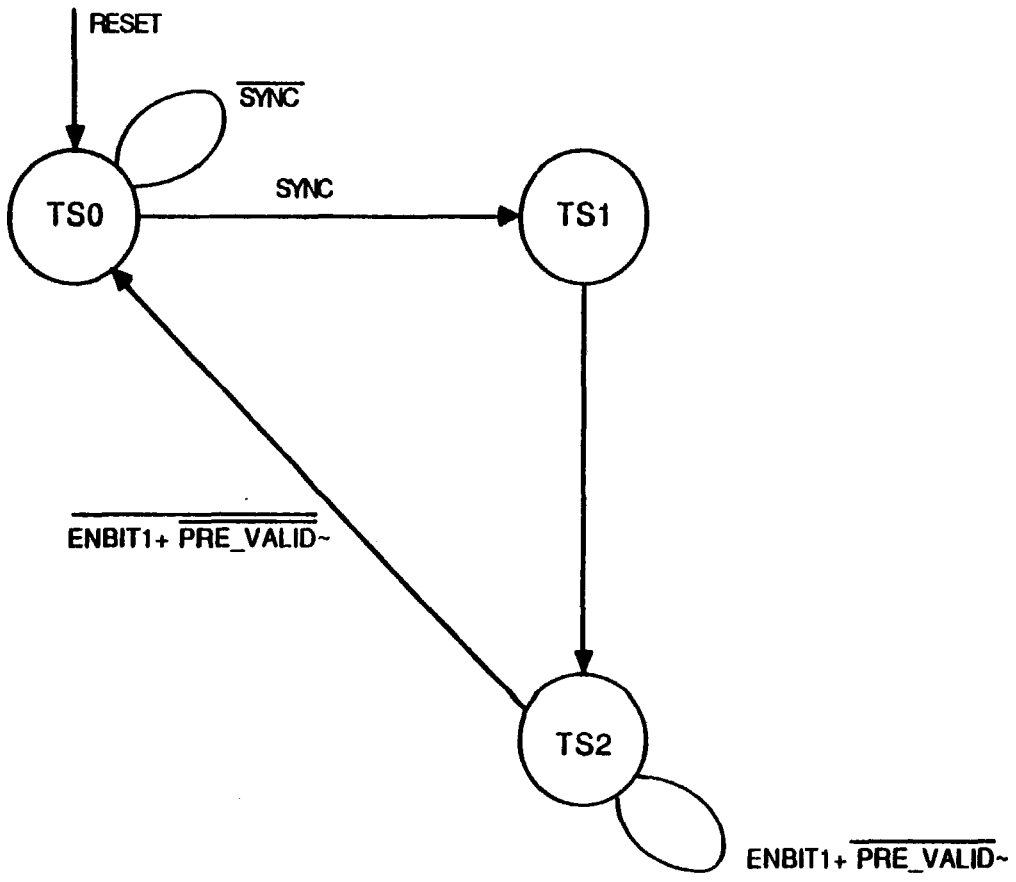
Timing 4b: 26.5MHz Encoded Event Transfer

10. Appendix E

State Machine
and
Pal Equation Used in DE



ENCODER PRIMARY STATE MACHINE



ENCODER DATA TRANSFER STATE MACHINE

EQUATIONS:

S0 = RESET + S0 * LOAD~ + S5 * (NOT (ENBIT1 + NOT (PRE_VALID~))): ;

S1 = S0 * (NOT LOAD~) * (NOT RESET);

S2 = S1;

S3 = S2;

S4 = S3;

S5 = (NOT RESET) * (S4 + S5 * (ENBIT1 + NOT (PRE_VALID~)));

TS0 = RESET + TS0 * (NOT SYNC) + TS2 * (NOT (ENBIT1 + PRE_VALID~)));

TS1 = TS0 * SYNC * (NOT RESET);

TS2 = (NOT RESET) * (TS1 + TS2 * (ENBIT1 + (NOT PRE_VALID~)));

```
module DE_CONTROL

title    'State Control for the Delay/Encoder

            Revisions:

                    Hector L. Gonzalez
                    January 29, 1990'

        DE_CNTRL            DEVICE          'EC20P8M';

"inputs

        S0, S3, S4, S5              PIN     1, 2, 3, 11;
        RESET                       PIN     9;
        SYNC                        PIN     10;
        !LOAD                       PIN     13;
        !EN_BIT1                    PIN     14;
        !PRE_VALID                  PIN     15;
        TS0, TS1, TS2               PIN     16, 22, 23;

"outputs

        PS0, PS1, PS4, PS5          PIN     [illegible]
        PTS0, PTS1, PTS2            PIN     [illegible]


equations

        PS0  =  RESET
              # S0 & !LOAD & !RESET
              # S5 & !(EN_BIT1 # PRE_VALID) & !RESET;

        PS1 =   S0 & LOAD & !RESET;

        PS4 =   S3 & !RESET;

        PS5 =   S4 & !RESET
              # S5 & (EN_BIT1 # PRE_VALID) & !RESET;

        PTS0 = RESET
              # TS0 & !SYNC & !RESET
              # TS2 & !(EN_BIT1 # PRE_VALID) & !RESET;

        PTS1 = TS0 & SYNC & !RESET;

        PTS2 = TS1 & !RESET
              # TS2 & (EN_BIT1 # PRE_VALID ) & !RESET;

end  DE_CONTROL
```

11. Appendix F

Parts List

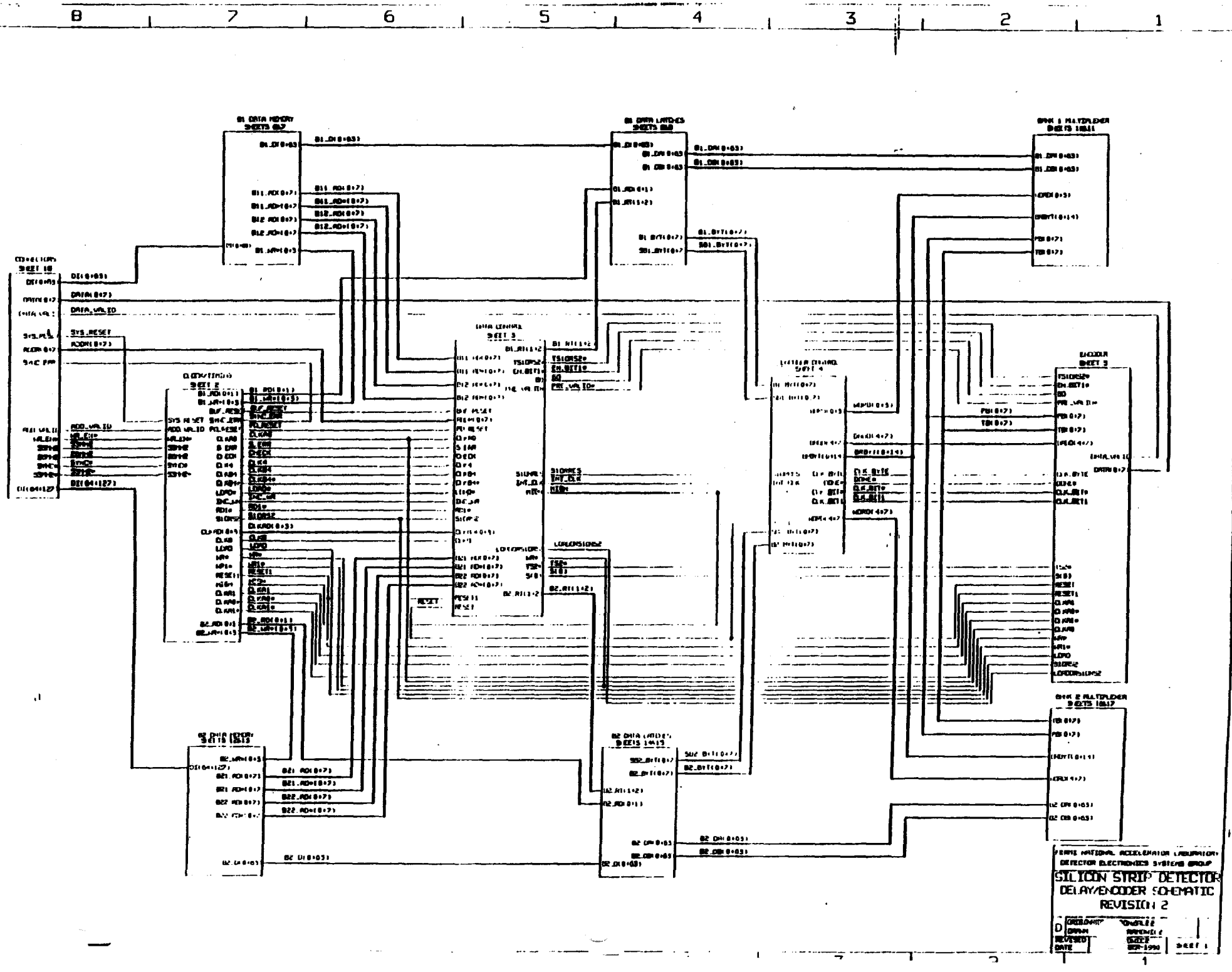
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Part Number	Quantity/	IEE(mA)	IEE(mA)	IEE(typ)	IEE(max	Price /	Cost/	Parts	Spares	Total	Parts	Qty. to	Qty.	Qty.	Total Cost /
2		Board	Typical	Maximum	Total	Total	1000 Qty	Board	Needed	Needed	Needed	on hand	Order	Ordered	Rec'd.	24
3																
4	MC10E016FN	2	151	181	0.30	0.36	$28.20	$56.40	48	5	53	49	4	0	0	$1,494.60
5	MC10H101P	7	20	26	0.14	0.18	$0.60	$4.20	168	17	185	161	24	0	0	$111.00
6	MC10H102P	6	20	26	0.12	0.16	$0.60	$3.60	144	14	158	374	-216	0	0	$94.80
7	MC10H103P	6	21	26	0.13	0.16	$0.60	$3.60	144	14	158	272	-114	0	0	$94.80
8	MC10H104P	2	35	35	0.07	0.07	$0.60	$1.20	48	5	53	109	-56	0	0	$31.80
9	MC10H107P	4	28	28	0.11	0.11	$0.65	$2.60	96	10	106	217	-111	0	0	$68.90
10	MC10H109P	18	11	14	0.20	0.25	$0.76	$13.68	432	43	475	793	-318	0	0	$361.00
11	MC10E111FN	1	48	60	0.05	0.06	$23.47	$23.47	24	2	26	41	-15	0	0	$610.22
12	MC10E112FN	1	47	56	0.05	0.06	$9.00	$9.00	24	2	26	36	-10	0	0	$234.00
13	MC10H131P	12	45	56	0.54	0.67	$1.50	$18.00	288	29	317	600	-283	0	0	$475.50
14	MC10H131FN	16	45	56	0.72	0.90	$1.90	$30.40	384	38	422	385	37	0	0	$801.80
15	MC10E131FN	3	58	70	0.17	0.21	$12.60	$37.80	72	7	79	131	-52	0	0	$995.40
16	MC10E151FN	6	65	85	0.39	0.51	$13.76	$82.56	144	14	158	162	-4	0	0	$2,174.08
17	MC10H158P	3	38	48	0.11	0.14	$1.35	$4.05	72	7	79	106	-27	0	0	$106.65
18	MC10H159FN	32	42	53	1.34	1.70	$2.03	$64.96	768	77	845	849	-4	0	0	$1,715.35
19	MC10H162P	3	61	76	0.18	0.23	$1.50	$4.50	72	7	79	82	-3	0	0	$118.50
20	MC10H164P	1	60	75	0.06	0.08	$1.45	$1.45	24	2	26	31	-5	0	0	$37.70
21	MC10H165P	3	105	131	0.32	0.39	$5.10	$15.30	72	7	79	87	-8	0	0	$402.90
22	MC10H176P	46	88	112	4.05	5.15	$2.36	$108.56	1104	110	1214	1202	12	0	0	$2,865.04
23	MC10H186P	6	88	110	0.53	0.66	$3.25	$19.50	144	14	158	143	15	0	0	$513.50
24	MC10H188P	1	42	42	0.04	0.04	$1.00	$1.00	24	2	26	31	-5	0	0	$26.00
25	MC10198P	1	80	100	0.08	0.10	$13.00	$13.00	24	2	26	47	-21	0	0	$338.00
26	MBM10422-5	32	150	175	4.80	5.60	$8.33	$266.56	768	77	845	9000	-8155	0	0	$7,038.85
27	TIE10H16P8-6	1	210	210	0.21	0.21	$30.41	$30.41	24	2	26	35	-9	0	0	$790.66
28	ICO-324-SGG SOC.	1	0	0	0.00	0.00	$1.65	$1.65	24	2	26	100	-74	0	0	$42.90
29	FDD3510	1	0	0	0.00	0.00	$8.00	$8.00	24	2	26	0	26	0	0	$208.00
30	FDD4010	1	0	0	0.00	0.00	$8.00	$8.00	24	2	26	100	-74	0	0	$208.00
31	FDA6010	3	0	0	0.00	0.00	$8.00	$24.00	72	7	79	78	1	0	0	$632.00
32	FDA7010	1	0	0	0.00	0.00	$8.00	$8.00	24	2	26	19	7	0	0	$208.00
33	4309R-101-470	8	0	0	0.00	0.00	$0.15	$1.20	192	19	211	69	142	0	0	$31.65
34	4308R-101-101	72	0	0	0.00	0.00	$0.15	$10.80	1728	173	1901	3000	-1099	0	0	$285.15
35	4309R-101-101	59	0	0	0.00	0.00	$0.20	$11.80	1416	142	1558	2500	-942	0	0	$311.60
36	2-532956-0	1	0	0	0.00	0.00	$9.50	$9.50	24	2	26	0	26	0	0	$247.00
37	534974-9	1	0	0	0.00	0.00	$26.00	$26.00	24	2	26	0	26	0	0	$676.00
38	50 OHM RES.	16	0	0	0.00	0.00	$0.04	$0.64	384	38	422	0	422	0	0	$16.88
39	100 OHM RES.	4	0	0	0.00	0.00	$0.04	$0.16	96	10	106	0	106	0	0	$4.24

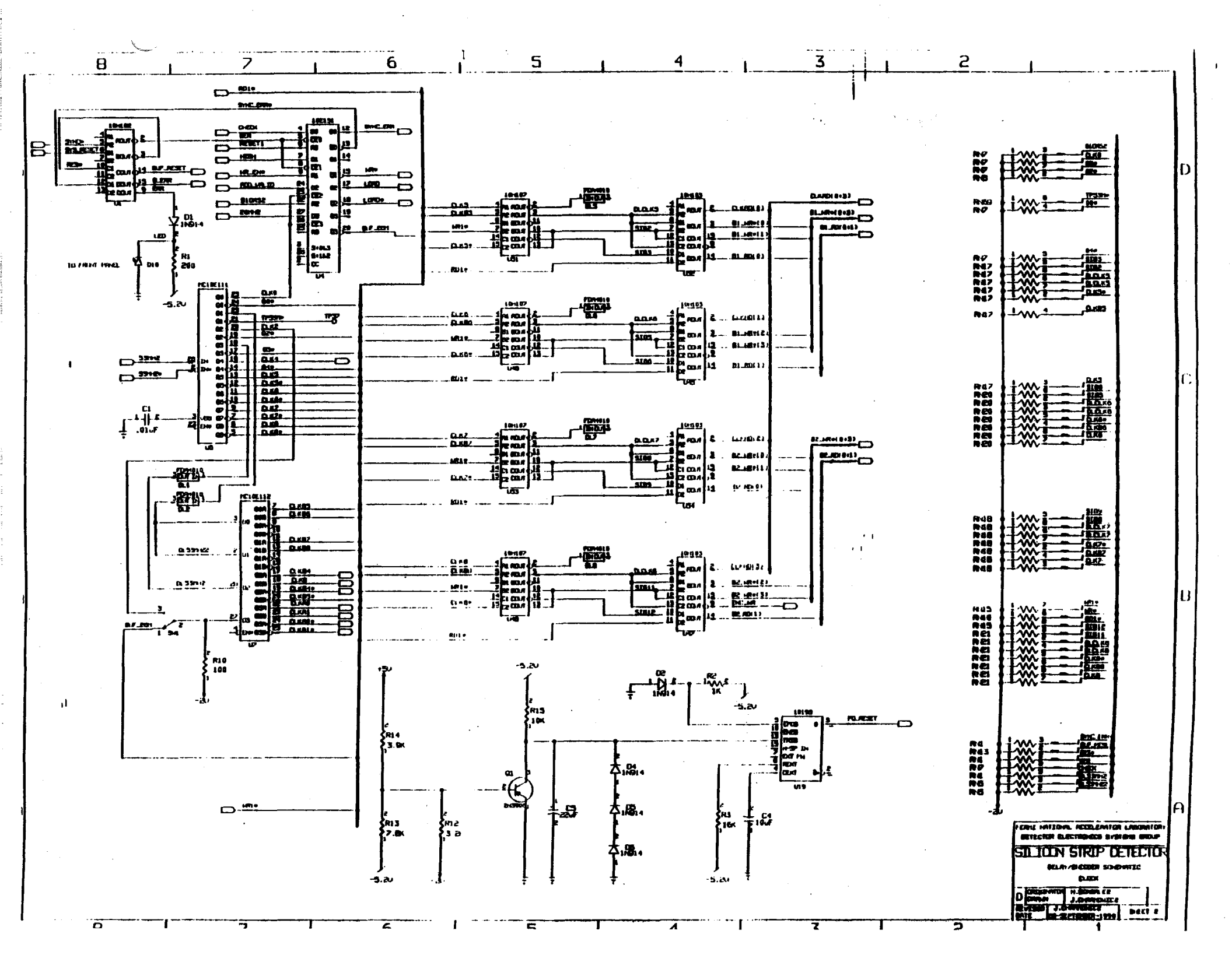
DELAY 24

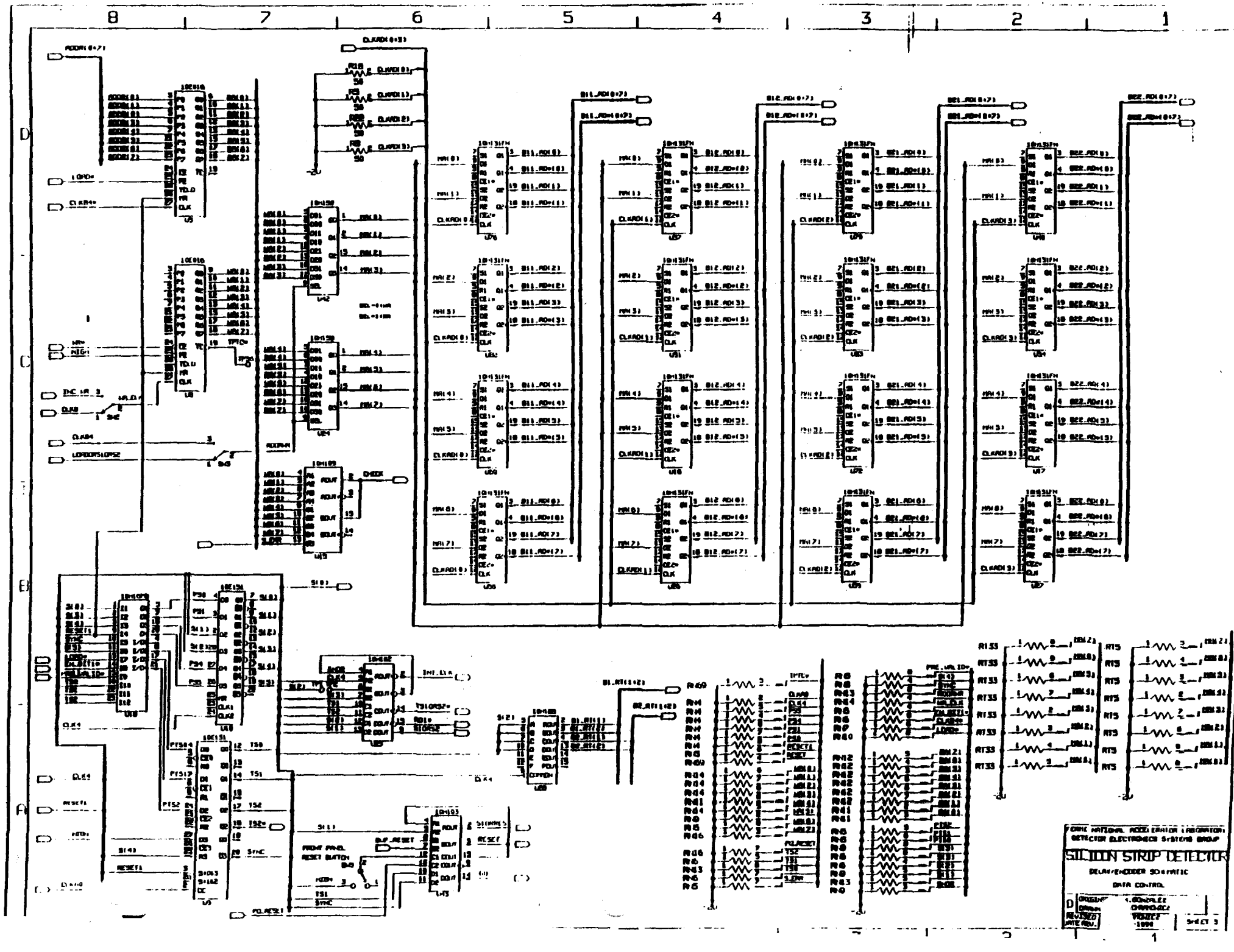
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Part Number	Quantity/	IEE(mA)	IEE(mA)	IEE(typ)	IEE(max)	Price /	Cost/	Parts	Spares	Total	Parts	Qty. to	Qty.	Qty.	Total Cost /
2		Board	Typical	Maximum	Total	Total	1000 Qty	Board	Needed	Needed	Needed	on hand	Order	Ordered	Rec'd.	24
3																
40	200 OHM RES.	1	0	0	0.00	0.00	$0.04	$0.04	24	2	26	0	26	0	0	$1.04
41	1K OHM RES.	3	0	0	0.00	0.00	$0.04	$0.12	72	7	79	0	79	0	0	$3.16
42	3.2K OHM RES.	1	0	0	0.00	0.00	$0.04	$0.04	24	2	26	0	26	0	0	$1.04
43	3.9K OHM RES.	1	0	0	0.00	0.00	$0.04	$0.04	24	2	26	0	26	0	0	$1.04
44	7.8K OHM RES.	1	0	0	0.00	0.00	$0.04	$0.04	24	2	26	0	26	0	0	$1.04
45	10K OHM RES.	1	0	0	0.00	0.00	$0.04	$0.04	24	2	26	0	26	0	0	$1.04
46	16K OHM RES.	1	0	0	0.00	0.00	$0.04	$0.04	24	2	26	0	26	0	0	$1.04
47	20 pF CAP	4	0	0	0.00	0.00	$0.20	$0.80	96	10	106	0	106	0	0	$21.20
48	KEMET 4.7 uF CAP	10	0	0	0.00	0.00	$0.14	$1.40	240	24	264	245	19	0	0	$36.96
49	10uF CAP.	1	0	0	0.00	0.00	$0.75	$0.75	24	2	26	0	26	0	0	$19.50
50	22UF CAP.	1	0	0	0.00	0.00	$0.75	$0.75	24	2	26	0	26	0	0	$19.50
51	47uF CAP.	2	0	0	0.00	0.00	$0.75	$1.50	48	5	53	0	53	0	0	$39.75
52	1N914 DIODE	7	0	0	0.00	0.00	$0.15	$1.05	168	17	185	0	185	0	0	$27.75
53	2N3906 TRANS.	1	0	0	0.00	0.00	$0.20	$0.20	24	2	26	36	-10	0	0	$5.20
54	ICTE-5 DIODE	2	0	0	0.00	0.00	$0.75	$1.50	48	5	53	0	53	0	0	$39.75
55	TP12H9AB SW.	1	0	0	0.00	0.00	$4.53	$4.53	24	2	26	31	-5	0	0	$117.78
56	275005 FUSE	7	0	0	0.00	0.00	$0.75	$5.25	168	17	185	0	185	0	0	$138.75
57	2-331272-2 SOC.	16	0	0	0.00	0.00	$0.23	$3.68	384	38	422	0	422	0	0	$97.06
58	40F6261 TEST PT.	37	0	0	0.00	0.00	$0.37	$13.69	888	89	977	1200	-223	0	0	$361.49
59	1447-0475 RED LED	1	0	0	0.00	0.00	$0.20	$0.20	24	2	26	0	26	0	0	$5.20
60	923CZ5U103M050B	221	0	0	0.00	0.00	$0.20	$44.20	5304	530	5834	4800	1034	0	0	$1,166.80
61	CIRCUIT BOARD	1	0	0	0.00	0.00	$400.00	$400.00	24	0	24	0	24	0	0	$9,600.00
62	FRONT PANEL	1	0	0	0.00	0.00	$31.00	$31.00	24	0	24	0	24	0	0	$744.00
63	PNL. MTG. HDWR.	1	0	0	0.00	0.00	$1.00	$1.00	24	0	24	0	24	0	0	$24.00
64																
65	*IEE Typical =*	*14.71*	*Amps*	*IEE Max =*	*17.99*	*Amps*	*Cost/Bd.=*	*1,437.41*								
66	*Power Typ =*	*76*	*Watts*	*Power Max=*	*94*	*Watts*										
67																
68	*Last Updated:*	17-Oct		*Total # of*	*Boards*	24	*Ttl. Cost =*	*34,497.84*								

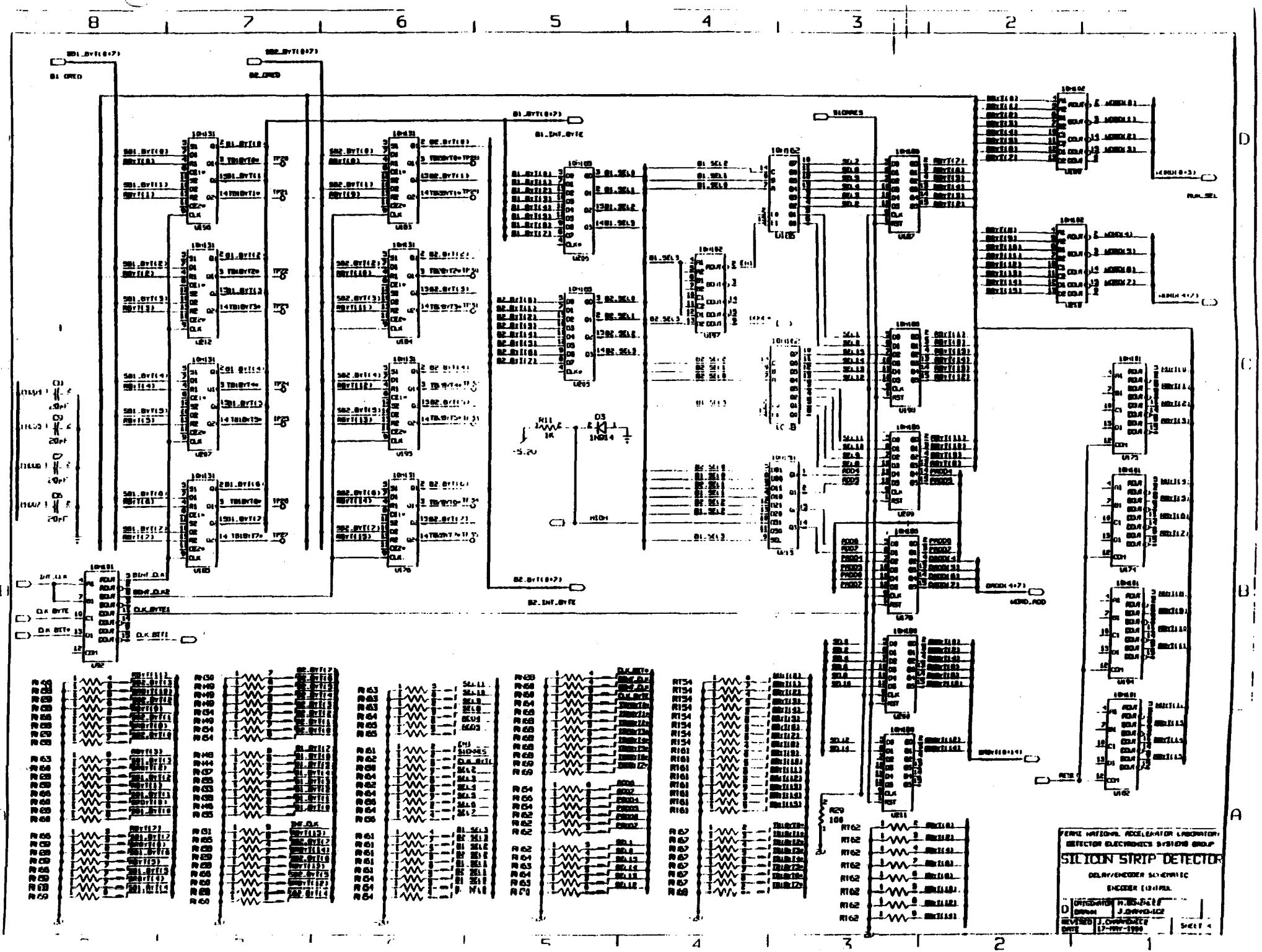
12. Appendix G

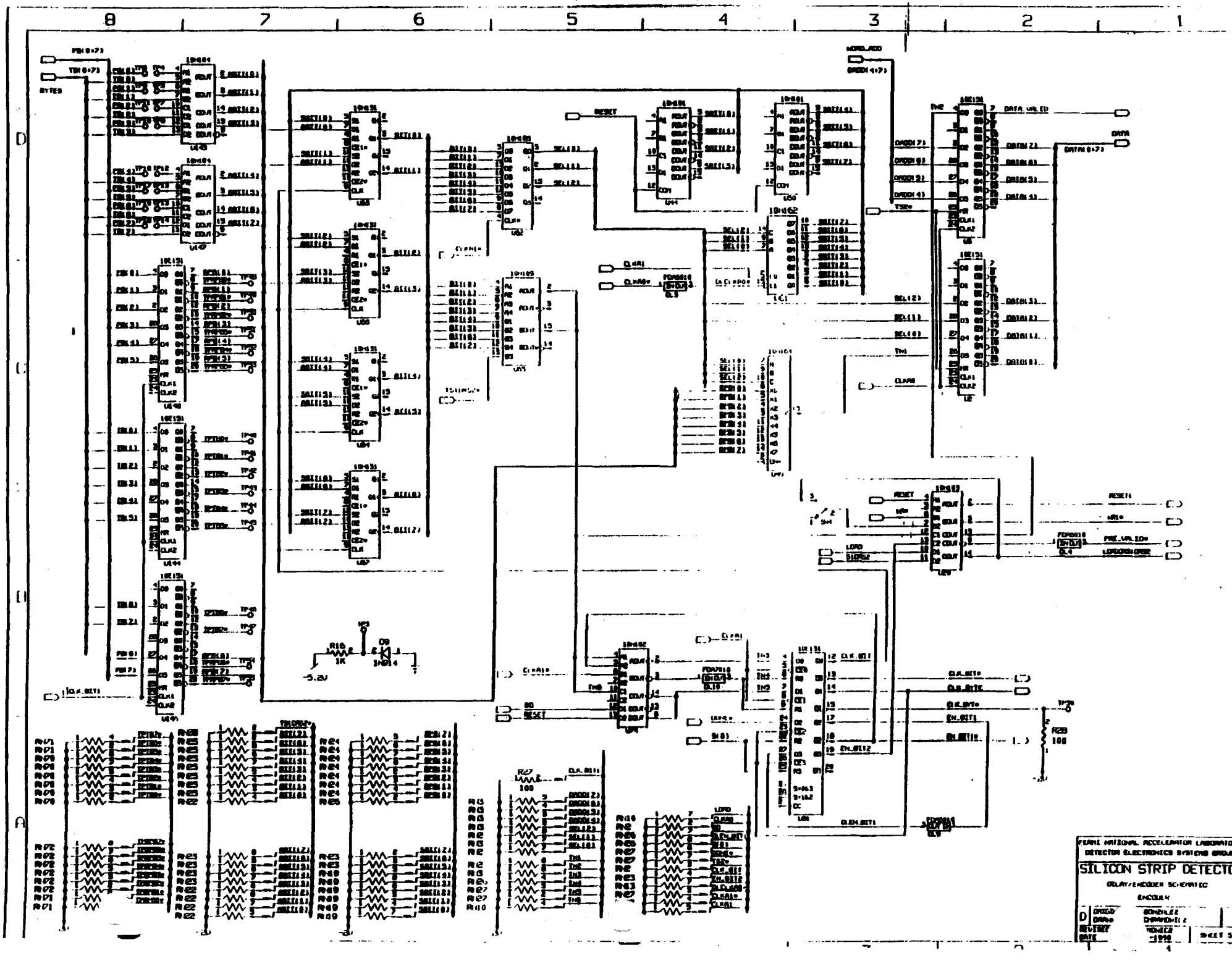
Circuit Diagrams

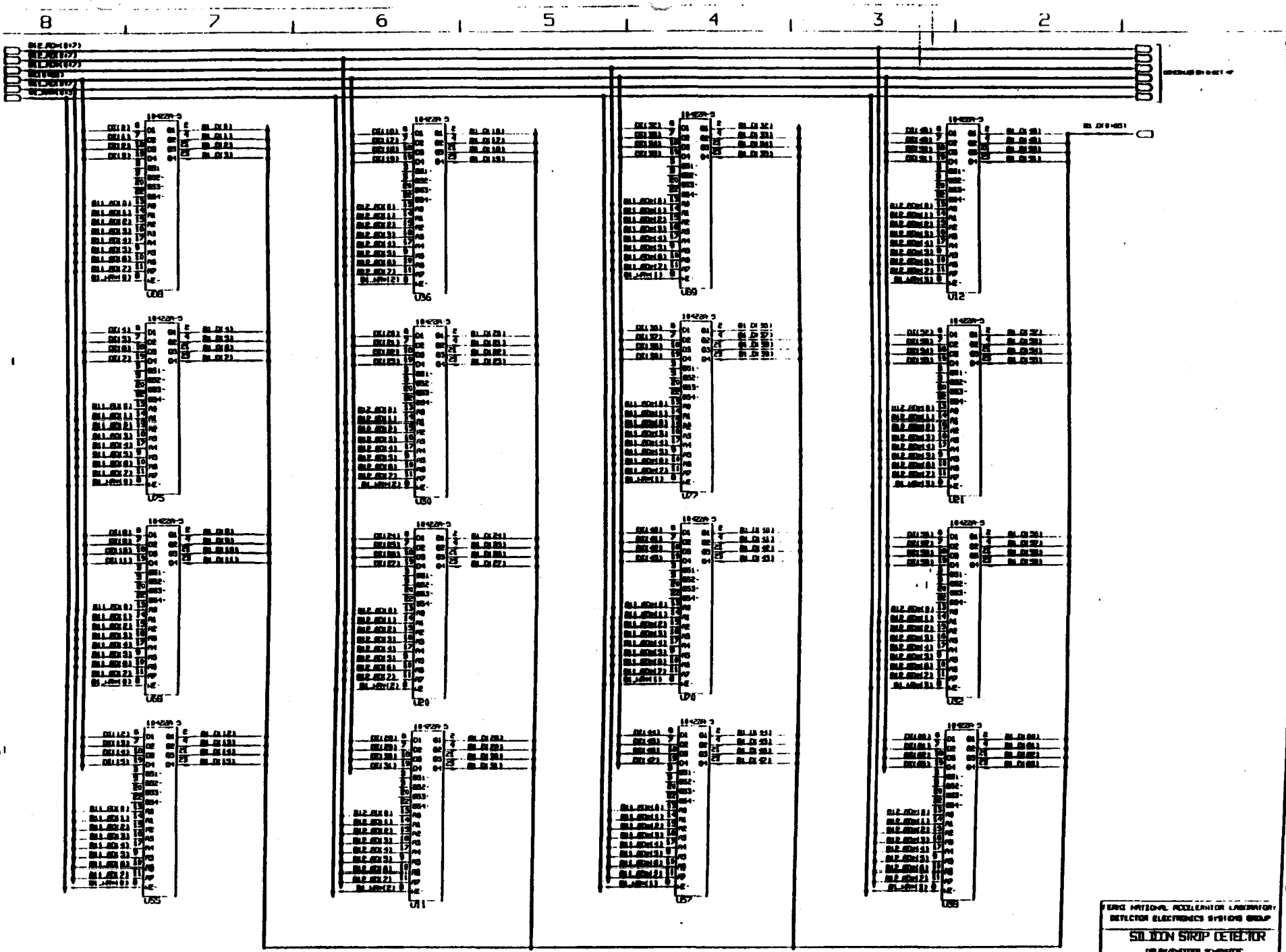

B1 DATA MEMORY
SHEETS 6&7
B1 DATA LATCHES
SHEETS 8&9
BANK 1 MULTIPLEXER
SHEETS 10&11
CONNECTOR
SHEET 10
DELAYS
SHEET 2
DELAY LOGIC
SHEET 3
LATENCY DELAY
SHEET 4
ENCODER
SHEET 5
B2 DATA MEMORY
SHEETS 12&13
B2 DATA LATCHES
SHEETS 14&15
BANK 2 MULTIPLEXER
SHEETS 16&17
SYS_RESET
ADDR[8:0]
DATA_VALID
DI[64:127]
RESET
LOADPULSE
LOADPULSE2
SILICON STRIP DETECTOR
DELAY/ENCODER SCHEMATIC
REVISION: 2
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP
DRAWN
REVISED
DATE
SHEET 1


SILICON STRIP DETECTOR


8
7
6
5
4
3
2
1
D
C
B
A
SILICON STRIP DETECTOR
DELAY/ENCODER SCHEMATIC
DATA CONTROL
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP
SHEET 3


FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP
SILICON STRIP DETECTOR
DELAY/ENCODER SCHEMATIC
ENCODER (DELPHI)


8
7
6
5
4
3
2
1
D
C
B
A
RESET
BYTES
DATA
-5.2V
SILICON STRIP DETECTOR
DELAY/ENCODER SCHEMATIC
ENCODER



FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP

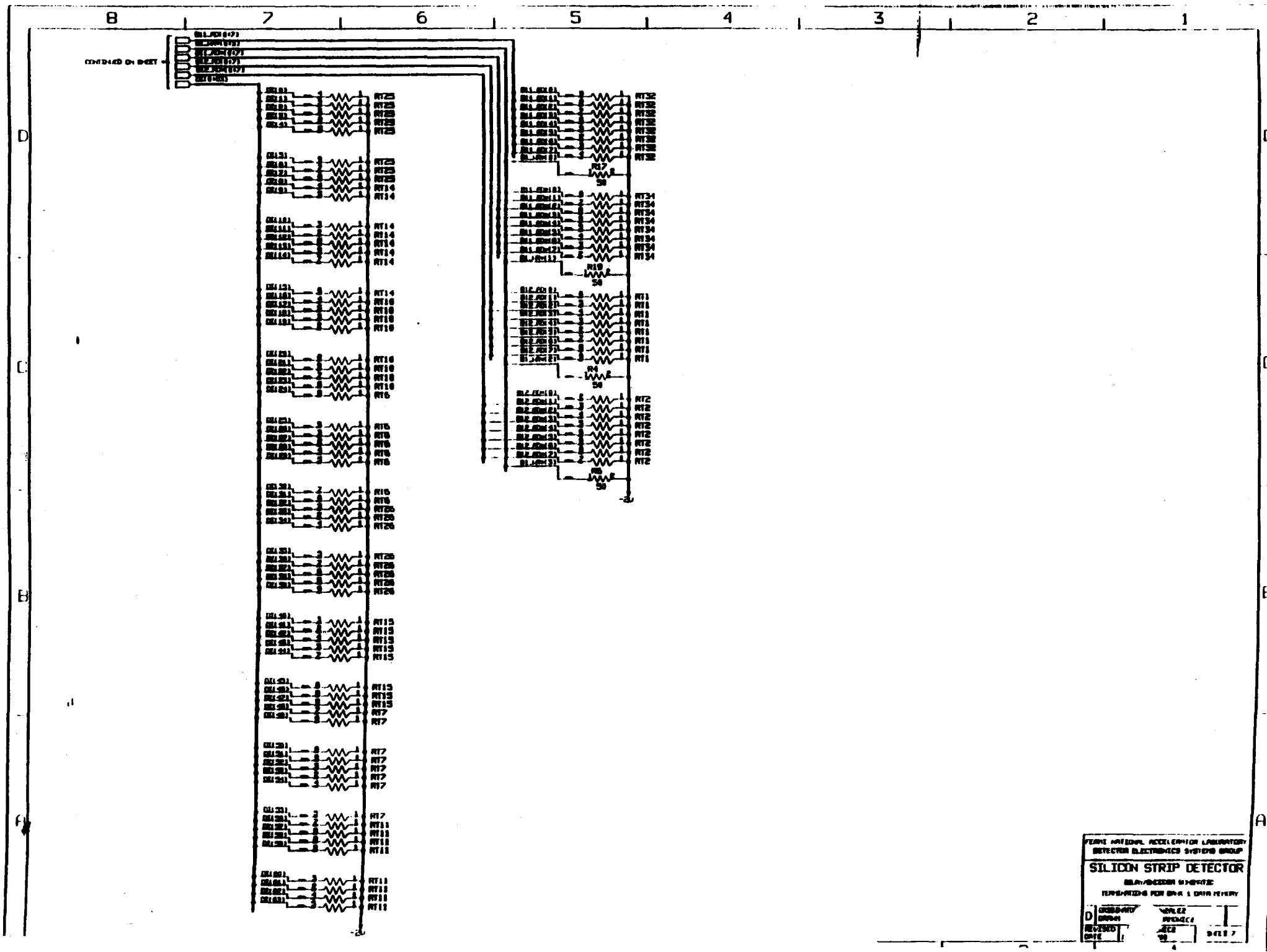
SILICON STRIP DETECTOR

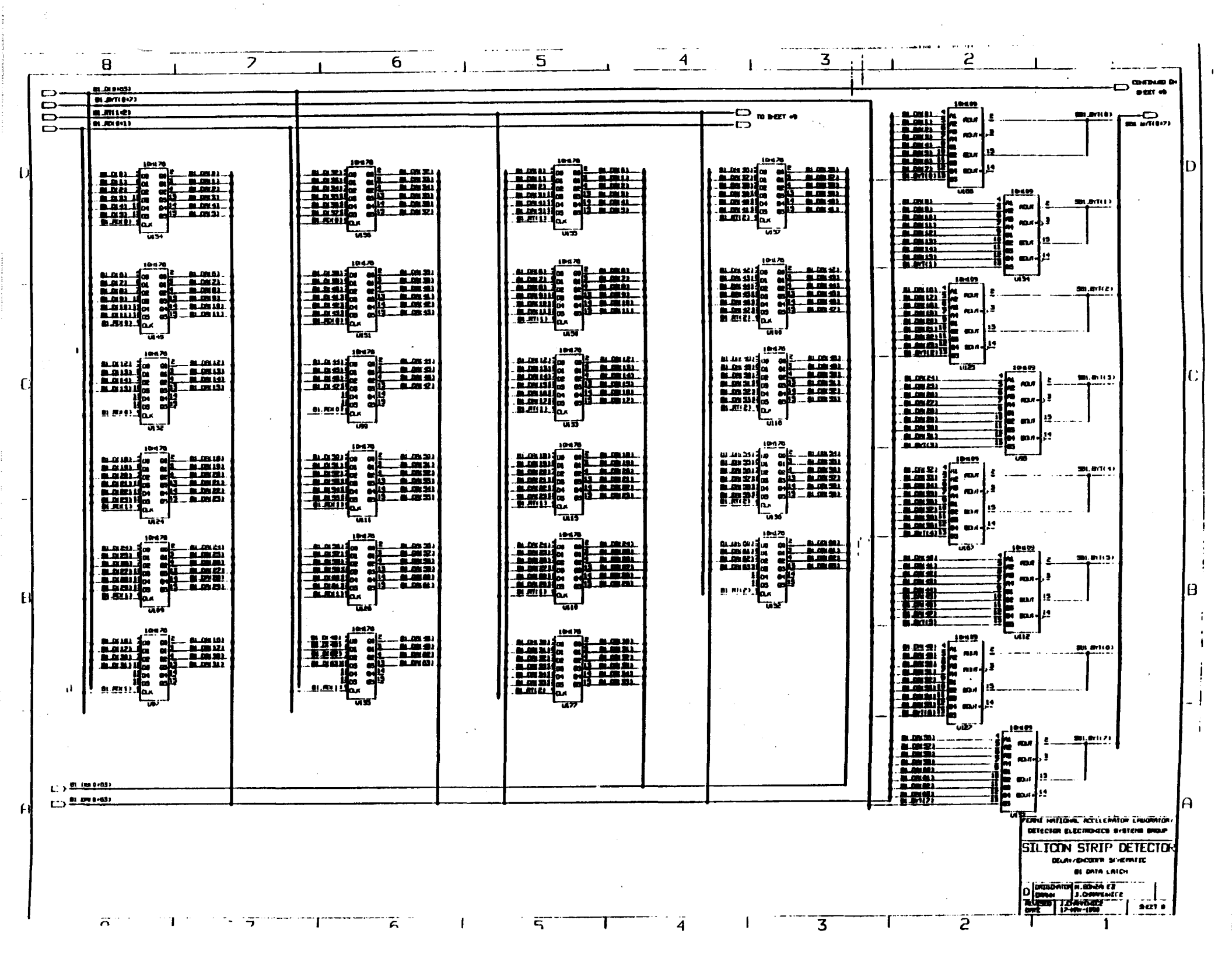
DELAY/DECODER SCHEMATIC

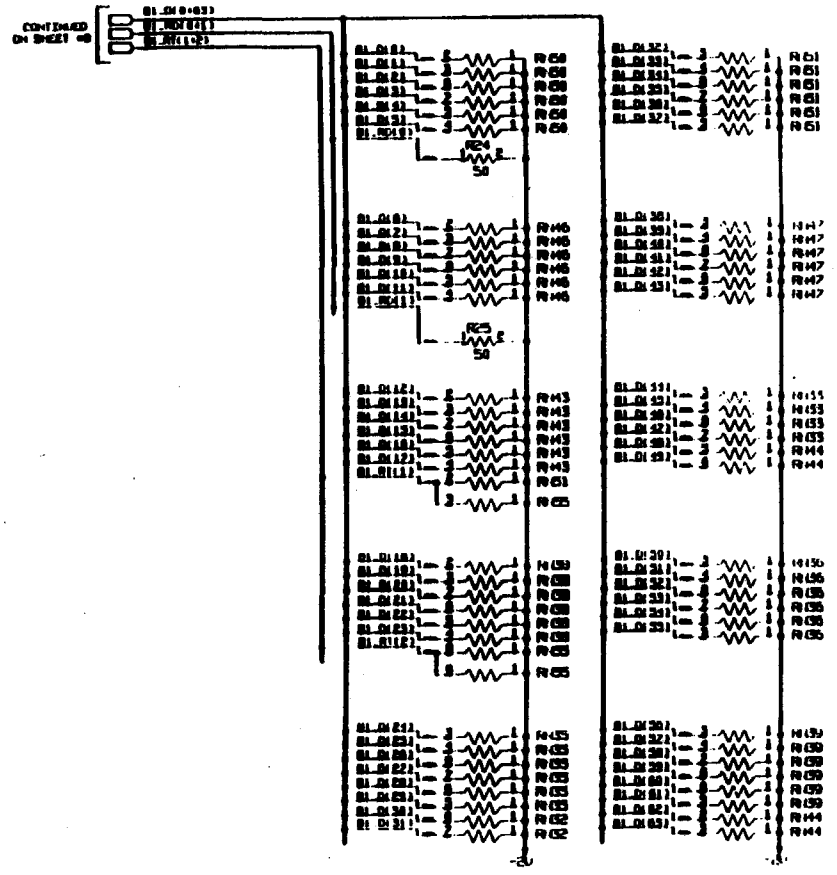
BANK 1 DATA MEMORY

D	DESIGNATOR	N. SCHMITT
	DRAWN	J. CHRISTIDIS
	REVISED	J. CHRISTIDIS
	DATE	17-NOV-1988

SHEET 6


8
7
6
5
4
3
2
1
D
C
B
A
CONTINUED ON SHEET
R17
50
R19
50
R4
50
R6
50
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP
SILICON STRIP DETECTOR
TERMINATIONS FOR SMX 1 DATA PENTRY
DRAWN
REVISED
DATE


FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP
SILICON STRIP DETECTOR
DELAY/DECODER SCHEMATIC
BI DATA LATCH



TERMINATIONS FOR BANK 1 DATA LATCHES
(SHEET 9)

FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP

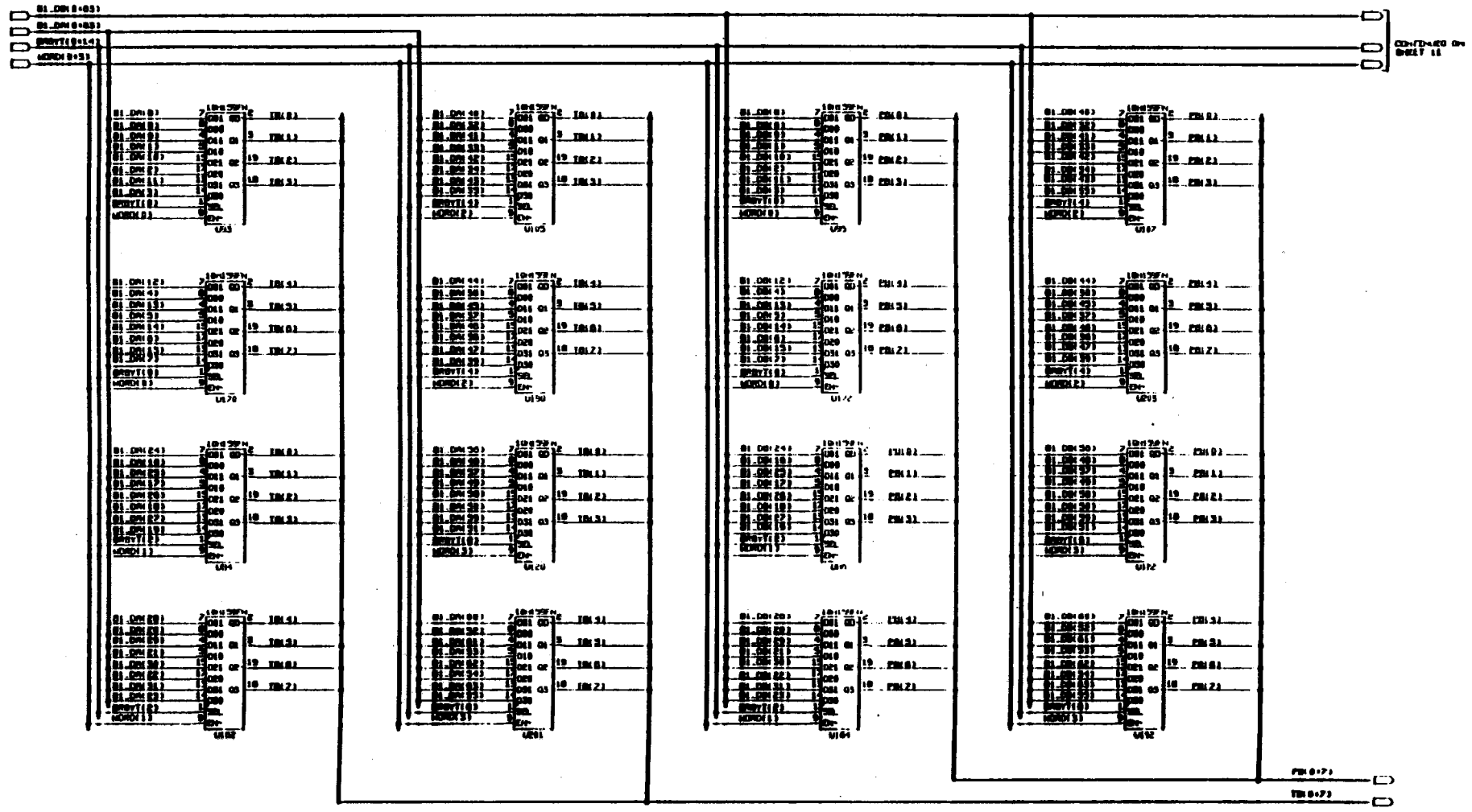
SILICON STRIP DETECTOR

DELAY/ENCODER SCHEMATIC

TERMINATIONS FOR BANK 1 DATA LATCHES

D	DESIGNED	M. GONZALEZ
	DRAWN	J. CHARGOIS
	REVISED	ROGERS
	DATE	4-1990

SHEET 9



FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP

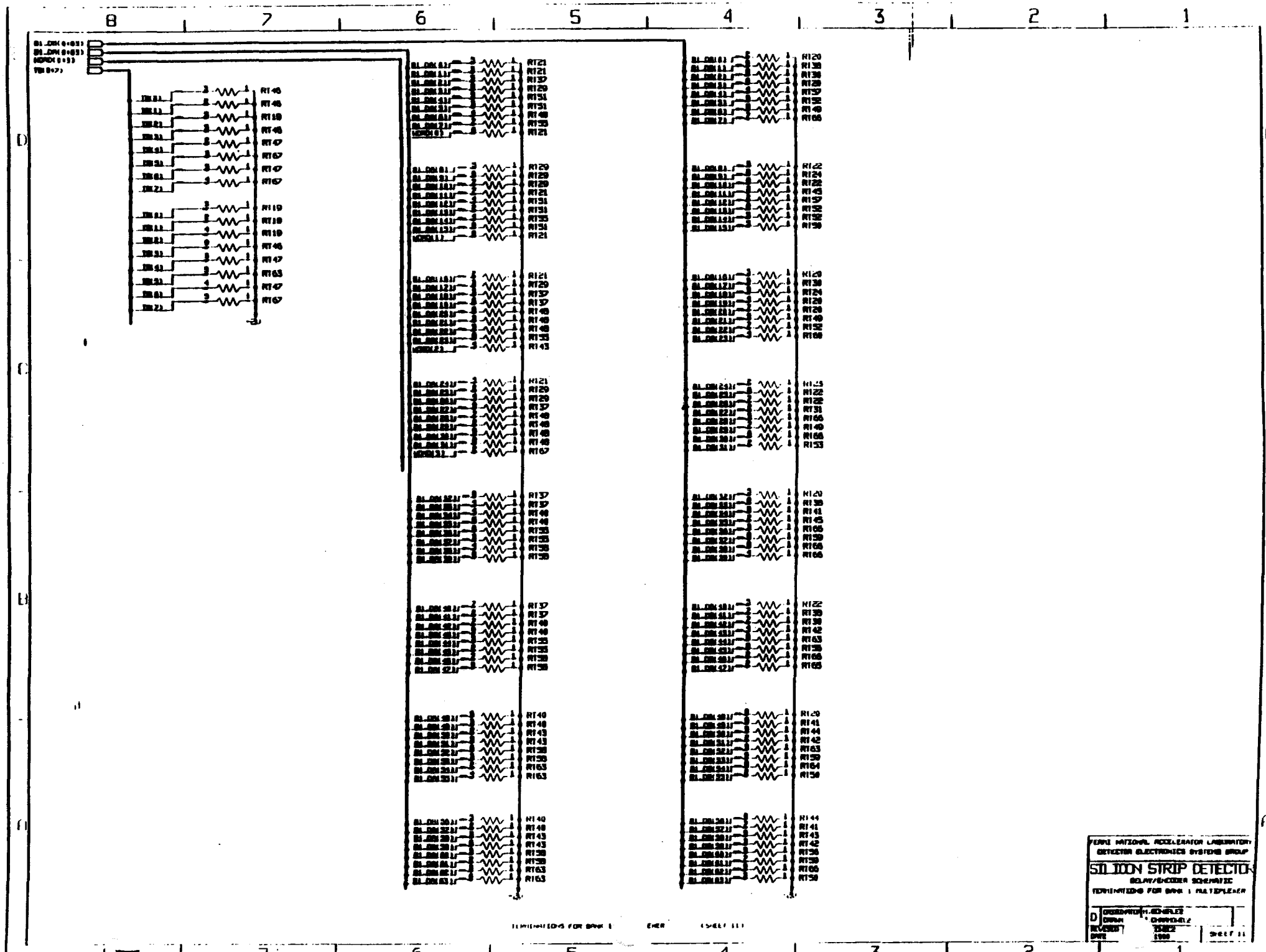
SILICON STRIP DETECTOR

DELAY/DECODER SCHEMATIC

DMM & MULTIPLEXER

D	DESIGNED	H.EDWARDS
	DRAWN	J.DAWKINS
	REVISED	J.DAWKINS
	DATE	12-FEB-1988

SHEET 10



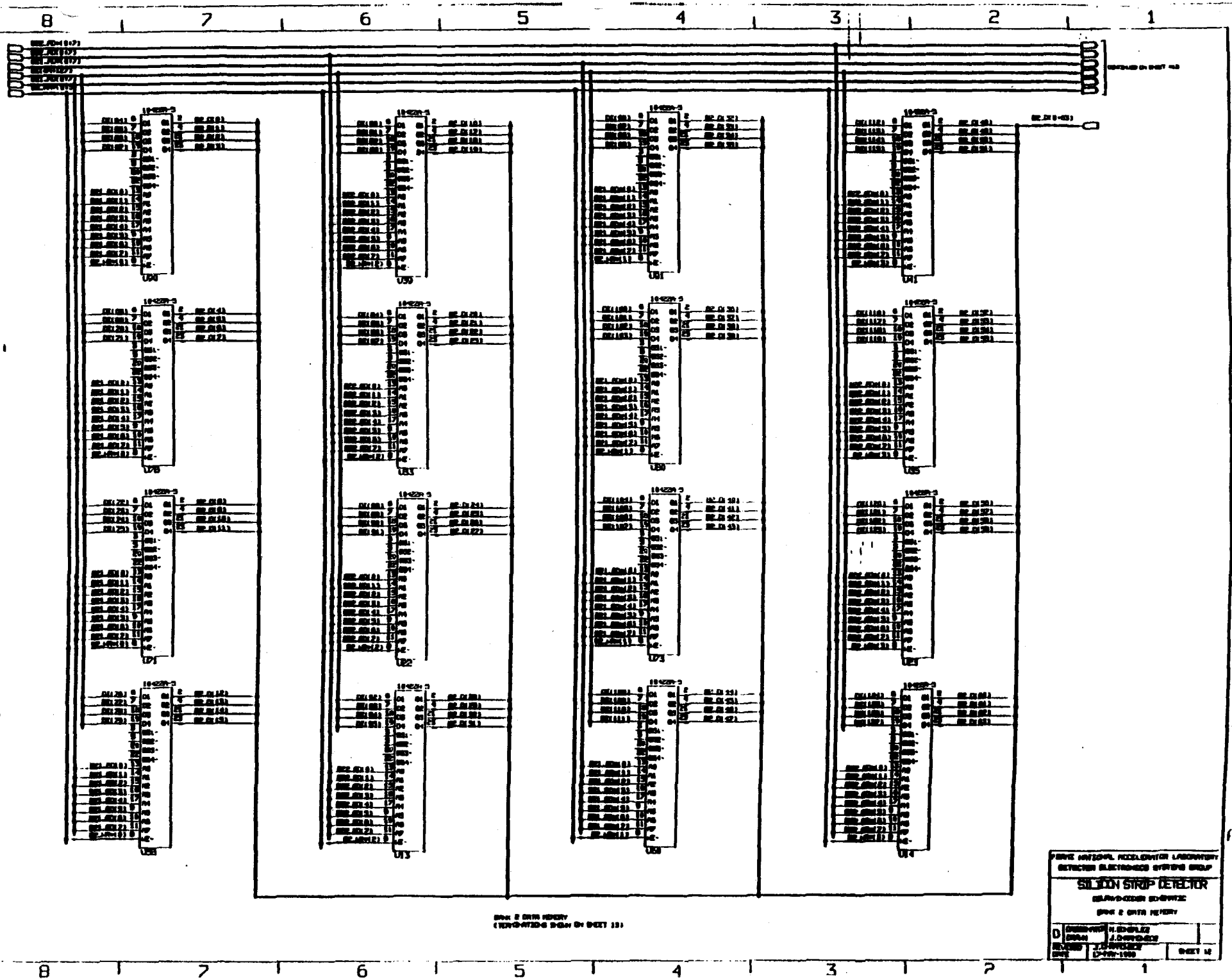
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP

SILICON STRIP DETECTOR

DELAY/ENCODER SCHEMATIC

TERMINATIONS FOR BANK 1 MULTIPLEXER

D



FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP

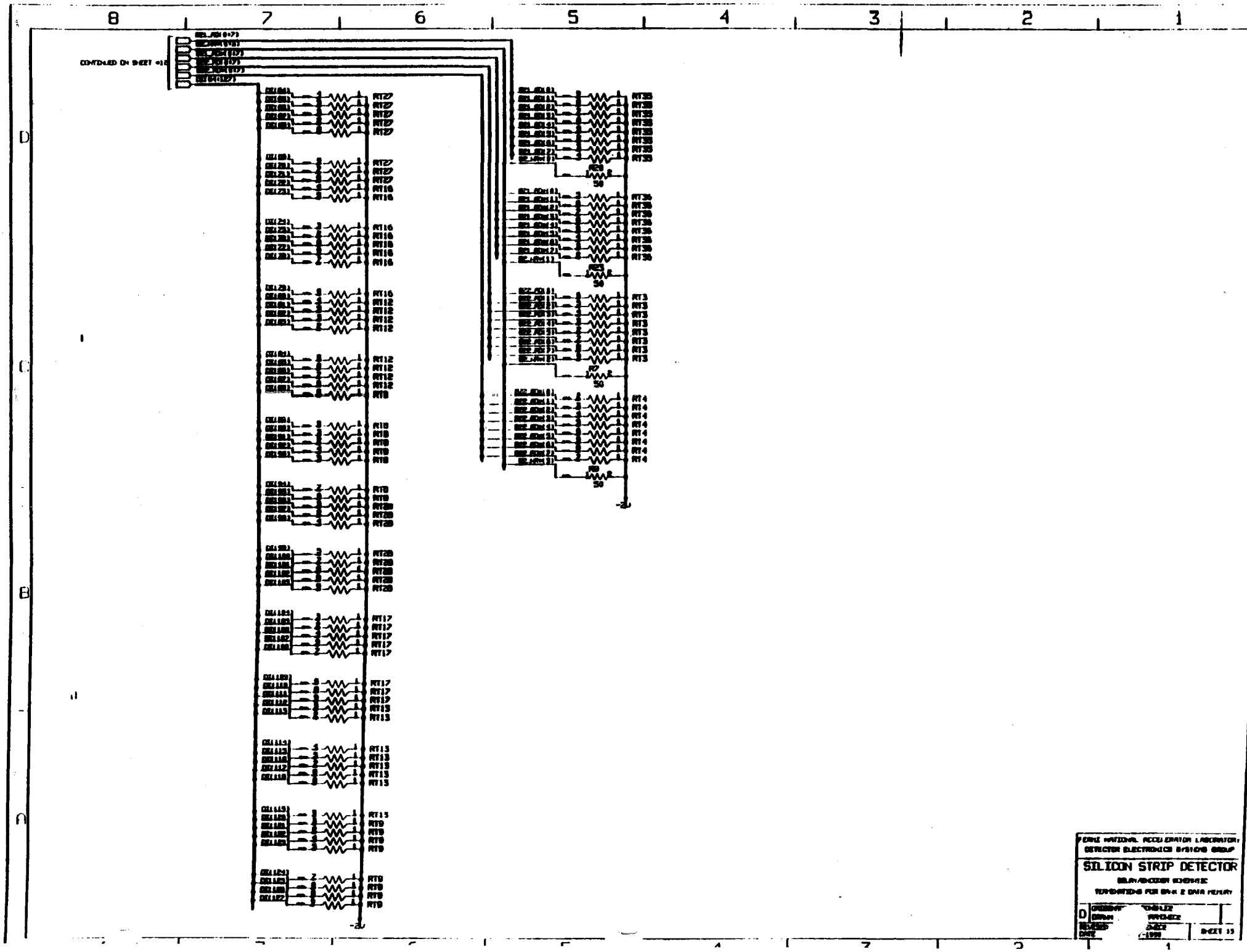
SILICON STRIP DETECTOR

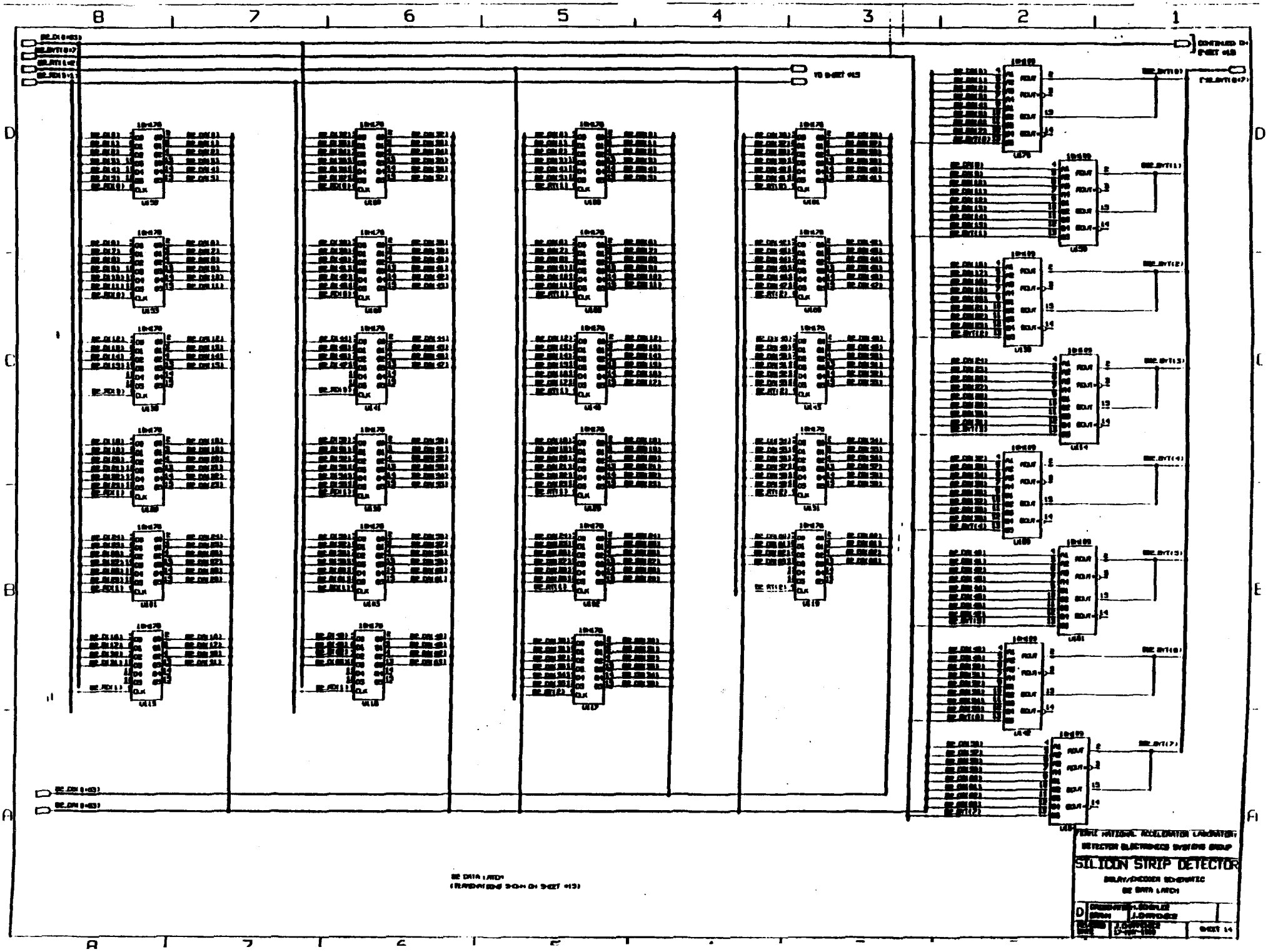
SCHEMATIC

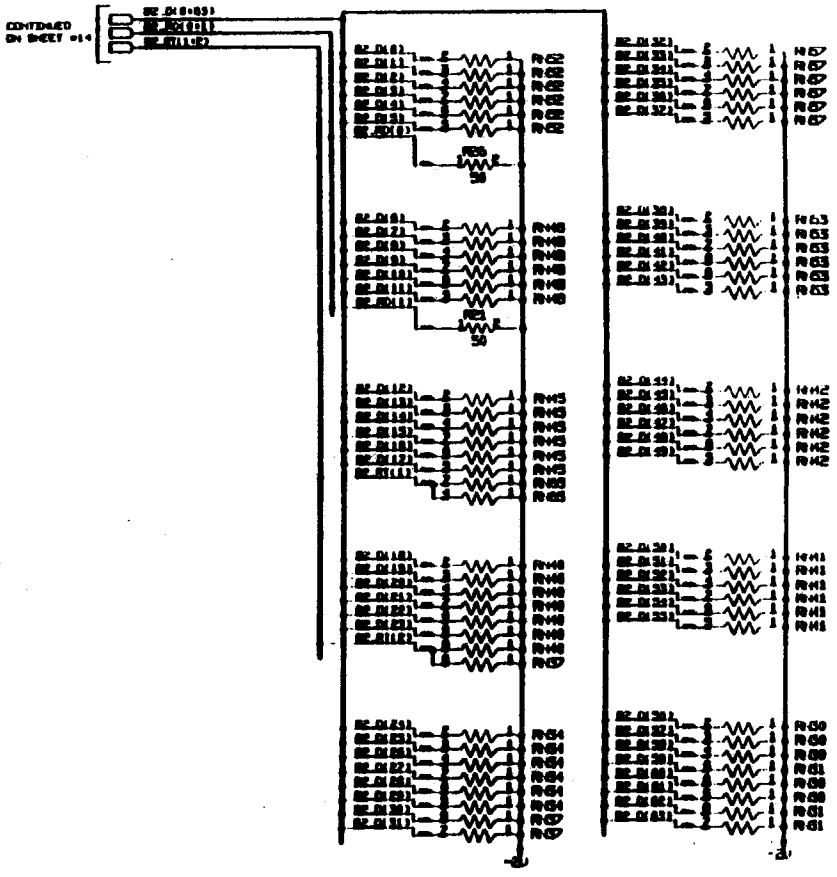
SVX II DATA READOUT

D	DESIGNED	H.GRIFFIN
	DRAWN	J.CHRISTIE
	REVISED	J.CHRISTIE
	DATE	1-FEB-1999

SHEET 12


8
7
6
5
4
3
2
1
D
C
B
A
CONTINUED ON SHEET
SILICON STRIP DETECTOR
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP
SHEET 13


SILICON STRIP DETECTOR

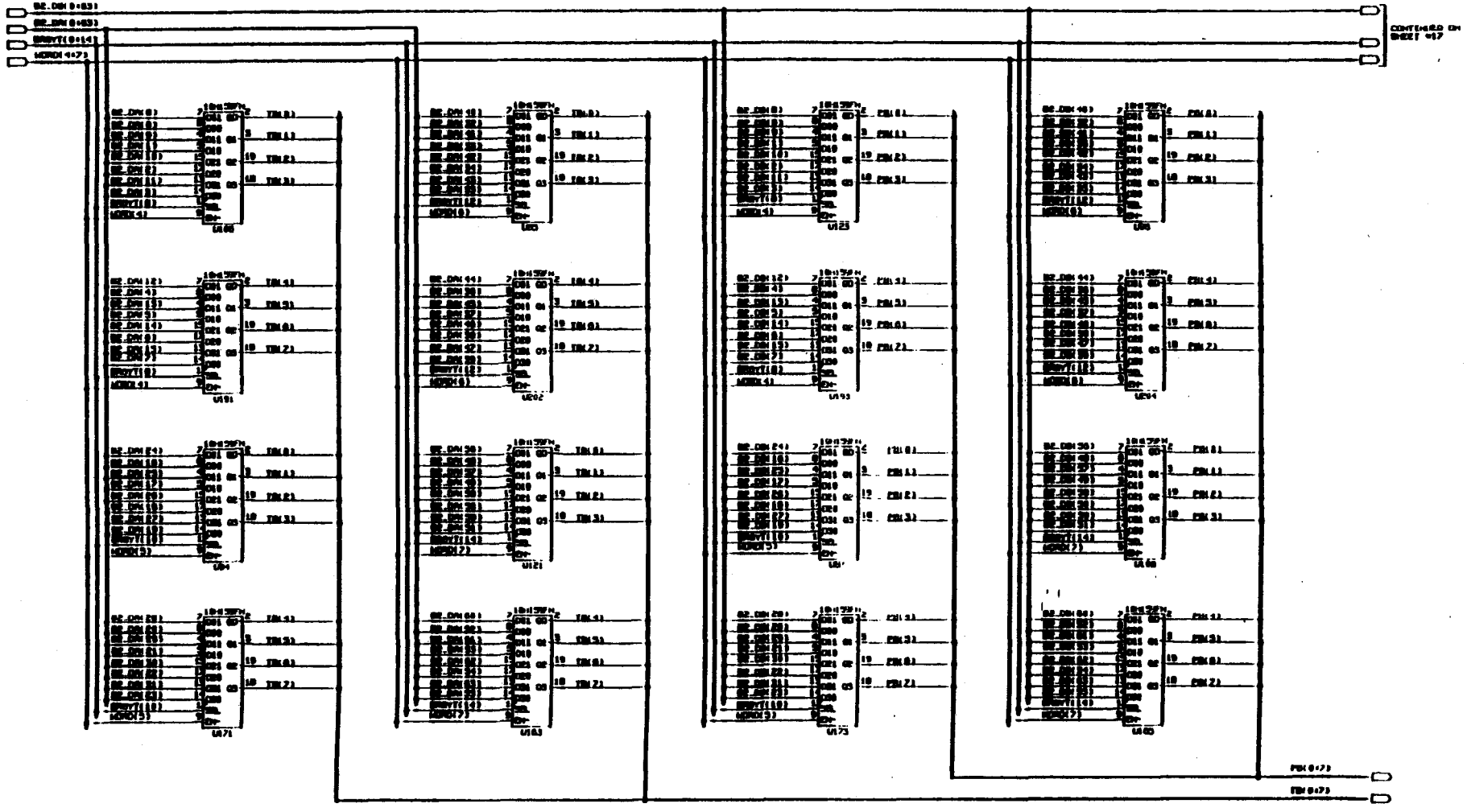


FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP

SILICON STRIP DETECTOR

DELAY/DECODER SCHEMATIC
TERMINATIONS FOR BANK 2 DATA LATCHS

D	DRWN		
	DES		
REVISED			SHEET 13
DATE		1991	



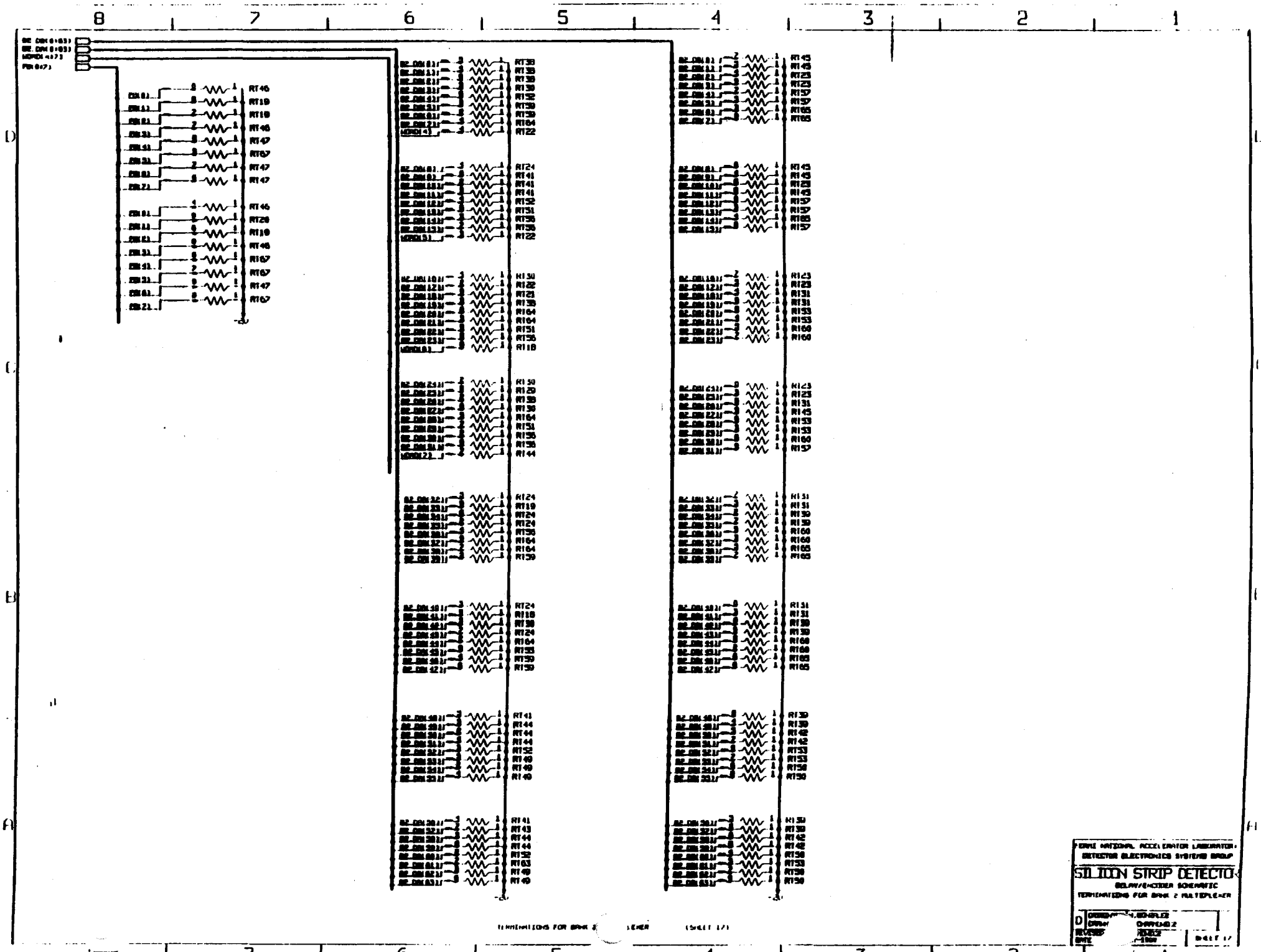
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP

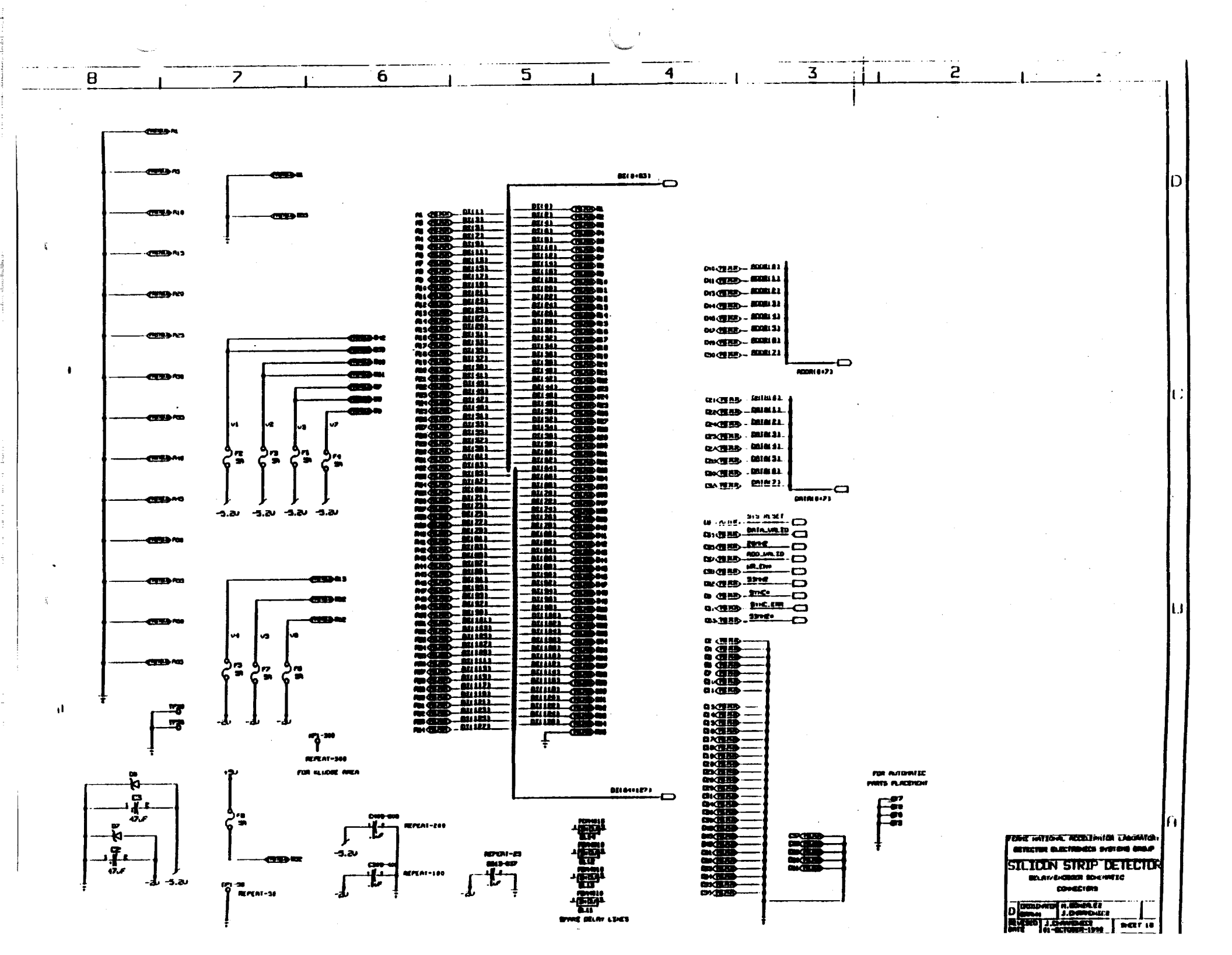
SILICON STRIP DETECTOR

DELAY/ENCODER SCHEMATIC

DRWG 2 MULTIPLEXER

D	DESIGNATION: DRAWN	H.BOWLES J.DOMINGUEZ
REVISED DATE	J.DOMINGUEZ 12-???-1990	SHEET 16


8
7
6
5
4
3
2
1
D
C
B
A
SILICON STRIP DETECTO
DELAY/EMITTER SCHEMATIC
TERMINATIONS FOR
SHEET 1/1


8
7
6
5
4
3
2
D
C
B
A
-5.2V
REPEAT-300
FOR GLIDE AREA
REPEAT-200
REPEAT-100
REPEAT-50
REPEAT-25
SPARE DELAY LINES
FOR AUTOMATIC
PARTS PLACEMENT
SILICON STRIP DETECTOR
DELAY/ENCODER SCHEMATIC
CONNECTORS
SHEET 10

Delay/Encoder Specifications

##

This is the specification of the Delay/Encoder module to be used on the silicon strip detector readout system for E771 and E789. Comments and questions can be delivered to Hector L. Gonzalez at Ext 2773.

##

Table of Contents

1. General Information

1.1. Purpose

The front end readout electronics for the Silicon Strip Detector is designed to process data at the RF bucket frequency, 53MHz. The Delay/Encoder(DE) module has been specified to accept data at 53MHz, provide a delay mechanism while a trigger decision is made, and generate an address hit list upon a Level 1 accept signal. A simplified block diagram is provided in Figure 2.

The delay element continuously stores data while the level 1 system is processing data corresponding to previously stored events. The delay is implemented in RAM and it is required for the control system to map the level 1 decisions into an eight bit address. The current implementation assumes that processing of an event takes about 1μsecond from the time that it is loaded into the DE. It is mandatory that the event acceptance be time ordered and the decision time be fixed with respect to the event occurrence. The address of accepted events is broadcasted by the Master Timing Controller to all Sequencers modules in the system and each Sequencer addresses the DE in its crate. The addressing mechanism triggers the DE to read the event from memory and transfer it to the encoder section.

The data encoding scheme uses the trigger bucket and the previous bucket simultaneously to generate an address hit list. A flag is asserted whenever the previous bucket has the bit set for the address been output. The address hit list is transmitted synchronously to a crate Sequencer module which serves as a crate controller and event builder for two planes of silicon strip data. The Sequencer is capable of transmitting hit data over fiber optic at 40Mbytes/sec or being readout through Fastbus.

This document includes figures and timing diagrams intended to simplify the specifications. In some cases, specifically the timing diagrams, the information is an attempt to specify the module and its interface with other system components.

1.2. Silicon Strip Readout System

This section presents a simplified block diagram of the silicon strip readout system. Figure 1 shows the interconnection of all the

modules that are referenced in this document. A brief description of each module follows.

PC - PostAmp/Comparator board, 12 per crate. Processes 128 pre-amp silicon strip signals, outputs discriminated data to the Delay/Encoder and outputs analog and digital sums to Level 1.

DE - Delay/Encoder board, 12 per crate. Provides event buffering for the PC discriminated data and for level 1 accepted events transmits a hit list to the Sequencer.

SEQ - Sequencer board, 1 per crate. Fans out system clock to PC and DE, initiates the encoding of a event, stores, pipelines and transmits encoded events to the next level. The events can be readout through FASTBUS.

FSCC- Fastbus Smart Crate Controller board, 1 per crate. Initializes the crate by excercising control over the SEQ, runs local diagnostics and provides an alternate data path to readout events.

MTC - Master Timing Control board, 1 per silicon strip readout system. The MTC synchronizes the 12 SEQ on the system by providing timing and control. Some of the functions that it performs are listed below:

- Distributes the RF clock to all SEQs.
- Maps a level 1 accept signal into an address of the DE memory and transmits addresses to all SEQs.
- Queue level 1 accepted events.
- Controls the write enable signal for the DE.
- Responds to READY and ERROR condition from the SEQs.
- Interface with the overall experiment controller.
- Synchronize System.

The readout system consists of 12 readout crates and a control crate that contain the MTC and other special modules. Each of the readout crates processes two planes of silicon strip data. Data processing is done in groups of 128 strips by a PC and DE pair. The 12 DEs in a crate send data, in parallel, to the crate SEQ. For a formal description of the system refer to the 'Silicon Strip Readout Implementation Plan' document.

In the context of this specification an **event** is the output of the PC and they are generated every 18.9nanoseconds.

1.3. Application

The DE is being designed for the Silicon Strip Readout System for E771 and E789. The function of the module is hardwired and there is no other application for it beyond the ones described on this document.

1.4. Packaging

The board is a single width FASTBUS module that does not implement any FASTBUS protocol.

1.5. Power Requirements

The worse case estimated current for the module are listed below:

Voltage	Current	Power
-5.2V	17A	86W
-2.0V	5A	10W
+5.0V	<1A	

The estimated -5.2 volts typical current is around 15 amps.

1.6. Cooling Requirements

The module will operate at the temperature range provided by the FASTBUS cooling system.

2. Theory of Operation and Operating Modes

The DE module is a single width board packaged in FASTBUS that does not implement a FASTBUS interface. The DE accepts PC discriminated data from 128 silicon strips, provides event buffering, encodes and transmits accepted events to a crate SEQ. The module communicates with the PC and SEQ through a special FASTBUS auxiliary backplane. The auxiliary connector signals for the DE are described in Appendix A.

The module is divided in two independent functions; the Delay and the Encoder, see Figure 2. The following sections provide a brief description for each function.

2.1. Delay

The Delay element receives 128 channels of discriminated data from the associated PC and provides buffering for upto 256 events (~4.8 μseconds). During data acquisition the DE continuously stores data in a FIFO like memory, while the level 1 system is making decisions for previously stored events. The DE does not implement any logic to prevent overwriting interesting events. This operation is delegated to the MTC which keeps track of the system write pointer (for DE) and the events been queued. For system implementation reasons, it is mandatory that event acceptance be time ordered and the decision time be fixed with respect to the event occurrence.

The delay element control logic requires that the 53MHz input (CLK2) be a 50% duty cycle clock. The logic splits the 18.9 nanoseconds time slice of each bucket into a read and write periods for a combined bandwidth of 106MHz. The write operation uses an address counter clocked by CLK2 and a write enable signal (WRITE*, asserted low) generated by the MTC and distributed in each crate by the SEQ.

The WRITE* signal is send 128 cycles before the SYNC* pulse. The DEs retime this signal using the CLK2 and SYNC* signals for synchronizing the start of event acquisition, see timing 2. The DEs will track their synchronization by checking that the write address is zero when the SYNC* signal is asserted. Note that prior to the assertion of WRITE* the DEs had been reset, which forces the write address to zero.

The readout of an event occurs when the MTC receives an accept pulse from the level 1 trigger system. The MTC maps the

pulse into an address for the DEs memory and sends the address to all SEQs in the system. The address generated by the MTC shall correspond to the previous bucket location. The previous and accepted buckets are loaded into registers and the Encoder is enabled to begin encoding that particular event.

2.2. Encoder

The Encoder is a simultaneous two bucket hit-list address generator. The encoding is performed in two stages, byte and bit levels, see Figure 3. At load time, the byte encoder performs byte integration, see Figure 4. The output of this process is a 16-bit word with a bit set for bytes with hit channels. At encoding time the byte encoder sequentially selects bytes to be processed by the bit encoder. The bit encoder (block ENCODER-8) loads the input data when ready and outputs a byte wide address stream of asserted bits. The encoded address is formed by concatenating the byte address with the bit address. In addition the bit encoder sets a flag (DATA0) whenever the previous bucket has a hit for the address being output. The hit list is generated from low to high address and no hit count is generated by the DE. An example of byte encoding is shown below:

Bit #	Previous Bucket	Trigger Bucket	Hit Type	Bit ADD (HEX)	Flag
7	1	0	A	7	1
6	0	0	-	-	-
5	0	1	B	5	0
4	1	1	C	4	1
3	0	0	-	-	-
2	0	0	-	-	-
1	1	1	C	1	1
0	0	1	B	0	0

Note: The encoder never looks at addresses without hits, this is represented with "-".

The alternative to remove type A hits is implemented on the prototype.

There are two options for the address hit list data transfer, 53MHz or 26.5MHz, CLK2 and CLK3 respectively. These options are switch selectable for each DE. Its implementation requires that the skew on the rising edge of CLK2 and CLK3 be kept to less than 2 nanoseconds.

3. Input/Output Specifications

The DE is a two port module, PC port and a SEQ port, See Figure 2. Both of these ports are implemented on an application specific Fastbus Auxiliary backplane that pairs a PC with a DE and has separate connections between each DE and the SEQ.

3.1. Communication Interfaces

3.1.1. PostAmp/Comparator Port

The PC port is a 128 bit uni-directional single-ended ECL connection from a PC to the associated DE. Synchronization of this port is controlled by the SEQ supplied CLK1 and CLK2 clocks. CLK1 is remotely programmed through FASTBUS and the rising edge is used by the PC to latch the silicon strip discriminated data. CLK2 is referenced (delayed) to CLK1 such that the PC output data is valid while CLK2 is high at the DE, see Timing 1. It is required that data on this port be valid for at least 12 nsececonds simultaneously at all DE in a crate.

3.1.2. Sequencer Port

The Sequencer port is the access port for the SEQ to readout the level 1 accepted events. This data port is a byte wide point-to-point connection designed to support a 53MHz data transfer rate.

The port provides an address bus, a data bus and control signals. To initiate a transfer the Sequencer supplies the DE with an event address and asserts the Add_Valid signal. After a fixed delay the Encoder will assert a Data_Valid signal (if hits present) and start transmitting the address hit list to the SEQ until completion. The data transfer is synchronous with CLK2 or CLK3 depending on the

user selected encoder operating frequency. A non-detailed timing diagram is provided in Timing 4.

3.1.3. Front Panel

The DE front panel is intended to provide information that will help diagnose problems on the DE. The prototype provides test points for the previous and trigger bytes been processed by the bit-encoder and the byte mask. The byte mask provides a bit for each of the 16 bytes of the 128 channels. A bit is asserted for the bytes that have a hit. As the bytes are processed the bits of the mask are cleared.

4. Initialization

Initialization of the DEs is achieved by asserting the RESET signal. Then the MTC initiates the enable of the write process and after the appropriate delay the enable of the Level 1 system.

After reset, the write counter points to location zero, the Encoder is in the ready state and all control signals driven by the DE are negated. The read counter is not initialized because it is loaded on demand when an event is accepted.

5. System and Module Diagnostics

5.1. Hardware Test

The DE does not implement any internal diagnostic tests or FASTBUS interface to access it's memory. The decision of excluding these features is based on timing and power considerations.

For debugging, a test module that emulates the PC output port and the SEQ was designed. The tester will be operated by an intelligent FASTBUS master, i.e. the FSCC or TSC.

5.2. System Test

For system diagnostics the PC should be able to supply data patterns that the DE stores into it's memory and events to be encoded are requested through the SEQ.

6. Appendix A

This appendix describes the DE auxiliary connector signals used on the silicon strip readout crates. In addition any other signals of interest are described.

CLK1 - A 53MHz clock driven by the SEQ and used by the PC to latch data. This clock is remotely programmable through the FASTBUS port on the SEQ.

CLK2 - A 53MHz clock driven by the SEQ. The clock is a delayed version of CLK1 used to synchronize the write process in the DE with the output data of the PC, to generate internal timing and as a reference when transmitting data to the SEQ at 53MHz.

CLK3 - A 26.5MHz clock driven by the SEQ. An alternate clock used to transmit data to the SEQ.

DI(0:127) - Input discriminated data driven by the associated PC. A 100 ohms termination is provided by the DE.

Address(0:7) - The address bus (bussed to all DE) driven by the SEQ to transfer event addresses to the DE.

Add_Valid - Signal driven by SEQ to validate ADD(0:7).

Write_En* - Write enable signal distributed on the backplane by the SEQ. The signal is controlled by the MTC or through a FASTBUS register on the SEQ.

Sync* - Synchronization signal generated by the MTC and used by the DEs to test write counter synchronization at each zero crossing. If a DE has a write counter different from zero then it is out of synchronization.

Sync_Err - Signal asserted by a DE that is out of synchronization and received by the SEQ. The signal is wire-ored on the backplane.

Reset - Reset signal distributed on the backplane by the SEQ.

Data_X(0:7) - Encoded hit list data bus for Delay/Encoder X, where X is a hex number assigned to each DE. Terminated by the SEQ.

Data_ValidX - Data valid signal asserted by Delay/Encoder X, where X is a hex number assigned to each DE. Terminated by the SEQ.

appendix B

This appendix is intended to explain the system diagnostics involving the PC output port, the DE and the SEQ. There are two system test of interest. The first one involves the complete silicon strip readout system (starting at PC), the other one performs test on selected crates. Details about these test have not been worked out and the implementation steps presented here should be modified as development of the involved modules progress. A series of steps follows:

Step1: The FSCC enable/disable specific PCs and select the data pattern type, i.e. static or dynamic. Also, it initializes the SEQ for global or local test.

Step2: The FSCC reset the DE through the SEQ.

Step3: Simultaneously enable the PC to generate patterns and the DE to start recording events. This is, the PC receives a signal of similar timing to the DE write enable signal.

Step4: Write an event address to the SEQ if the test is local or to the MTC if the test is global.

Step5: For local test, the FSCC reads the event from the SEQ. For global test the SEQ sends the event to the next level.

Step6: Check the event and display/record errors.

Step7: If test enable, go to Step1 or Step 4.

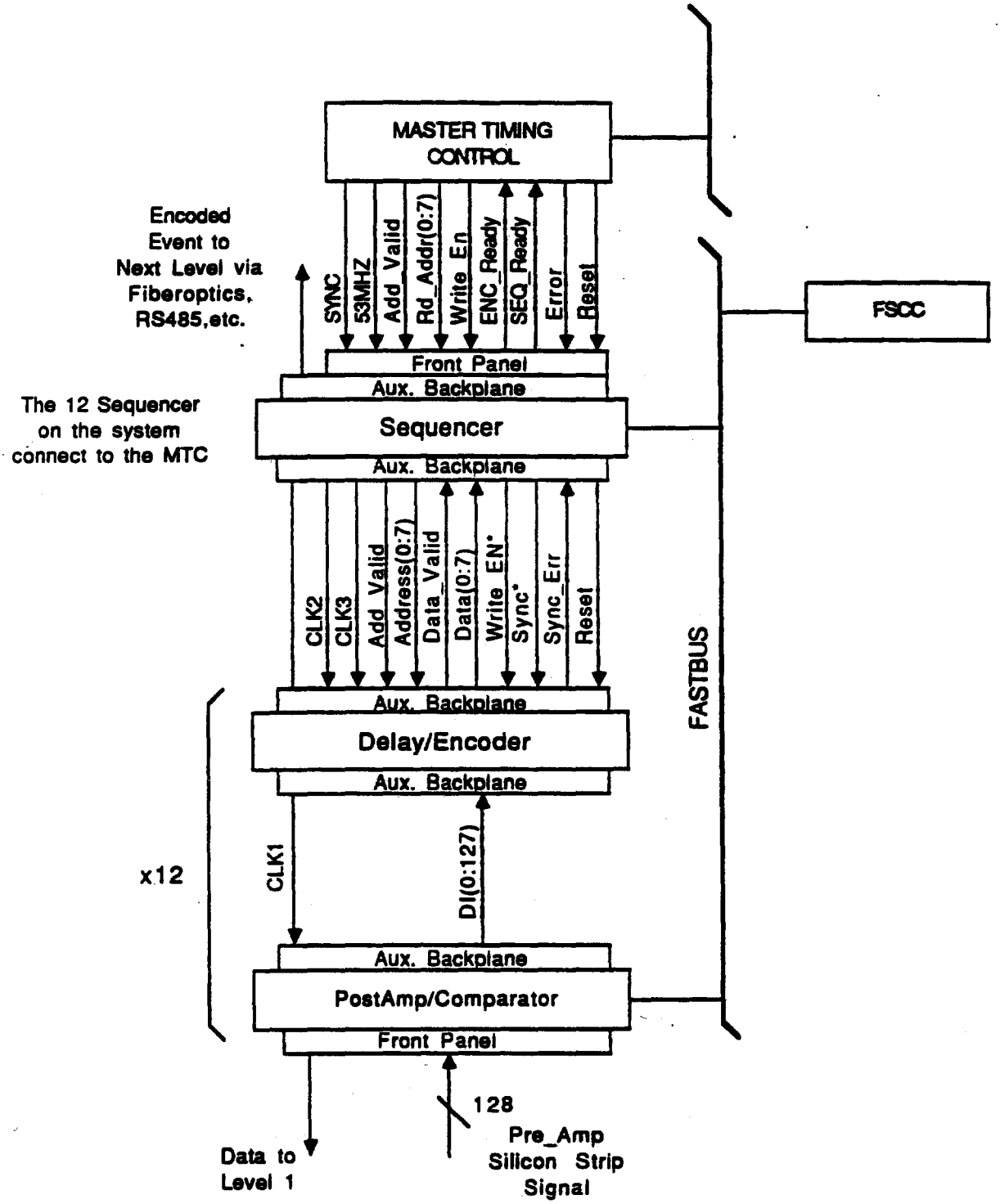
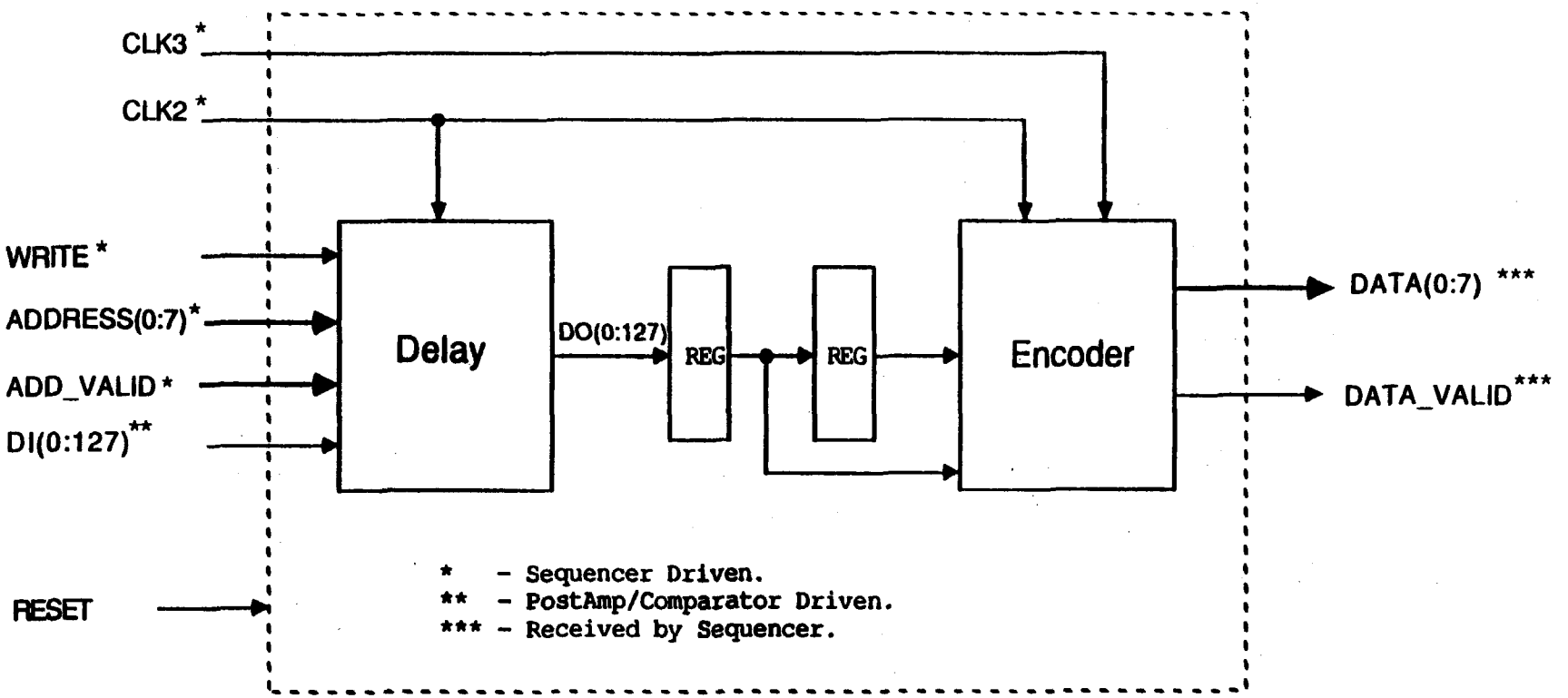


FIGURE 1: Silicon Strip Readout System



PC PORT:

DI(0:127)	- PC output data.

SEQ PORT:

WRITE	- Write enable for the delay memory.
ADDRESS(0:7)	- Accepted bucket address.
ADD_VALID	- Validates ADDRESS(0:7).
DATA(0:7)	- Channel to transmit the address hit list.
DATA_VALID	- Enable the Sequencer to clock data into its FIFO.

Figure 2: Delay/Encoder Block Diagram

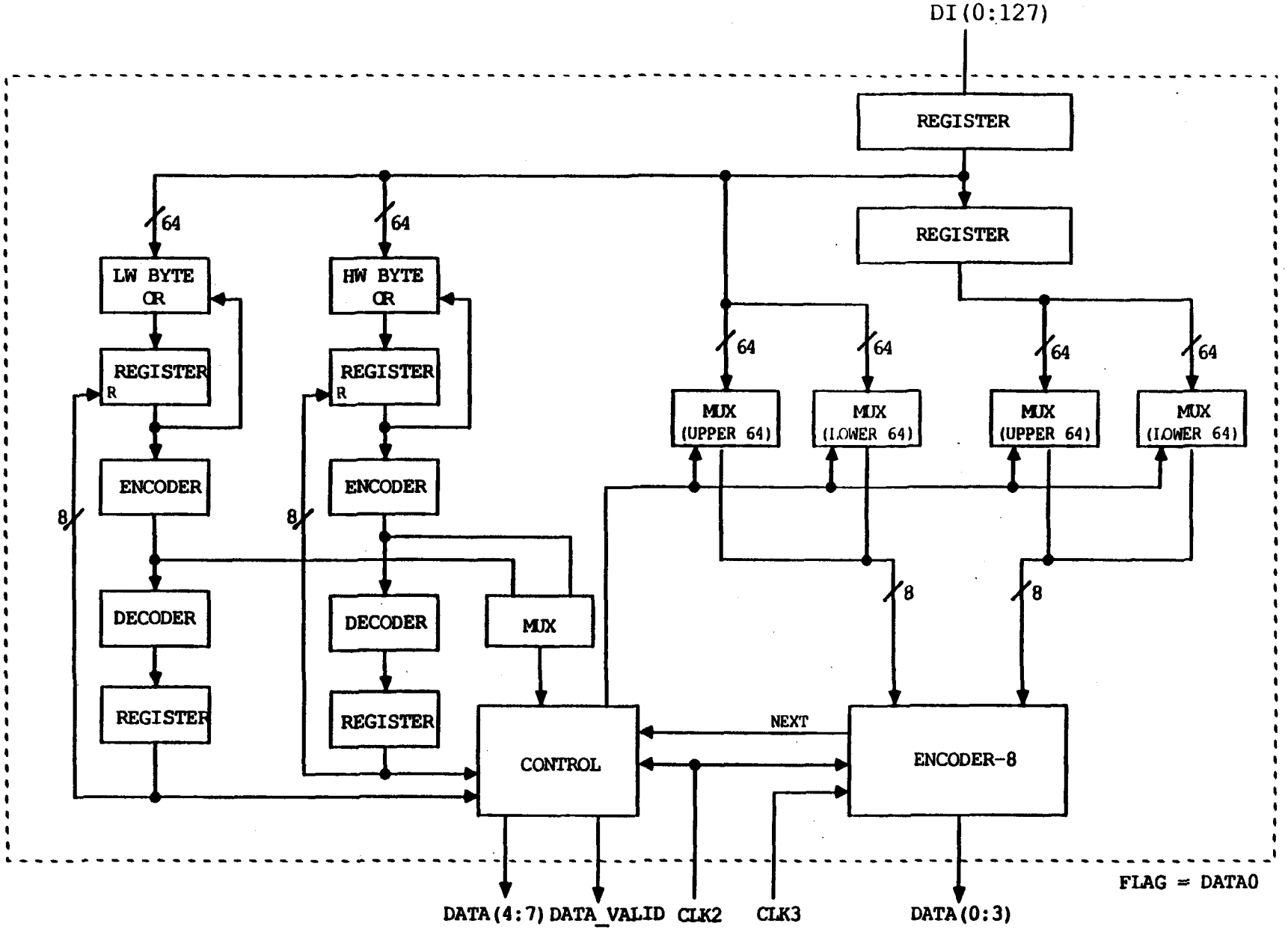


Figure 3: 128-BIT ENCODER

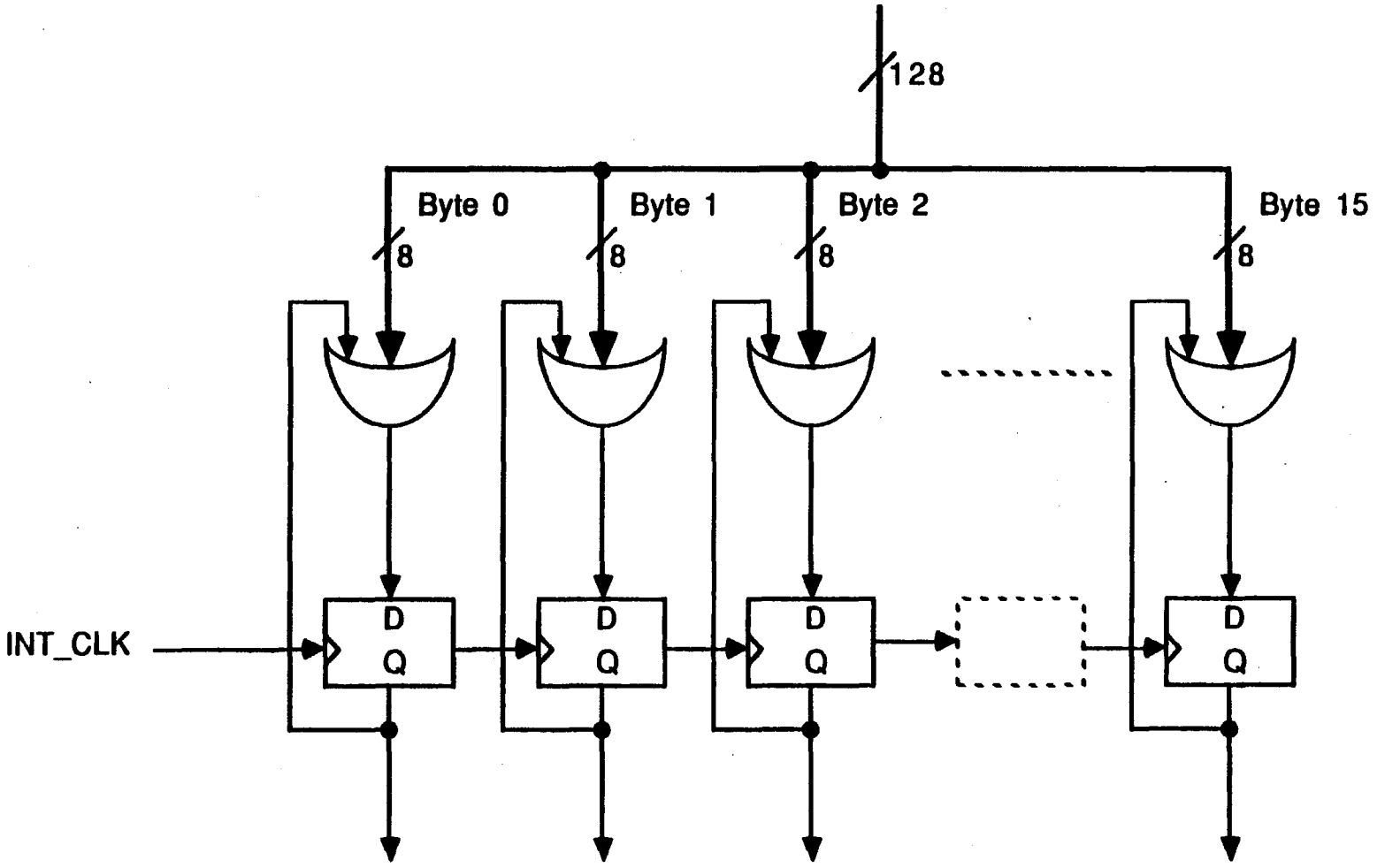
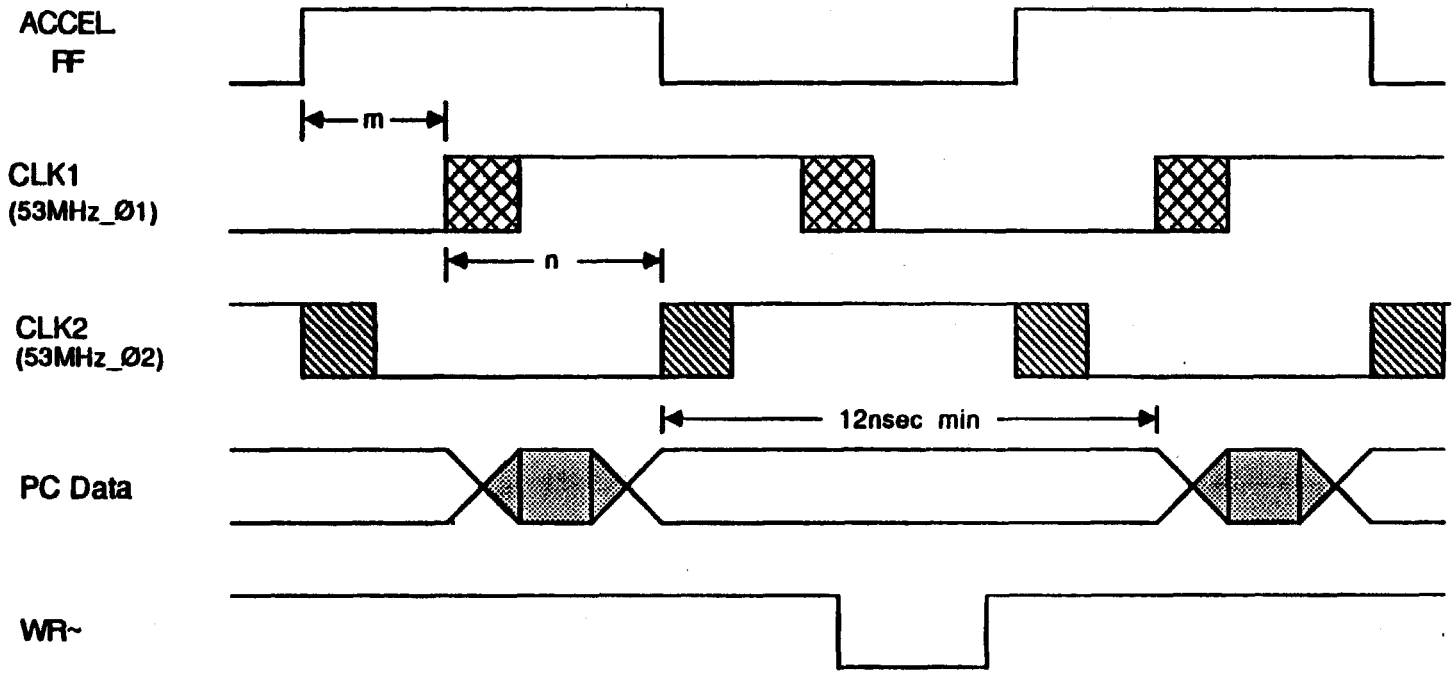
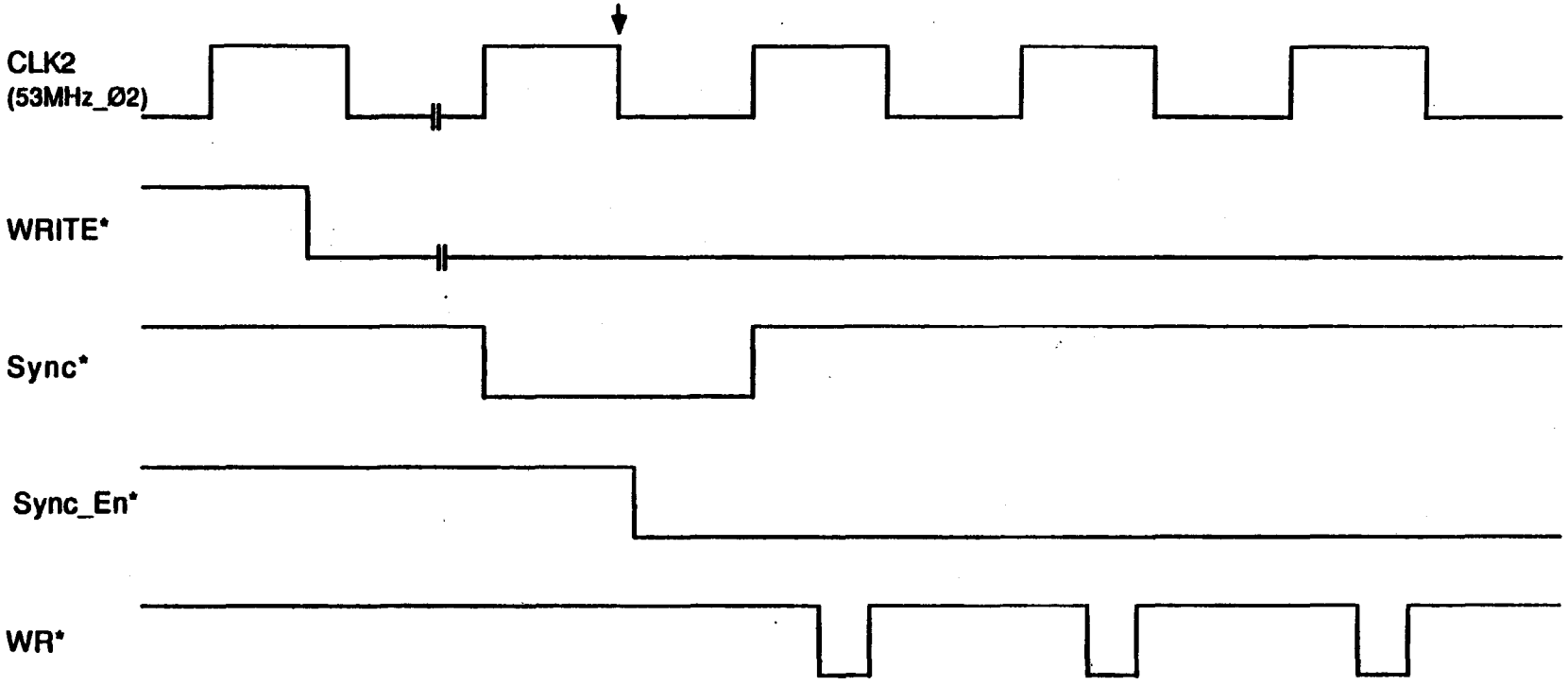


Figure 4: Byte Integration



m - Delay added to synchronize the PC to the accelerator RF.
n - Delay added to CLK1 clock to synchronize the DE to the PC output data.

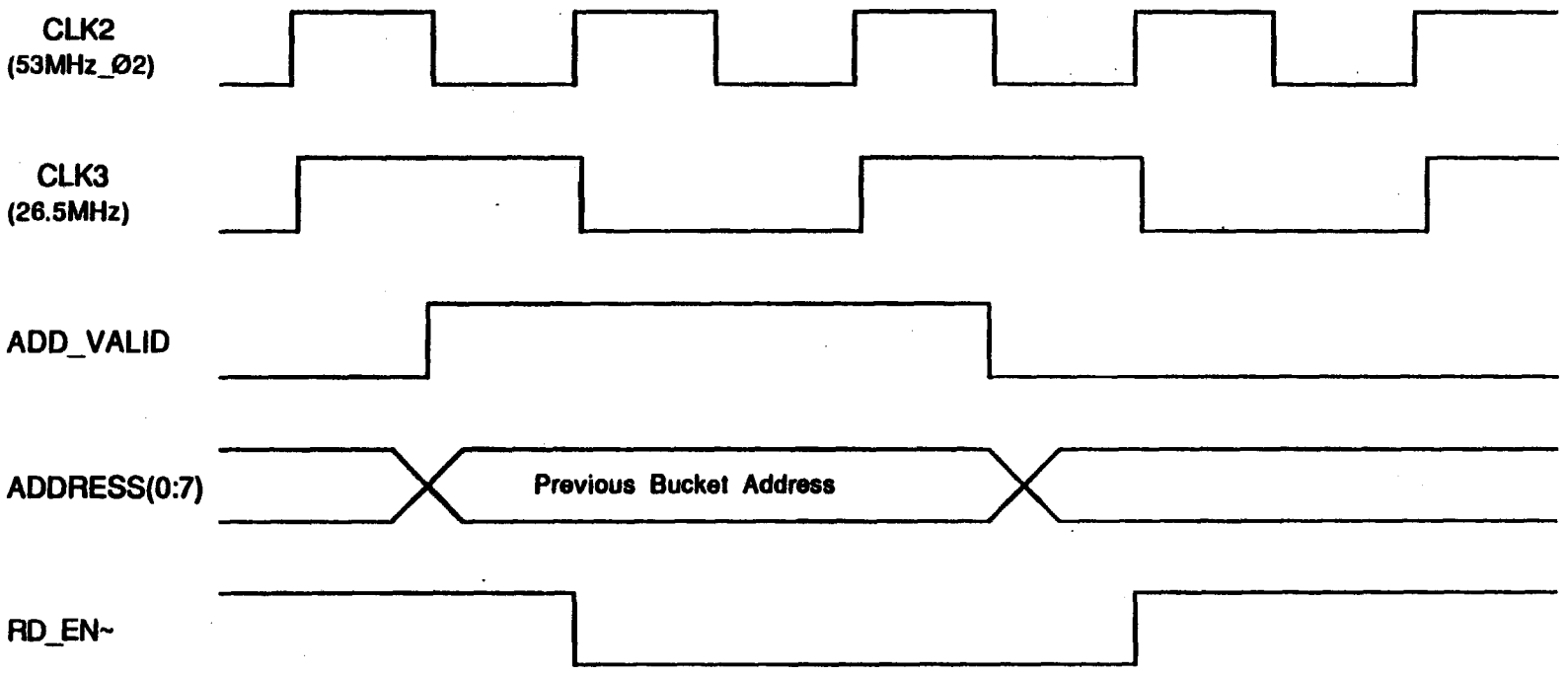
Timing 1: Synchronization of DE to PC.



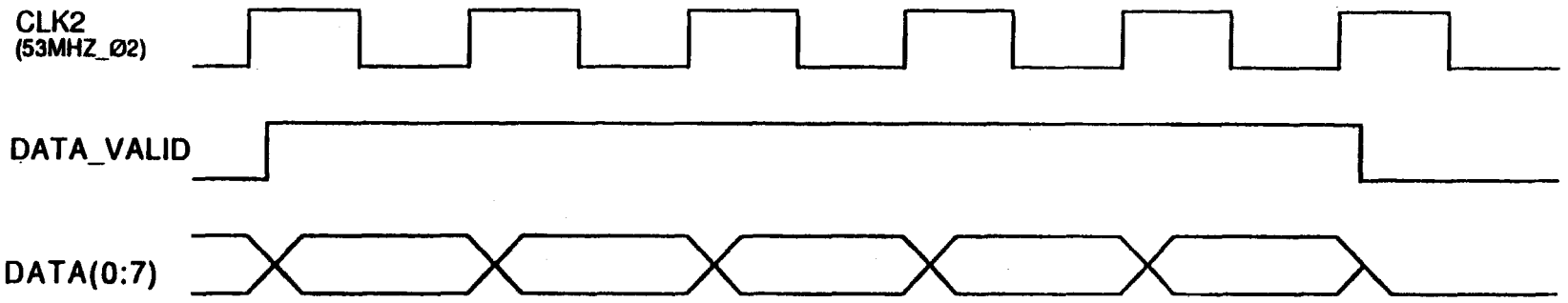
DESCRIPTION:

WRITE* - Write enable signal on the auxiliary backplane driven by Sequencer.
SYN_EN* - DE internally synchronized write enable.
WR~ - Write pulse to memory.

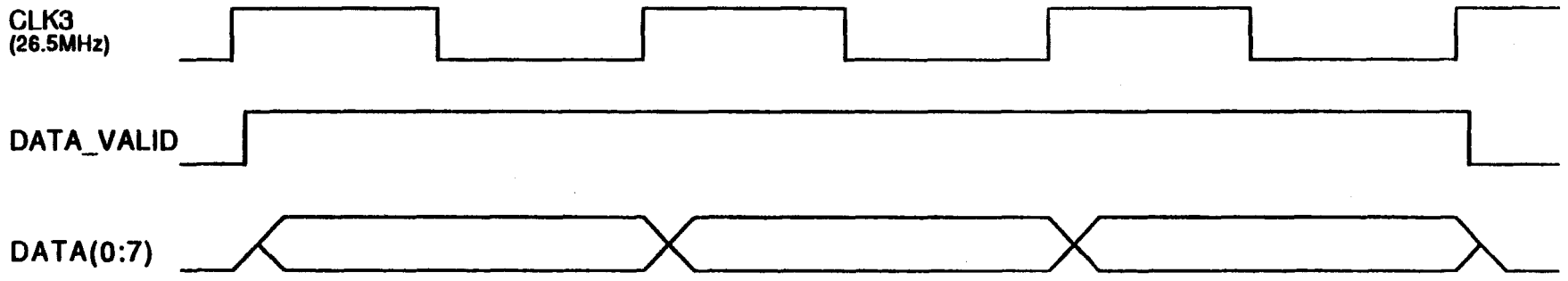
Timing 2: Write Enable Synchronization.



Timing 3: Address Transfer from SEQ to DE.



Timing 4a : 53MHz Encoded Event Transfer



Timing 4b: 26.5MHz Encoded Event Transfer

Fermi National Accelerator Laboratory

January 3, 1991

TO: Distribution
FROM: Carl Swoboda
SUBJECT: Sequencer Hardware Description

The attached document is the "as built" hardware description for the SSD Sequencer module. PLease add this document to your SSD Readout System binder.

Distribution:
David Christian
Brad Cox (E771)
Peter Garbincius
Franco Grancagnolo (E771, MS 219)
Hector Gonzalez
Bill Haynes
Rick Kwarciany
Wolfgang Kowald (E771)
Mark Larwill
Cash McManus (E771)
Garry Moore
Tom Nash
John Peoples
Ruth Pordes
Matteo Recagni (E771)
Robert Trendler
Ken Treptow

Fermi National Accelerator Laboratory

Silicon Strip Detector Readout System

SEQUENCER MODULE HARDWARE DESCRIPTION

R. DeMaat, M. Larwill, A. Romero

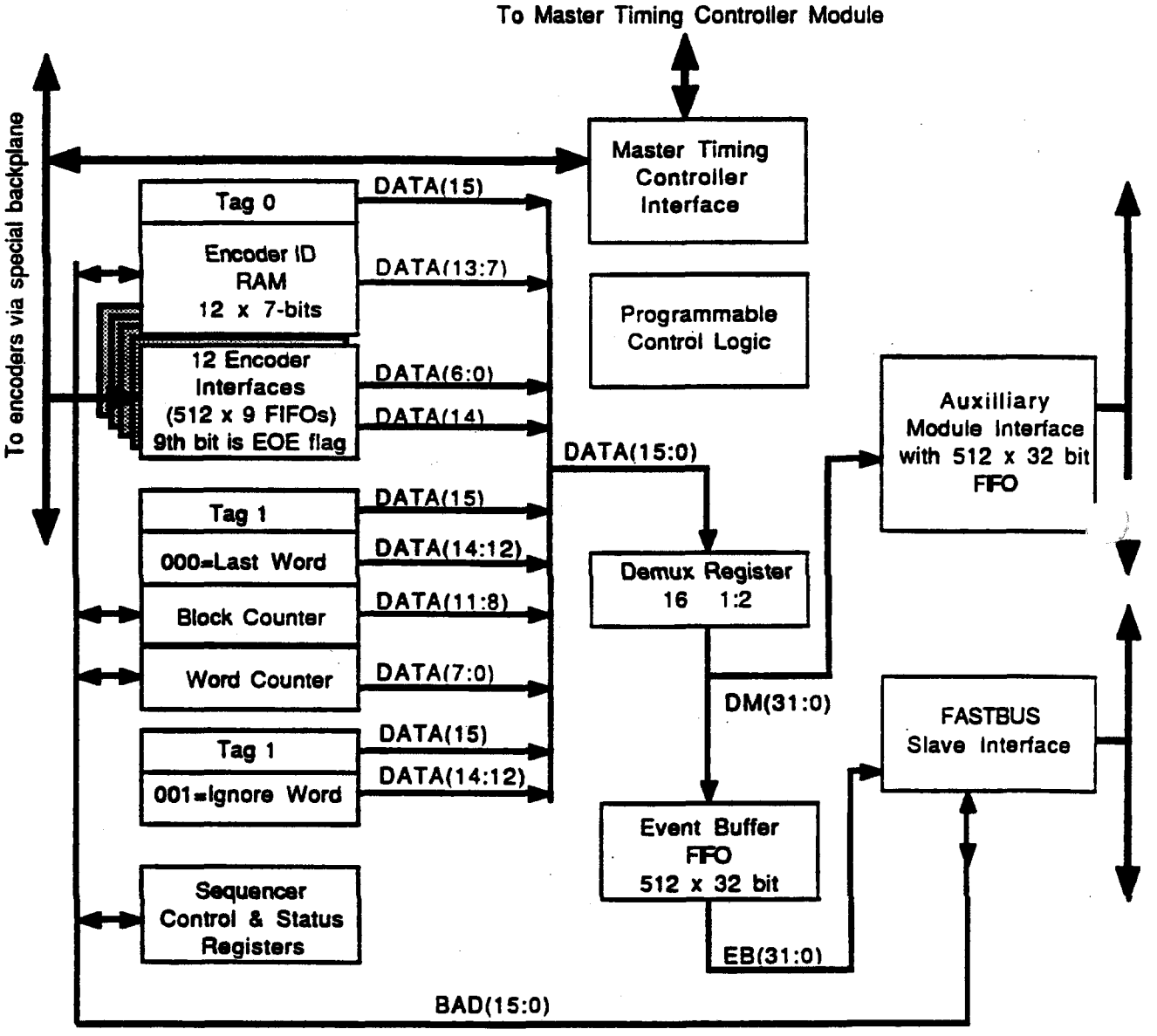
December 6, 1990

Fermilab Document Number HN 99

1. GENERAL INFORMATION

1.1. Purpose

This document describes the "Silicon Strip Detector Readout System Sequencer Module" hereafter referred to as the sequencer. As one component of a larger system, this module accepts silicon strip detector data from the twelve encoder modules in its FASTBUS crate, combines the data and outputs it to other system components via FASTBUS or via an auxiliary link such as RS-485, ECLine or fiber optics. The general block diagram of the sequencer is as follows:

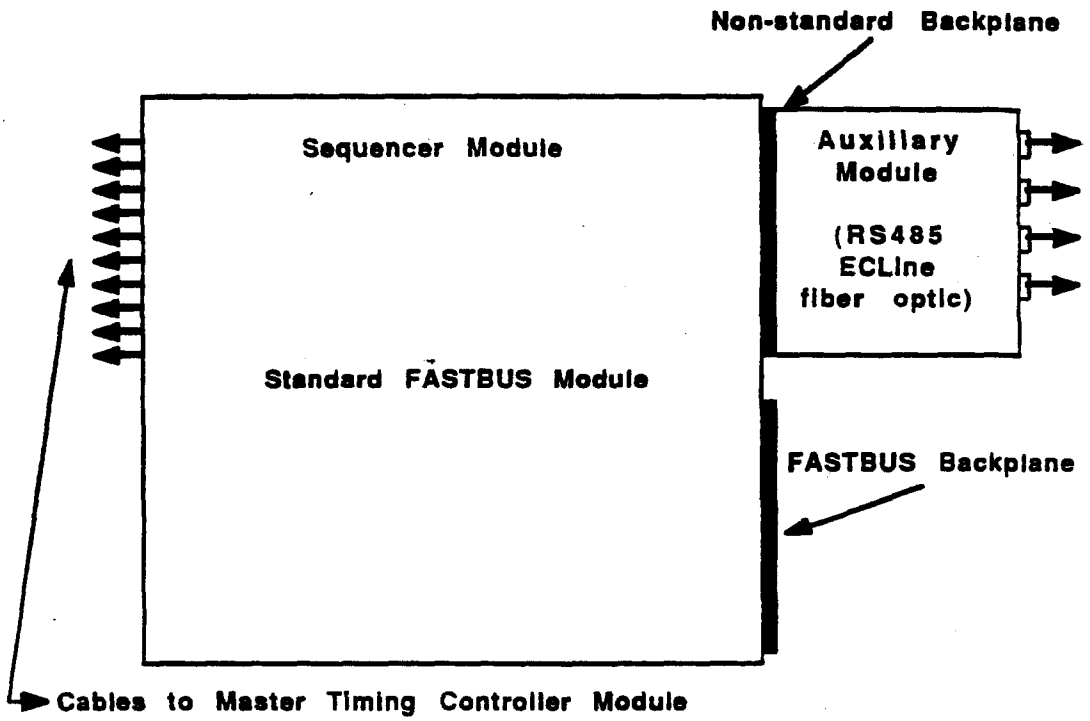


1.1.1. Standard Bus System Used

The sequencer is a single width FASTBUS module and includes FASTBUS slave capability. The FASTBUS slave is used for the initialization of some programmable parameters on the module and may be used to read out the encoded silicon strip detector data.

In addition to the standard FASTBUS interface, the sequencer uses a non-standard backplane on the FASTBUS auxiliary connector position to communicate with the encoder and postamp/discriminator modules. This backplane is optimized for high speed parallel transfers.

Some of these FASTBUS auxiliary connector pins are not used by the special backplane to communicate with the other modules. These pins are used instead by the sequencer to communicate with an I/O link driver such as RS-485, ECLine or fiber-optics. A special auxiliary interface module plugs into the back of the FASTBUS crate for this function. The figure describes this pictorially.

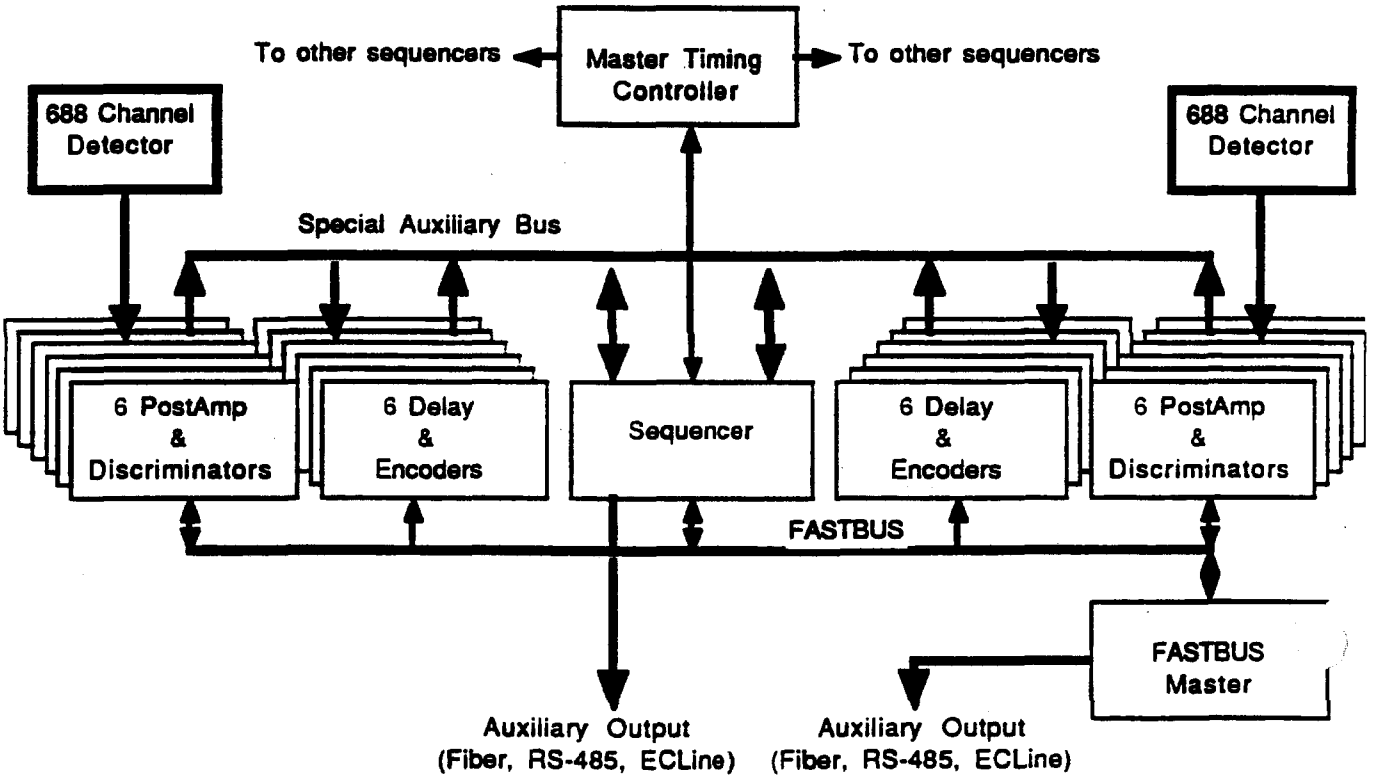


1.1.2. Number of Channels

Data comes into the sequencer from 12 encoder modules in the FASTBUS crate. The data is concatenated and may leave the sequencer via FASTBUS or auxiliary link (RS-485, ECLine or fiber optics).

1.2. Application

The sequencer module sits in the middle of the FASTBUS crate and communicates with the FASTBUS encoder modules via the special auxiliary backplane. Front panel connectors on the module connect via cables to the master controller to provide system clock, trigger commands, system RESET, errors and etc. The sequencer sends its data out via a rear mounted auxiliary card. Data may also be read via FASTBUS transfers. The sequencer may be used at Fermilab in experiment E771 and may be adaptable to other applications. In the E771 application, 24 planes of detector will be used requireing 12 crates of readout electronics. Each crate of electronics will be configured as pictured here:



1.3. Packaging

1.3.1. Module

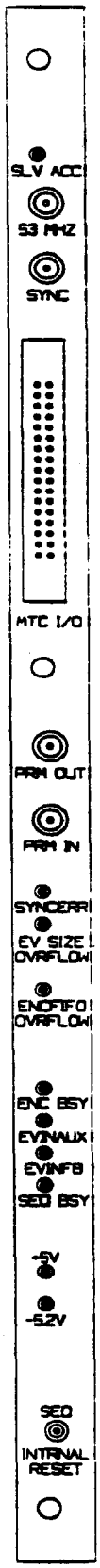
The sequencer is implemented on a standard single width FASTBUS module which is 15.878" X 14.437" plus front panel. The module plugs into a FASTBUS crate which is then installed in an equipment rack along with the FASTBUS power and cooling system.

If the auxiliary port is used to output data, then a FASTBUS standard size auxiliary module is also used which contains logic and interface hardware to provide a high speed RS-485, ECLine or fiber optic data link. This auxiliary module plugs into the rear of the backplane at the sequencer module position in the slot provided for it by the FASTBUS crate.

1.3.2. Front and Rear Controls Connectors and Displays

1.3.2.1. CLK2 Delay Switch

The 53 Mhz clock received via a front panel connector is delayed by a programmable delay line and sent to the post amp/discriminator modules as a signal named CLK1. CLK1 is delayed by another programmable delay line and sent to the encoder modules as a signal named CLK2. The total amount of delay of CLK1 and CLK2 is also introduced on a signal called SYNC. The programmable delay line for CLK1 is programmed by FASTBUS, but the delay line for CLK2 is set by a 6-bit DIP switch. This switch provides a binary progressive delay from 0 to 31.5 nsec in .5 nsec increments.



SLV ACC	The Slave Access LED illuminates whenever the Sequencer is accessed by FASTBUS
53 MHZ	53 MHZ clock input (NIM)
SYNC	SYNC signal input (NIM)
MTC I/O	34 pin MTC interface port
PRM OUT	Permit Out output conn.
PRM IN	Permit In input conn.
SYNCERR	Sync Error, LED
EV SIZE OVRFLOW	Event Size Overflow, LED
ENCFIFO OVRFLOW	Encoder Fifo Overflow, LED
ENC BSY	Encoder Busy, LED
EVINAUX	Event in Auxiliary Buffer Fifo, LED
EVINFB	Event in FASTBUS Buffer Fifo, LED
SEQ BSY	Sequencer Busy, LED
+5V	+5V present, LED
-5.2V	-5.2V present, LED
SEQ INTRNAL RESET	Sequencer internal circuitry reset button

1.3.2.2. Front Panel Displays

The sequencer module displays the following indicators of module operation and status on its front panel.

- FASTBUS SLAVE ACCESS; this LED indicator lights for a minimum of 100 msec each time a FASTBUS access to the sequencer occurs.

- SYNC ERROR, This LED indicator lights and remains lighted until RESET by FASTBUS any time an encoder module in the crate signals the sync error condition. Sync error is signalled by an encoder module if its write counter is not at the zero count when the sync signal is received from the Master Timing Controller via the sequencer.

- EVENT SIZE OVERFLOW, this LED indicator is lighted by the occurance of an event which produces more that 255 'hits'. The sequencer does not 'know' this error has occurred until the data has already been received into its encoder FIFOs and is being counted as it is read out to FASTBUS and/or the auxiliary port. The occurrance of this error does not justify having to initialize the system yet a problem exists in how to dispose gracefully of the extra data. In normal operation the control logic will truncate an event at 255 'hits', properly insert the 'LAST WORD' into the data stream, and remove all extra data from the encoder FIFOs without FASTBUS intervention. Optionally, under the control of a bit in CSR0, the control logic will output the additional data while allowing the word counter to wrap around. This will be useful for system tests.

- ENCODER FIFO OVERFLOW, This LED indicator lights and remains lighted until RESET by FASTBUS any time the encoder interface FIFOs are overfilled by data from the encoders. This is a system failure caused by the Master Timing Controller not using the SEQ_READY signal properly. When SEQ_READY is not asserted, the encoder data FIFOs are at least half full and the sequencer should not be given any new trigger addresses. At this time, there is room in the FIFOs to hold the data generated by a single trigger which may have been on its way to the sequencer before the Master Timing Controller received the SEQ_READY being deasserted. Any additional triggers sent by the Master Timing Controller after SEQ_READY is deasserted may generate enough data to overfill the FIFOs and generate the error.

- ENCODERS BUSY, This LED indicator is lighted by the logical 'OR' of the DATA VALID signals from each encoder in the crate. The signal is not 'stretched' or latched and normally occurs too fast to be seen. If this LED remains lighted, an encoder in the crate has failed and has effectively halted the system.

- EVENT IN AUXILIARY INTERFACE BUFFER, This LED indicator is lighted by the presence of data in the Auxiliary Interface FIFO. It will stay lit as long as that FIFO is not empty.

- EVENT IN FASTBUS BUFFER, This LED indicator is lighted by the presence of data in the FASTBUS event FIFO. It will stay lit as long as that FIFO is not empty.

- SEQUENCER BUSY, This LED indicator is lighted by the logical 'OR' of the FIFO half-full signals from each encoder FIFO ion the sequencer module. The signal is not 'stretched' or latched and normally occurs too fast to be seen. If this LED remains lit, the sequencer is not processing the FIFO data out to FASTBUS and/or the auxiliary interface and has effectively halted the system. This may be caused by data not being removed from the FASTBUS event buffer or by the auxiliary link not being ready, or by PERMIT IN not being asserted.

- +5 VOLT STATUS, this LED indicates that the board's +5 volt bus is powered up.

- -5.2 VOLT STATUS, this LED indicates that the board's -5.2 volt bus is powered up.

1.3.2.3. Front Panel Connectors

Signals marked '+' are 'party lined' in a single flat cable. One end connects to the Master Timing Controller (MTC) which terminates the signals. The sequencers attach at various places along the cable. The last sequencer module on the cable provides signal terminations.

- 53 Mhz CLOCK input, terminated NIM, from MTC.

+ WRITE ENABLE input, differential ECL, asynchronous to clock, from MTC. This signal is bussed to each encoder module over the auxiliary backplane. Enables encoders to accept hit data.

- SYNC input, terminated NIM, synchronous to clock, from MTC. This signal gets delayed by the same amount as CLK1 plus the same amount as CLK2 before being bussed to the encoder modules over the auxiliary backplane to test synchronization at each zero count.

+ RESET input, differential ECL, asynchronous to clock, resets control logic and fifos, bussed over the auxiliary backplane to the encoders where it resets write counters.

+ EVENT ADDRESS input, differential ECL, 8-bits, asynchronous to clock, bussed to encoder modules over the auxiliary backplane to designate a stored event. Note that this is actually the address of the event previous to the one requested by the trigger system. This is because the encoders are required to evaluate the previous RF bucket for 'hits' when encoding an event.

+ ADDRESS VALID input, differential ECL, asynchronous to clock, bussed to encoder modules over the auxiliary backplane to initiate encoding of a stored event.

+ SEQ_READY output, wire OR'd ECL, active low, signals that the sequencer has room in its fifos for events and thus can accept "read addresses".

+ ENC_READY output, wire OR'd ECL, active low, this is the sum of all encoder data valid signals. It signals to the master controller that the encoders are ready to accept another trigger.

+ ERROR output, wire OR'd ECL, active high, signals that the sequencer fifos have been overfilled by the encoders or that an encoder has lost synchronization. A fatal error requiring system RESET.

- PERMIT IN - Single ended TTL. A signal from the upstream module indicating that the sequencer may transmit. It's used to allow multiple sequencers to feed a party line.

- PERMIT OUT - Single ended TTL. A signal to a downstream module used to allow the next sequencer in a daisy chain to use the auxiliary party line if applicable.

1.3.2.4. Special Auxiliary Backplane Signals

- CLK1, delayed from the front panel 53 Mhz clock. Individually routed to each post amp/discrimminator module. The delay between the 53 Mhz clock received via the front panel, and this signal, is programmable via FASTBUS from 0 to 32 nsec in .5 nsec increments.

- CLK2, delayed from CLK1, individually routed to each encoder module. The delay between CLK1 and this signal is programmable via a 6-bit DIP switch from 0 to 32 nsec in .5 nsec increments.

- CLK3, 26.5 Mhz, CLK2 divided by two, bussed to all encoder modules for use as a data clock.

- RESET, Asynchronous, Bussed to all discriminators and encoder modules.

- WRITE ENABLE, Synchronized to 53 Mhz CLK2, bussed to all encoder modules, active low.

- EVENT ADDRESS, Asynchronous , 8-bits. Used to designate which memory location in the encoders to read out and encode. Note that this is actually the address of the event previous to the one requested by the trigger system. This is because the encoders are required to evaluate the previous RF bucket for 'hits' when encoding an event.

- ADDRESS VALID, Asynchronous , derived from the Master Timing controller signal . Used to validate Event Address.

- EVENT DATA, Synchronized to 26.5 Mhz CLK3, validated by data valid, 8-bits from each encoder module. Not bussed.

- DATA VALID, Synchronized to 26.5 Mhz CLK3, validates the event data, 1-bit from each encoder module. Not bussed.

- SYNC, This signal is received at the front panel, then delayed by the total of CLK1 delay plus CLK2 delay, then sent to the encoder modules. It is used to verify that all encoder module write counters are at the same count (zero) at sync time. Active low.

- SYNC ERROR, Asynchronous bussed signal from the encoder modules, signifies that one or more of the encoders is out of sync with the system.

1.4. Power Requirements

+5 Volts @ 9 Amps

-5.2 Volts @ 3 Amps

-2 Volts @ 2 Amps

1.4.1. Control and Monitoring Requirements

No requirement exists for a special control or monitoring of the power supply to the sequencer module. The normal protection provided by the FASTBUS crate enviremental system is sufficient.

1.5. Cooling Requirements

The sequencer represents a heat load of 65 watts which must be absorbed and carried away by the FASTBUS cooling system.

1.6 Sequencer Module SSD Auxiliary Backplane Pin List

C Connector	B Connector	A Connector
C01-H53MHZ,Ø1 Clock, Slot 25,23,	B01-Reset	A01-H53MHZ,Ø1 Clock,Slot 11,
C02-L53MHZ,Ø1 Clock, 21,19,17,15	B02-GND	A02-L53MHZ,Ø1 Clock,9,7,5,3,1
C03-Hit Data 0, Slot 24	B03-Fiber Error	A03-Hit Data 0, Slot 2
C04-Hit Data 1, Slot 24	B04-GND	A04-Hit Data 1, Slot 2
C05-Hit Data 2, Slot 24	B05-Fiber Wait	A05-Hit Data 2, Slot 2
C06-Hit Data 3, Slot 24	B06-GND	A06-Hit Data 3, Slot 2
C07-Hit Data 4, Slot 24	B07-Fiber Clock	A07-Hit Data 4, Slot 2
C08-HIt Data 5, Slot 24	B08-GND	A08-Hit Data 5, Slot 2
C09-Hit Data 6, Slot 24	B09-Sync	A09-Hit Data 6, Slot 2
C10-Hit Data 7, Slot 24	B10-Sync Err	A10-Hit Data 7, Slot 2
C11-Data Valid, Slot 24	B11-GND	A11-Data Valid, Slot 2
C12-VCC, +5.0 Volts	B12-Fiber D00	A12-VEE, -5.2 Volts
C13-Hit Data 0, Slot 20	B13-GND	A13-Hit Data 0, Slot 6
C14-Hit Data 1, Slot 20	B14-Fiber D01	A14-Hit Data 1, Slot 6
C15-Hit Data 2, Slot 20	B15-GND	A15-Hit Data 2, Slot 6
C16-Hit Data 3, Slot 20	B16-Fiber D02	A16-Hit Data 3, Slot 6
C17-Hit Data 4, Slot 20	B17-GND	A17-Hit Data 4, Slot 6
C18-HIt Data 5, Slot 20	B18-Fiber D03	A18-Hit Data 5, Slot 6
C19-Hit Data 6, Slot 20	B19-GND	A19-Hit Data 6, Slot 6
C20-Hit Data 7, Slot 20	B20-Fiber D04	A20-Hit Data 7, Slot 6
C21-Data Valid, Slot 20	B21-GND	A21-Data Valid, Slot 6
C22-GND	B22-Fiber D05	A22-GND
C23-Hit Data 0, Slot 16	B23-GND	A23-Hit Data 0, Slot 10
C24-Hit Data 1, Slot 16	B24-Fiber D06	A24-Hit Data 1, Slot 10
C25-Hit Data 2, Slot 16	B25-GND	A25-Hit Data 2, Slot 10
C26-Hit Data 3, Slot 16	B26-Fiber D07	A26-Hit Data 3, Slot 10
C27-Hit Data 4, Slot 16	B27-GND	A27-Hit Data 4, Slot 10
C28-HIt Data 5, Slot 16	B28-Fiber D08	A28-Hit Data 5, Slot 10
C29-Hit Data 6, Slot 16	B29-GND	A29-Hit Data 6, Slot 10
C30-Hit Data 7, Slot 16	B30-Fiber D09	A30-Hit Data 7, Slot 10
C31-Data Valid, Slot 16	B31-GND	A31-Data Valid, Slot 10
C32-GND	B32-Fiber D10	A32-VEE, -5.2 Volts
C33-Left 26 MHZ Clock	B33-GND	A33-Right 26 MHZ Clock
C34-Hit Data 0, Slot 14	B34-Fiber D11	A34-Hit Data 0, Slot 12
C35-Hit Data 1, Slot 14	B35-GND	A35-Hit Data 1, Slot 12
C36-Hit Data 2, Slot 14	B36-Fiber D12	A36-Hit Data 2, Slot 12
C37-Hit Data 3, Slot 14	B37-GND	A37-Hit Data 3, Slot 12
C38-Hit Data 4, Slot 14	B38-Fiber D13	A38-Hit Data 4, Slot 12
C39-HIt Data 5, Slot 14	B39-GND	A39-Hit Data 5, Slot 12
C40-Hit Data 6, Slot 14	B40-Fiber D14	A40-Hit Data 6, Slot 12
C41-Hit Data 7, Slot 14	B41-GND	A41-Hit Data 7, Slot 12
C42-Data Valid, Slot 14	B42-Fiber D15	A42-Data Valid, Slot 12
C43-GND	B43-GND	A43-VCC, +5.0 Volts
C44-Hit Data 0, Slot 18	B44-Fiber Mux Enable	A44-Hit Data 0, Slot 8
C45-Hit Data 1, Slot 18	B45-Fiber User 2	A45-Hit Data 1, Slot 8
C46-Hit Data 2, Slot 18	B46-GND	A46-Hit Data 2, Slot 8
C47-Hit Data 3, Slot 18	B47-GND	A47-Hit Data 3, Slot 8
C48-Hit Data 4, Slot 18	B48-Event Address Valid	A48-Hit Data 4, Slot 8
C49-HIt Data 5, Slot 18	B49-Event Address Wrt En	A49-Hit Data 5, Slot 8
C50-Hit Data 6, Slot 18	B50-Event Address 0	A50-Hit Data 6, Slot 8
C51-Hit Data 7, Slot 18	B51-Event Address 1	A51-Hit Data 7, Slot 8
C52-Data Valid, Slot 18	B52-Event Address 2	A52-Data Valid, Slot 8
C53-VCC, +5.0 Volts	B53-Event Address 3	A53-GND
C54-Hit Data 0, Slot 22	B54-Event Address 4	A54-Hit Data 0, Slot 4
C55-Hit Data 1, Slot 22	B55-Event Address 5	A55-Hit Data 1, Slot 4
C56-Hit Data 2, Slot 22	B56-Event Address 6	A56-Hit Data 2, Slot 4
C57-Hit Data 3, Slot 22	B57-Event Address 7	A57-Hit Data 3, Slot 4
C58-Hit Data 4, Slot 22	B58-GND	A58-Hit Data 4, Slot 4
C59-HIt Data 5, Slot 22	B59-GND	A59-Hit Data 5, Slot 4
C60-Hit Data 6, Slot 22	B60-Fiber User 1	A60-Hit Data 6, Slot 4
C61-Hit Data 7, Slot 22	B61-Fiber User 0	A61-Hit Data 7, Slot 4
C62-Data Valid, Slot 22	B62-GND	A62-Data Valid, Slot 4
C63-VEE, -5.2 Volts	B63-Fiber Mux Control	A63-GND
C64-H53MHZ,Ø2 Clock, Slot 24,22,	B64-GND	A64-H53MHZ,Ø2 Clock,Slot 12,10,
C65-L53MHZ,Ø2 Clock, 20,18,16,14	B65-VTT, -2 Volts	A65-L53MHZ,Ø2 Clock 8,6,4,2

2. THEORY OF OPERATION AND OPERATING MODES

The sequencer module sits in the middle of the FASTBUS crate and communicates with the FASTBUS encoder modules via a special auxiliary backplane. Front panel connectors on the module connect via cables to the master controller to provide system clock, trigger commands, system RESET, errors and etc. The sequencer sends its data out via a rear mounted auxiliary card. Data may also be read via FASTBUS transfers.

2.1. Basic Operation

At power up and system reset time (i.e., the reset generated by the Master Timing Controller) the Sequencer Module will go into the initialization mode. In this mode CLK1 is not being driven onto the Auxiliary Backplane. The control logic waits for the Master Timing Controller to assert Write_Enable and then waits for the Master Timing Controller to assert Sync. When Sync is received the Sequencer will start driving CLK1 onto the Auxiliary Backplane. At this point the module is in normal running mode.

On receipt of an event address from the Master Timing Controller, the sequencer broadcasts it to the encoder modules via the special auxiliary backplane. Each encoder module extracts from memory, the 128 channels of hit information from that event and encodes it into an ordered list of 8-bit binary numbers. Each number in the hit list is comprised of a 7-bit binary number identifying the strip which had been driven above threshold by an ionizing particle and 1-bit to indicate if that microstrip had also registered a hit during the previous bucket (thus indicating that the current hit might actually be a residual from that bucket). All encoders do their encoding concurrently and send the data over individual data paths on the auxiliary backplane to FIFOs in the sequencer module.

As the sequencer begins receiving data from the encoder modules, it commences delivering it in order to its I/O ports. Beginning with the lowest numbered encoder FIFO, the data is transferred into the auxiliary card for transmission and into the FASTBUS buffer FIFO for readout. All the data from the 12 encoders are concatenated into a single ordered list. The ordering of the list is set by the encoder modules as lowest hit address to highest.

As each encoder FIFO produces an 'end of event token', the control logic moves on to the next. When the last FIFO has been emptied, the control logic adds the contents of the BLOCK COUNT/WORD COUNT and an IGNORE WORD if necessary to fill out the last 32-bit word in the data stream.

Referring to the block diagram, the sequencer consists of seven functional blocks.

2.1.1. Master Timing Controller Interface

- Receive 53 Mhz Clock. A function of the sequencer is to distribute clocks to the discriminator and encoder modules. A front panel connector on the sequencer receives a 53 Mhz clock from the master control module. This clock is exactly the same phase at all the sequencers. This clock has a fixed phase relationship to the signals arriving from the silicon microstrip detector. The sequencer derives two 53 Mhz clocks from this signal, CLK1 and CLK2. CLK1 is delayed from the input clock by a FASTBUS programmable amount and is driven to the discriminator modules via transmission lines on the auxilary backplane. CLK2 is delayed from CLK1 by a switch selectable amount and is driven to the encoder modules via transmission lines on the auxilary backplane. A divide-by-two version of CLK2 called CLK3 is bussed by the sequencer to the encoder modules to be used as a data readout clock.

- Receives system WRITE ENABLE signal from the Master Timing Controller.

- Receives SYNC signal from the Master Timing Controller.

- Receives RESET signal from the Master Timing Controller.

- Receives EVENT ADDRESSES and ADDRESS VALID from the Master Timing Controller.

- Sends 'sequencer ready' signal to the Master Timing Controller (SEQ_READY).

- Sends 'encoders ready' signal to the Master Timing Controller (EN_READY).

- Sends ERROR signal to the Master Timing Controller. This signal alerts to an encoder FIFO overflow, encoder out of sync and others to be added. FASTBUS can interrogate the Error Status Register to evaluate the error. Event size greater than 255 is not an error condition.

2.1.2. Encoder Interface

- Contains the FIFOs which receive data from each of the 12 encoder modules.
- Deposits 'end of event' token into each FIFO after all data has been received from the encoder.
- Sends RESET signal to the Encoder modules on command of the front panel signal.
- Sends WRITE ENABLE signal to the Encoder modules.
- Sends EVENT ADDRESSES and ADDRESS VALID to the Encoder modules.
- Sends SYNC to the Encoder modules delayed by CLK1 plus CLK2 delay.
- Receives SYNC ERROR from the Encoder modules.
- Receives EVENT DATA and DATA VALID from each Encoder module.

2.1.3. FASTBUS Interface

- Allows a FASTBUS master to read/write the control and status register.
- Allows a FASTBUS master to read/write the PLANE/ENCODER RAM.
- Allows a FASTBUS master to read/write the BLOCK COUNT/WORD COUNT.
- Allows a FASTBUS master to read/write the CLK1 delay value and read the CLK2 delay value.
- Allows a FASTBUS master to read the EVENT BUFFER FIFO.

2.1.4. Auxiliary Interface

The auxiliary interface is a high speed outlet for event data. Unless disabled by a CSR0 bit, the sequencer will drive data out this port directly, bypassing the need for software intervention to read out data over FASTBUS. This interface is designed so that it can drive an RS-485, ECLine or fiber optic auxiliary module. The auxiliary module provides the special interface requirements of the link. The signals provided by the sequencer to the fiber optic interface are listed below. The other types of interface modules ie. RS-485 and ECLine will have to be designed to utilize these same signals.

The auxiliary port interface includes the following signals. See the appendixes for the actual pin numbers.

- D0-D15. 16 multiplexed data lines to the auxiliary module.
- MUX ENABLE to the auxiliary module.
- MUX CONTROL to the auxiliary module.
- WAIT signal from the auxiliary module.
- CLOCK from the auxiliary module.
- ERROR signal from the auxiliary module.
- USER 1- USER 3. Three user defined signals.

2.1.5. Block/Word Counters

The BLOCK COUNT/WORD COUNT is a 16-bit entity which is normally used as the last valid word of a concatenated event. It consists of a 4-bit type code, a 4-bit counter for use as a BLOCK COUNT and an 8-bit counter for use as a WORD COUNT. The 4-bit type code is a binary '1000' placed in bit position 15-12 to identify the data as being the BLOCK COUNT/WORD COUNT of an event as opposed to data. The 4-bit BLOCK COUNT is intended to be used as a processed event identifier to provide parallel synchronization among the 12 FASTBUS crates. The 8-bit WORD COUNT is intended to be used to count the number of 'hits' in each event. The counters are cleared and incremented individually by the control logic, but their combined contents are always output as a 16-bit word. The counters may be written and read via FASTBUS for diagnostic purposes.

2.1.6. FASTBUS Event FIFO

FASTBUS output will be via a single event FIFO (32 X 512). Unless disabled by a CSR0 bit, a full event including LAST WORD and IGNORE WORD if necessary will be loaded into this FIFO as it is being sent out the auxiliary port. Output to FASTBUS will be in 32 bit words. A 32 bit word count will be generated and inserted at the end of the event record exactly as is done for the auxiliary port. If the FASTBUS port is being used for readout, the sequencer will not start to read a second event out through the auxiliary port until the event in the FASTBUS FIFO has been completely read out. A CSR0 bit will disable the FASTBUS event FIFO to allow data output via the auxiliary port without regard to the status of the event FIFO.

2.1.7. Control Logic

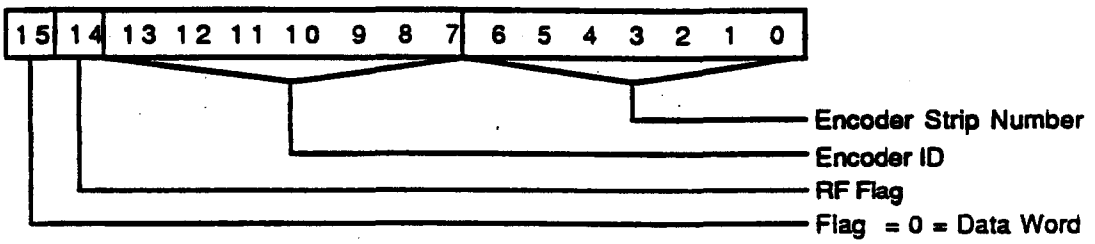
The control logic controls the input fifos, block and word counters, data identifier RAM, auxilliary port interface FIFO and FASTBUS event FIFO. This logic is implemented with programmable array logic devices (PALs).

2.2. Addressing Modes

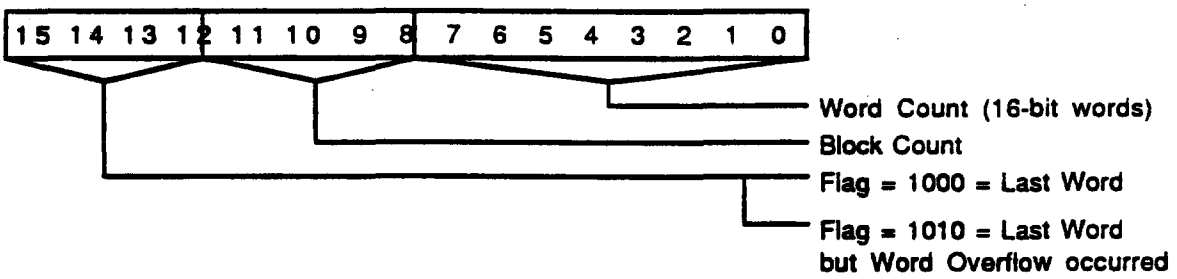
2.2.1. Data Transfer Description and Rates

The sequencer is designed to transfer silicon strip detector data over the auxiliary fiber optic link at peak instantaneous rates approaching 80 nsec per 32-bit word. The sequencer is designed to support block transfer data reads from its FASTBUS event buffer at peak instantanaeous rates approaching 150 nsec per 32-bit word. Whether data is output via the auxiliary port or read via the FASTBUS slave interface, it will always be in 32-bit words and each 32-bit word will always be comprised of two 16-bit words. The two 16-bit words words will each be of one of three types. The three types are illustrated below:

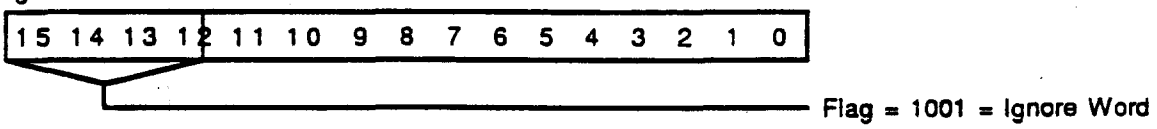
Data Word Format



Last Word Format



Ignore Word Format

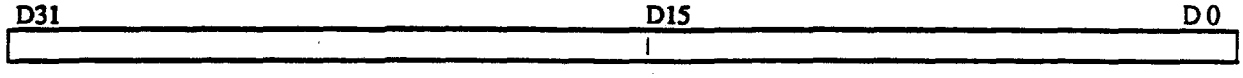


2.2.2. Internal Control, Status Registers, and Bit Descriptions

There are several registers on the Sequencer that are accessible via FASTBUS for initialization, performance monitoring, or diagnostics. These registers are described in the following subsections.

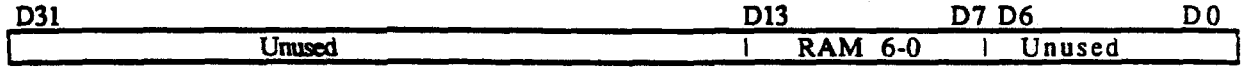
2.2.2.1. FASTBUS mandatory CSR0

CSR Space Hex Address		Description
0000 0000	CSR 0	Bit descriptions:
	Read , Write	
	Bit <6>,<22>	Enable/Disable sequencer loading of FASTBUS event FIFO.
	Bit <7>,<23>	Enable/Disable sequencer loading of auxiliary interface FIFO.
	Bit <8>,<24>	Enable/Disable overflow truncation.
	Bit ,<30>	Reset sequencer.
	Bits<31:16>,	Manufacturer's ID and device type.



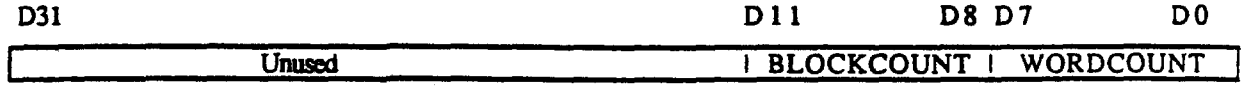
2.2.2.2. Read/Write The PLANE/ENCODER RAM

CSR Space Hex Address	Description
C000 0000 - C000 000B	There are 16 RAM locations accessible by FASTBUS of which 0 thru 11 are used to hold the 7-bit plane/encoder identifying data. 12 locations are required, one for each encoder FIFO. Although FASTBUS read/writes utilized 32-bits, only 7-bits are used for this operation and the rest are undefined. This memory may by written and read via FASTBUS not only for setting up the module but for diagnostics as well. This is only a 7-bit RAM. The MSB will always read '0' because this is a tag that differentiates DATA WORDs from LAST WORD or IGNORE WORD in the data stream.



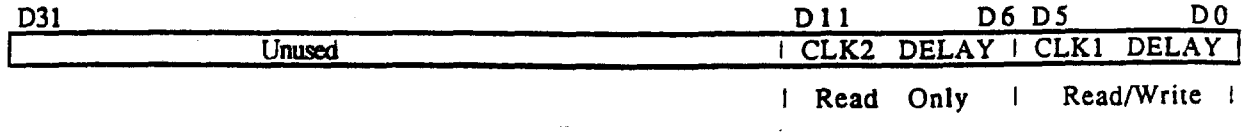
2.2.2.3. Read/Write The BLOCK COUNT/WORD COUNT

CSR Space Hex Address	Description
C000 0010	The BLOCK COUNT/WORD COUNT counters may be written and read via FASTBUS for diagnostic purposes.



2.2.2.4. Read/Write The CLK1 Delay Value, Read The CLK2 Delay Value

CSR Space Hex Address	Description
C000 0011	The CLK1 delay value is FASTBUS read/writeable, 0.5 ns per step, 64 steps. The 53 Mhz clock received via the front panel is delayed by this value to produce CLK1. This value is also added to the switch settable delay for CLK2 to produce the total delay value for the SYNC signal. The CLK2 delay value is set by hardware DIP switches on the card itself. The setting of those switches are read back in bits D11 through D6.



2.2.2.5. Read-Only Error Status Register

CSR Space Hex Address	Description
C000 0012	FASTBUS readable, bit D0 is set if a Sync Error has been detected by a Delay/Encoder. Bits D1 through D12 correspond to encoder FIFO channels 1 through 12. If one of these bits is set then its corresponding FIFO has overflowed.

D31	D12	D1	D0
Unused	Encoder Fifo Overflow Flags		Sync Err Flag

2.2.2.6. Read The EVENT BUFFER

In order to minimize overhead, the FASTBUS master may attempt to read from the event buffer even before it recieves a trigger and will be forced to WAIT only until data begins streaming into the FIFO. The master must not allow itself to time out while waiting for data. At the end of event processing, the sequencer will load the LAST WORD which contains the BLOCK COUNT/WORD COUNT; and an IGNORE WORD if necessary into the FIFO. After this word is read by the FASTBUS master, SS=2 would terminate the transfer.

The EVENT BUFFER is a 32-bit wide X 512 deep FIFO that holds a single event. The 32-bit word is made by packing two 'hits' which are normally 16-bits. To speed up FASTBUS transfers, this FIFO will be read out by a FASTBUS master using block transfer reads. There is a CSR0 bit to disable the event buffer if the event buffer is not to be used. This bit prevents the control logic from waiting for the buffer to be ready or strobing data into the buffer. This feature allows a FASTBUS master to sample events for histogramming or other reasons.

Data Space Hex Address	Description
0000 0000	Block transfer read of the event FIFO. Generate a primary address cycle to DATA space followed by block transfer reads. There is no secondary address cycle. The data transfers will be stalled by WAIT during the data cycle until event data begins streaming into the FIFO. The overhead involved with a 68020 powered FASTBUS Smart Crate Controller is as follows: 1) 300 nsec for the move instruction that executes a FASTBUS primary address cycle. 2) 300 nsec for the move instruction that starts the execution of a block transfer. 3) About 150 nsec per 32-bit word of data during the block transfer. The overhead from 1 and 2 is not incurred if the master executes the primary address cycle before a trigger is even received. The last word in the FIFO will have the 'LAST WORD' flag, the BLOCK COUNT and the WORD COUNT. If the number of words is odd, the last word will end up in the LS byte position of the 32-bit longword and an IGNORE WORD with a binary flag of '1100' will end up in the MS word. If the number of 32-bit words in the FIFO is even, the LAST WORD will end up in the MS word of the last 32-bit longword.

Data words:

D31	D15	D0
n+1 Data Word		n Data Word

Last word (if odd number of words):

D31	D15	D0
Ignore Word		Last Word

Last word (if even number of words):

D31	D15	D0
Last Word		n+x Data Word

2.2.3. Error Responses

The ERROR signal is sent to the Master Timing Controller when a Sync Error is detected of when and encoder FIFO overflows. The system software can then read the Error Status Register to determine specifically which of the encoder FIFOs overflowed or if a Sync Error had been detected by this sequencer.

2.3 Overview of the Sequencer Data Pipeline Operation

The twelve channel Delay/Encoder interface circuit control logic idles until it receives an Address Valid (ADVAL) pulse from the MTC. The Sequencer synchronizes the ADVAL signal and passes it on to all twelve Delay Encoder Modules. The Sequencer also uses ADVAL to start the Sequencer's Delay Encoder Timeout Counter. Upon receiving the ADVAL signal each Delay Encoder Module, that has data to send, will place an 8 bit data word on it's private Hit Data lines and activate it's own Hit Data Valid (DVAL) signal. There are twelve Encoder Fifo Circuits on the Sequencer which correspond to the twelve Delay Encoders. Encoder Fifo Circuits which receive a DVAL from their corresponding Delay Encoder will begin to write the 8 bit hit data words into their fifos (these will be referred to as the Encoder Fifos). On every positive transition of the CLK3 signal Delay Encoders with data to send will issue new 8 bit data words and the corresponding Sequencer Encoder Fifo Circuits will write this data into their fifos. When a Delay Encoder has no more data to send it deactivates DVAL. The Sequencer's Encoder Fifo Circuits see DVAL go away and they then write an End of Event Word (EOE word) into their fifos. The Encoder Fifos are 9 X 512. Bit 8 of the Encoder Fifo (the MSB) is high while hit data is being written and is low when we write the EOE word (I will refer to this bit from now on as the EOE tag). Encoder Fifo Circuits which do not receive DVAL signals before the Timeout Counter runs down (240ns after ADVAL received from the MTC will write an EOE tag into their fifos.

After the Timeout Counter runs down a trigger counter is incremented. Because the value contained in this trigger counter is no longer zero the Sequencer's Fifo Array Control Unit will begin a readout cycle by issuing a PRELOAD signal and decrementing the trigger counter. (Note that during a readout cycle the Sequencer will probably receive new triggers and the Sequencer's Encoder Fifo Circuits will be performing fifo writes concurrently with the readout cycle). The Preload signal causes all twelve Encoder Fifo Circuits to perform one data read from their fifos. Each Encoder Fifo Circuit places the 9 bit data word from this read in its output register. The Fifo Array Control Unit Examines the twelve bit vector comprised of the EOE tags from each Encoder Fifo Circuit. The Fifo Array Control Unit then reads out the first Encoder Fifo which has hit data. The data which is read out is placed in the P2 (pipeline stage 2) data register. When the Fifo Array Control Unit detects an EOE tag it immediately begins reading the next Encoder Fifo that contains hit data. There are no gaps in the pipeline data stream caused during this process. Channels which timed out are not read out.

Data from the P2 data register and a Delay Encoder ID from the Encoder ID RAM form a 16 bit word. The first 16 bit word constructed is written to the lower 16 bits of the 32 bit DMUX register. The next 16 bit data/ID word is written to the upper 16 bits of the DMUX register. 16 bit data/ID words continue to be alternately written to the lower and upper halves of the DMUX register. When the Fifo Array Control Unit detects that all event data words have been read out of the Encoder Fifos it issues the DONE signal to the Pipeline Control State Machine. This state machine will do one of two things. If there were an even number of data/ID words written to the DMUX register the state machine will cause a Last Word to be written to the lower 16 bits of the DMUX register and then it will cause an Ignore Word to be written to the upper 16 bits of the DMUX register. If there was an odd number of data/ID words written to the DMUX register, the state machine will cause a Last Word to be written to the upper 16 bits of the DMUX register. The last word is composed of the 8 bit word count, the 4 bit block count, and the 4 bit Last Word Identifier. The lower twelve bits of the Ignore word are undefined and the four remaining bits contain the Ignore Word ID Tag.

The 32 bit words constructed in the DMUX register will be written to either one (or both) of the 32 X 512 output fifos. One of the output fifos is called the FASTBUS Event Buffer; the other is known as the Auxiliary Event Buffer.

When the PIPEHOLD signal becomes active the data in the Sequencer Data Pipeline will be frozen in place and the current state of all state machines is preserved. The PIPEHOLD signal does not restrict the flow of data from the Delay Encoder Modules to the Sequencer Encoder Fifos. Only the sequencer's internal data pipeline is frozen. Below you will learn about situations which cause PIPEHOLD to become active.

If the FASTBUS Event Buffer is enabled but the Auxiliary Event Buffer is not enabled, DMUX register data will only be written to the FASTBUS Event Buffer. In this mode, once one complete event has been written to the FASTBUS Event Buffer, further writes are disabled until the previous event has been completely read out from the FASTBUS Event Buffer. In this mode we also freeze the Data Pipeline after each complete event has

been written into the FASTBUS Event Buffer. The pipeline remains frozen until the event in the FASTBUS Event Buffer has been completely read out. Note that, when in this mode, the pipeline will also be frozen if the FASTBUS event buffer becomes half full.

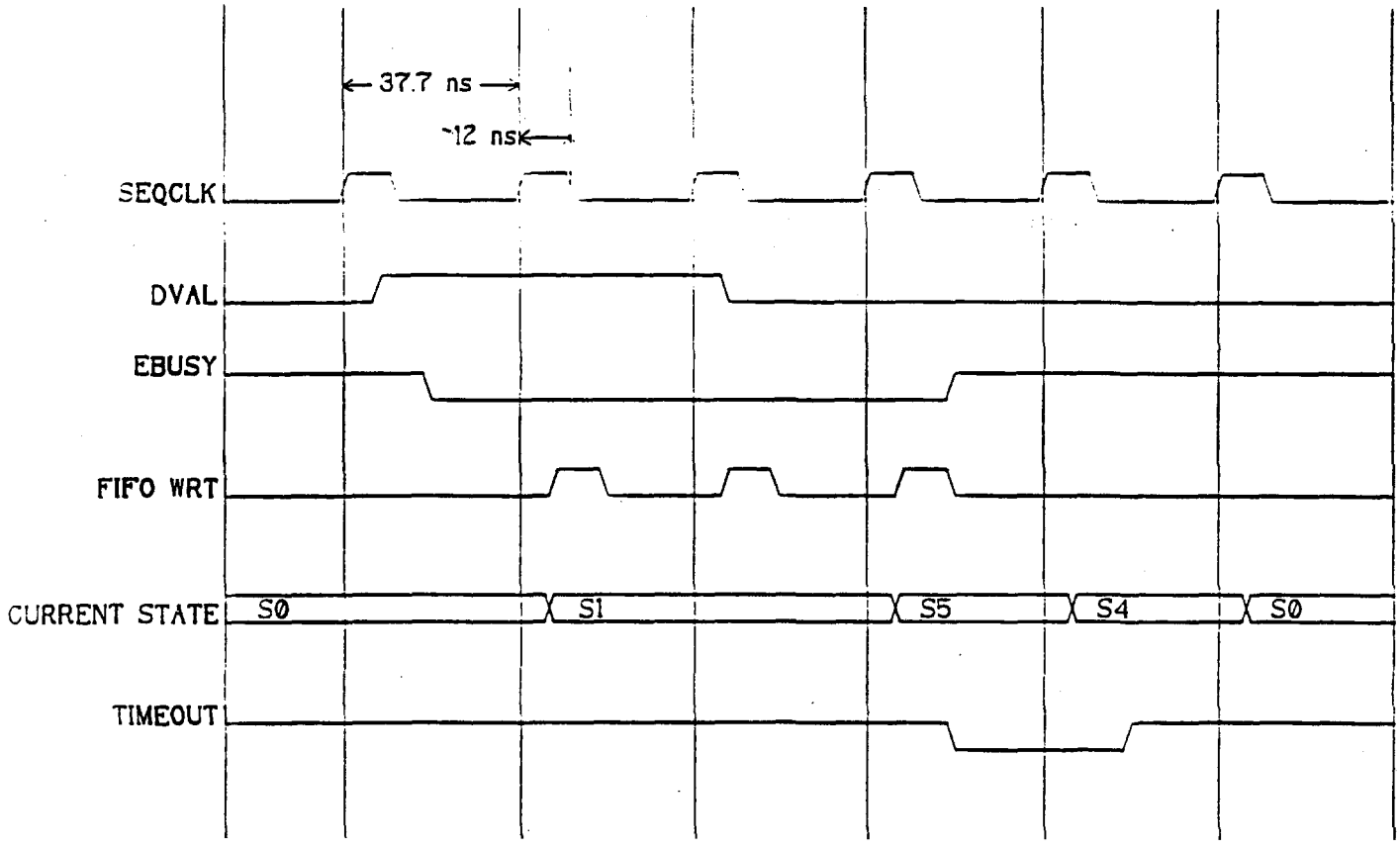
If both the FASTBUS and Auxiliary Event Buffers are enabled data from the DMUX register will be written to both event buffers. In this mode, once one complete event has been written to the FASTBUS Event Buffer, further writes to the FASTBUS Event Buffer are disabled until the previous event has been completely read out from the FASTBUS Event Buffer. Note that the presence of an event in the FASTBUS Event Buffer will NOT cause the pipeline to freeze. Data will continue to flow into the Auxiliary Event Buffer at full speed. This means that the FASTBUS Event Buffer will only receive a portion of the total events which travel through the pipeline. If the Auxiliary Event Buffer becomes half full the pipeline will freeze. It will remain frozen until the Auxiliary Event Buffer is no longer half full. In this mode, if the FASTBUS Event Buffer becomes half full the pipeline will be frozen. This should almost never happen since: 1) we only allow one event to exist in the FASTBUS Event Buffer and 2) most single events will contain less than 256 events. Thus Auxiliary data path performance will not be affected by the slower FASTBUS Data Path.

If only the Auxiliary Event Buffer is enabled, events will only be written to the Auxiliary Event Buffer. PIPEHOLD will only be activated in this mode when the Auxiliary Event Buffer becomes half full.

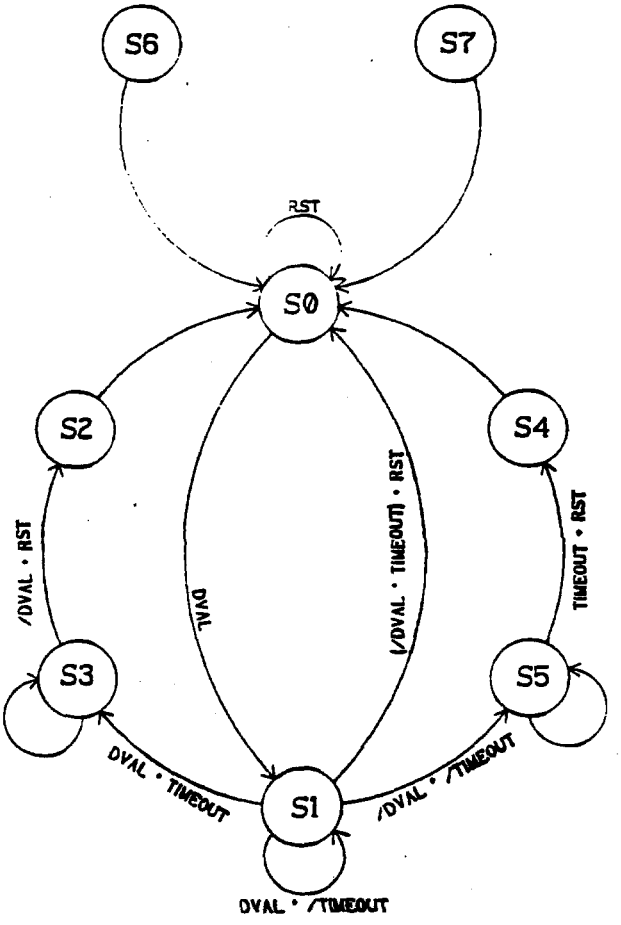
2.3.1 Encoder FIFO Circuits

There are twelve Encoder Fifo Circuits with one corresponding to each of the twelve Delay/Encoders. Each of these circuits contains an Encoder Fifo (75C01), a control pal and an output register section. The output register contains a data register pair (two 74AS574 8-bit registers and delay line) and a 74F74 1-bit register. The function of the control pal is to control the gating of the fifo read and fifo write signals and also to latch the occurrence of a FIFO full error condition. After Delayed Reset (DLYRST) is removed both the fifo write and fifo read signals go low. We use a delayed version of reset because fifo timing parameters require fifo reset to be inactive for 15ns before the read or write signals are dropped low. As shown in the timing diagrams below EBUSY becomes active after DVAL goes active and SEQCLK goes low. Whenever EBUSY is active, SEQCLK is allowed to flow out of the pal and clock data into the fifo. Once active EBUSY stays active as long as DVAL is active. When DVAL goes inactive the state machine in this pal will allow EBUSY to stay active for one more clock tick to allow an EOE tag to be written into the Fifo. After the EOE tag is written and SEQCLK has gone low we allow EBUSY to go inactive. By using SEQCLK as a qualifier for both the activation and deactivation of EBUSY we eliminate glitches on the fifo write line. Besides functioning as a gating control signal EBUSY, when active, is also used to remove the Encoder Ready signal which will be sent to the MTC. The operation of the control pal state machine is shown in the state diagram below. In State S0 the state machine waits for a DVAL signal from the Delay Encoder. State S0 is also the only state in which we allow an EOE tag to be written due to a time out. States S1 and S3 hold the fifo write gating signal (EBUSY) active until DVAL is inactive. If DVAL goes inactive before time out occurs the state machine enters S5 and waits for the time out signal to occur. This prevents us from writing 2 EOE tags into the fifo (one for the event data and one because time out occurred). States S2 and S4 are present to insure that all state transitions are 1-bit transitions. This will help to reduce noise that may occur because there are twelve of these circuits operating at the same time. States S6 and S7 are not used and are defined only for the purpose of trap state prevention. The operation of the fifo read gating control is similar to that of fifo write. Refer to the pal file ENCEN.ABL included with this document for more information on this chip.

Encoder FIFO Write Timing Diagram



Encoder FIFO Circuit State Machine



When an 8 bit event data word is read from the Encoder Fifo it is placed in the input register of the data register pair. The reason two data registers are used is to eliminate clock skew induced hold time violations when passing data between pipeline stages. In this circuit, and in all other pipeline stages, a write signal derived from the Sequencer's on board 25 Mhz clock writes data into the input register of the register pair. This signal delayed by 12ns is used to write the output register. This insures that data being sent to a subsequent pipeline stage will not change before that subsequent stage can "grab" the data. Bit 8, the MSB (the EOE Tag bit) of the 9 bit data word, which is read from the Encoder Fifo, is placed in the 74F74.

2.3.2 FIFO Array Control Unit

The Fifo Array Control Unit is comprised of clock distribution circuitry, the Delay Encoder Timeout Counter (ARTMO.ABL), the trigger counter (TRIGCNT.ABL), the array control state machine (ARLOAD.ABL), and readout channel selection logic (RDEN1.ABL, RDEN2.ABL, and ARIDAD.ABL).

The delay encoder timeout counter pal is a very simple state machine. This state machine sits in a state waiting for ADVAL. When ADVAL occurs it starts a binary count. After six rising edges of SEQCLK this counter enters the timeout state. As soon as the SEQCLK goes low the encoder timeout signal ENCOTMO is made active. ENCOTMO is sent to all Encoder Fifo Circuits whether they need it or not. After timeout is sent, this chip issues the TRINC signal to increment the trigger counter. During the timeout count this chip issues a signal called EBUSY13 which causes the sequencer to tell the MTC that the Encoders are not ready. We need this feature to account for the case where none of the Delay Encoders have data to send. We don't want the MTC to send the Sequencer another ADVAL until the current timeout count has finished.

The trigger counter pal's purpose is to tell the array control state machine whether or not the there are events in the Encoder Fifos to be read out. There are events to be read out if trigger counter's NOTEMP signal is active. The NOTEMP signal is active whenever the trigger count value is not equal to zero. The trigger counter is incremented if the increment signal (TRINC) is active and the decrement signal is not active. Note that the TRINC signal is issued after there is something in every encoder fifo (data or EOE tag) from the current trigger being encoded by the Delay Encoder. The trigger counter is decremented if the decrement signal (DEC) but not the increment signal (TRINC) is active. While the sequencer's data pipeline is processing an event, the trigger counter will allow up to 30 triggers to be received before activating the HF signal which deactivates the Sequencer Ready signal. If the MTC ignores the fact that the Sequencer Ready signal has gone away the trigger counter will be able to receive 33 more triggers before the ERROR signal is sent to the MTC.

If the pipeline is not frozen, the pipeline control state machine has completed the previous event readout cycle and the NOTEMP signal is active the array control state machine will issue the PRELOAD command to the Encoder Fifo Circuits. The array control state machine will then activate the event readout enable signal (EVENTEN) and keep it active until an active DONE signal is received from the readout channel selection logic. When DONE is received if no errors (full encoder fifo or trigger counter overflow) have occurred the array control state machine will return to the hold state and wait for the NOTEMP signal to become active again. If, when DONE is received, an error has occurred the array control state machine will enter the ERROR state and remain there until the sequencer is reset.

The readout channel selection logic decodes the 12-bit EOE tag vector and determines which of the twelve Encoder Fifo Circuits (channels) will be read out. The readout channel selection logic also uses the EOE tag vector to produce a Encoder ID address.

2.3.3 Data Word Construction Unit

The Data Word Construction Unit is comprised of the P2 register (two 74AS574 registers), the Block Counter (74F569), the Word Counter (74AS867), counter control logic (BWCOUNTC.ABL), the Encoder ID Ram (CY7C190), the Flag Register (74AS244), the Pipeline Control State Machine (BWSTATEM.ABL), the Pipeline Control State Decoder (BWSTDCD.ABL), write signal gating control (BWGATEC.ABL and BWHOLD.ABL), the DMUX Register (eight 74AS574 registers) and as always, miscellaneous glue logic (BWRAMCON.ABL).

The P2 Register is a back to back register pair comprised of two 74AS574 8-bit registers. The P2 input register receives data over the 8 bit EFDATA bus from any one of the Encoder Fifo Circuits. The P2 output register, when enabled, places its data on the 16 bit DATA bus. Note that after flowing through the P2 Register bits 1-7 of the EFDATA bus are transferred to bits 0-6 of the DATA bus. Bit 0 (the previous hit bit) of the EFDATA bus is transferred to bit 14 of the DATA bus.

The Block Counter is a 4-bit counter that counts the number of events (triggers) the sequencer has processed. The Word Counter is an 8-bit counter that counts the number of 8-bit data words contained in an event. Both the word and block counters can be written to and read from by FASTBUS. Care should be taken to not write to

the counters during data taking operations. **Programmers should avoid writing FF(Hex) to the word counter since this puts the counter into an overflow condition.**

The Pipeline Control State Machine will, upon sampling an active EVENTEN signal, enter the DATLO state. This causes the Pipeline State Decoder to enable the P2 output register and the ID Ram so they can place their contents on the DATA bus. While in this state the state machine allows the DMUX register gating circuitry (BWGATEC.ABL) to pass a write strobe (DMLOW) to the low half of the DMUX register. The next clock tick (if DONE has not occurred) will cause the state machine to enter the DATHI state. In this state the P2 data register and the ID Ram are still enabled. The gating circuitry will now, however, be allowed to pass a write strobe to the upper half of the DMUX register. The state machine will switch back and forth between the DATLO and DATHI states until it receives an active DONE signal or, if truncation is enabled, a word counter overflow occurs. If an active DONE signal is received and the state machine is in the DATHI state the state machine will enter the LASTLO state. The LASTLO state enables the Block Counter and Word Counter output registers for output onto the DATA bus. The LASTLO state also causes the Flag register to put a 4 bit Last Word Identifier on the data bus. These items are written into the low half of the DMUX Register. The next rising edge of the 25 Mhz clock causes the state machine to enter the IGNORHI state. In this state the Flag Register will place a 4 bit Ignore Word Identifier on the DATA bus. If the state machine is in the DATLO state when DONE occurs, the state machine will enter the LASTHI state and the Lastword (Block and Word Counts and Last Tag) will be written into the upper half of the DMUX register.

If event truncation is enabled, a word counter overflow (more than 255 data words in a single event) occurs and DONE is not active the state machine enters the TRUNCHI state and a Lastword with a special 4 bit truncation tag is written to the upper half of the DMUX register. After the Lastword and possibly the Ignore word are written to the DMUX register, the state machine enters the Flush state. It will stay in the FLUSH state until EVENTEN goes away. Under most conditions EVENTEN will have gone away before the state machine enters this state and the state machine will exit this state on the next clock tick. If the state machine enters this state because of a word counter overflow it stays in this state until the extra data from this overflow event has been flushed from the Encoder Fifos. After leaving the FLUSH state the state machine will always enter the SET COUNT state in which the Word Counter will be cleared and the Block counter will be incremented. The state machine leaves the SET COUNT state on the next clock tick and goes to the WAIT state towait for EVENTEN to occur.

The write pulse gating logic is designed to provide glitch free write pulses (at the correct time) to the P2 Register, the DMUX register (low and high halves), and to the Auxiliary and FASTBUS Event Buffers.

3. INPUT/OUTPUT SPECIFICATIONS

3.1. Communication Interfaces

3.1.1. Fiber Optic Auxiliary Port

The sequencer auxiliary port can be fitted with an auxiliary module which provides a high speed fiber optic link. Details about this auxiliary module is found in another document by the Detector Electronic Systems Group. Data delivered by the sequencer fiber optic port will have the same format as data delivered by the sequencer FASTBUS event buffer interface. However, the fiber optic link protocol requires some additional control operations to support handshaking, error reporting, etc.

There is a CSR0 bit which may be used by a FASTBUS master to disable the auxiliary port if the port is not to be used. This bit prevents the control logic from waiting on a port ready signal and from strobing data into the auxiliary port.

3.1.1.1. Communication Protocol

The sequencer communicates with the auxiliary card in the following fashion:

When the Sequencer has data in its Auxiliary interface FIFO the sequencer drives Auiliary output data lines 0-15 with the first 16-bit data word and asserts MUX ENABLE. The auxiliary card provides a clock signal to the sequencer and all transfers are synchronized with this clock. Subsequent 16-bit data words are transferred by alternating the level of the MUX CONTROL signal in order to tell the auxiliary card whether the lower 16 bits or the upper 16 bits of the 32 bit data word is currently being sent.

First and all subsequent normal data words (transferred 16 bits at a time, MUX CONTROL is toggled):

D31	D15	D 0
n+1 Data Word		n Data Word

Last word (if odd number of words):

D31	D15	D0
Ignore Word	Last Word	

Last word (if even number of words):

D31	D15	D0
Last Word	n+x Data Word	

4. SYSTEM, MODULE, CIRCUIT, OR CHIP DIAGNOSTICS

4.1. Hardware

4.1.1. Sequencer Test Card

The SSD Sequencer Test Card (STC) is a FASTBUS card that is designed to reside in a Delay/Encoder slot of an SSD crate. The STC possesses a FASTBUS interface and can be controlled and read out by a FASTBUS master. In order to test the SSD Sequencer the STC simulates the Sequencer interface of the SSD Master Timing Controller (MTC) and the Sequencer interface of the SSD Delay/Encoder (D/E).

The twelve channels of the Sequencer can be individually tested by moving the STC to each of the twelve D/E slots and repeating the test procedure. However, provisions are included for a ribbon cable connector that can be used to control a twelve channel Auxiliary card which can send data to all twelve Sequencer channels.

The STC has a 1024 by 9 Event Data RAM which is used to store the data that will be sent to the Sequencer when a cycle is initiated. Although 128 is the largest number of hits that the D/E is capable of sending out, having 1024 pieces of data available allows us to test the Sequencer's ability to handle a FIFO overflow.

Locations addressable from FASTBUS:

C000 0000	MTC Event Address Offset register which stores the eight bit value of the Event Address Offset that will be issued to the Delay/Encoder. Read/write.	
C000 0001	Bits <0>, <16>	Assert/deassert the MTC Write_Enable signal.
C000 0001	Bit <1>	Assert the MTC Reset signal.
C000 0001	Bit <2>	Assert the STC Reset signal.
C000 0001	Bit <3>, <19>	Start a cycle by asserting Address_Valid.
C000 0002		Delay/Encoder simulator Event Address Offset latch. Read/Write.

0000 0000 to 0000 03FF in Data Space.	1024 RAM locations for Event Data, RF Flag, and an additional bit that indicates to the STC's circuit that the end of event has been reached. 1024 by 9 is the total.
0000 0400 to 0000 07FF in Data Space.	1024 RAM locations that reside on the twelve channel D/E test card.

RIBBON CABLE SIGNALS FOR THE TWELVE CHANNEL DELAY/ENCODER TEST CARD

EVENT_DATA(0:8)	Seven bit Event Data, one bit RF Flag, one bit Last Strip flag
ED_ADDR(0:9)	Address for the Event Data
DATA_STRB	Event Data Data Strobe
DATA_RD	Data Read/Write Control Signal
RUN_MODE	Asserted for running the test. Deasserted for writing or reading Event Data.
DTC_PRESENT	This signal will indicate to the STC that the twelve channel Delay/Encoder Test Card is present and connected to the STC via the ribbon cable.

RIBBON CABLE SIGNALS FOR THE AUXILIARY INTERFACE TEST CARD

AUX_DAT(0:15)	Auxiliary Data
AT_FIFO_READ	Auxiliary Test Card FIFO Read Signal
AT_FIFO_EMPTY	Auxiliary Test Card FIFO Empty
COM_STRB	Command Strobe
COM_RD	Command Read/Write Control Signal

COMMAND(0:3)	Command Bits
ATC_PRESENT	This signal indicates to the STC that the Auxiliary Interface Test Card is present and connected to the STC via the ribbon cable.

4.1.2. Operating Instructions

Load the Event Data RAM into the STC via FASTBUS.

Issue an MTC RESET with a FASTBUS write to the STC.

Generate the SYNC signal with a FASTBUS write to the STC.

Assert WRITE_ENABLE with a FASTBUS write to the STC.

Output an Event Address Offset with a FASTBUS write to the STC.

Assert the MTC ADDRESS_VALID a FASTBUS write to with the STC.

The Delay/Encoder emulator circuit should then transfer data to the Sequencer.

Read the Event Data out of the FASTBUS Event Buffer FIFO with the FSCC and compare it with test data stored in the FSCC's memory.

Read the Event Data out of the Auxiliary Interface Test Card's FIFO via FASTBUS through the Test Control Card and compare the data to test data stored in the FSCC's memory.

Read the Delay/Encoder Event Offset Address Latch and compare the value to its test value stored in the FSCC's memory.

4.2. Software

4.2.1. Diagnostic Test Description

A detailed description of the Sequencer Diagnostic Software is shown in the Fermilab Program Note number PN 434 *Silicon Strip Detector System Single Board Diagnostic Tests.*

Software to test the funtionality to the sequencer includes the following tests:

Write and read the CLK1 delay value via FASTBUS.

Read the CLK2 delay value via FASTBUS.

Write and read the twelve encoder identification RAM locations from FASTBUS.

Write and read the block counter from FASTBUS.

Write and read the word counter from FASTBUS.

Write and read CSR 0 from FASTBUS.

Read the Error Status Register from FASTBUS.

Silicon Microstrip Detector Readout System

Sequencer Module

Bob DeMaat, Marc Larwill, Andy Romero

****

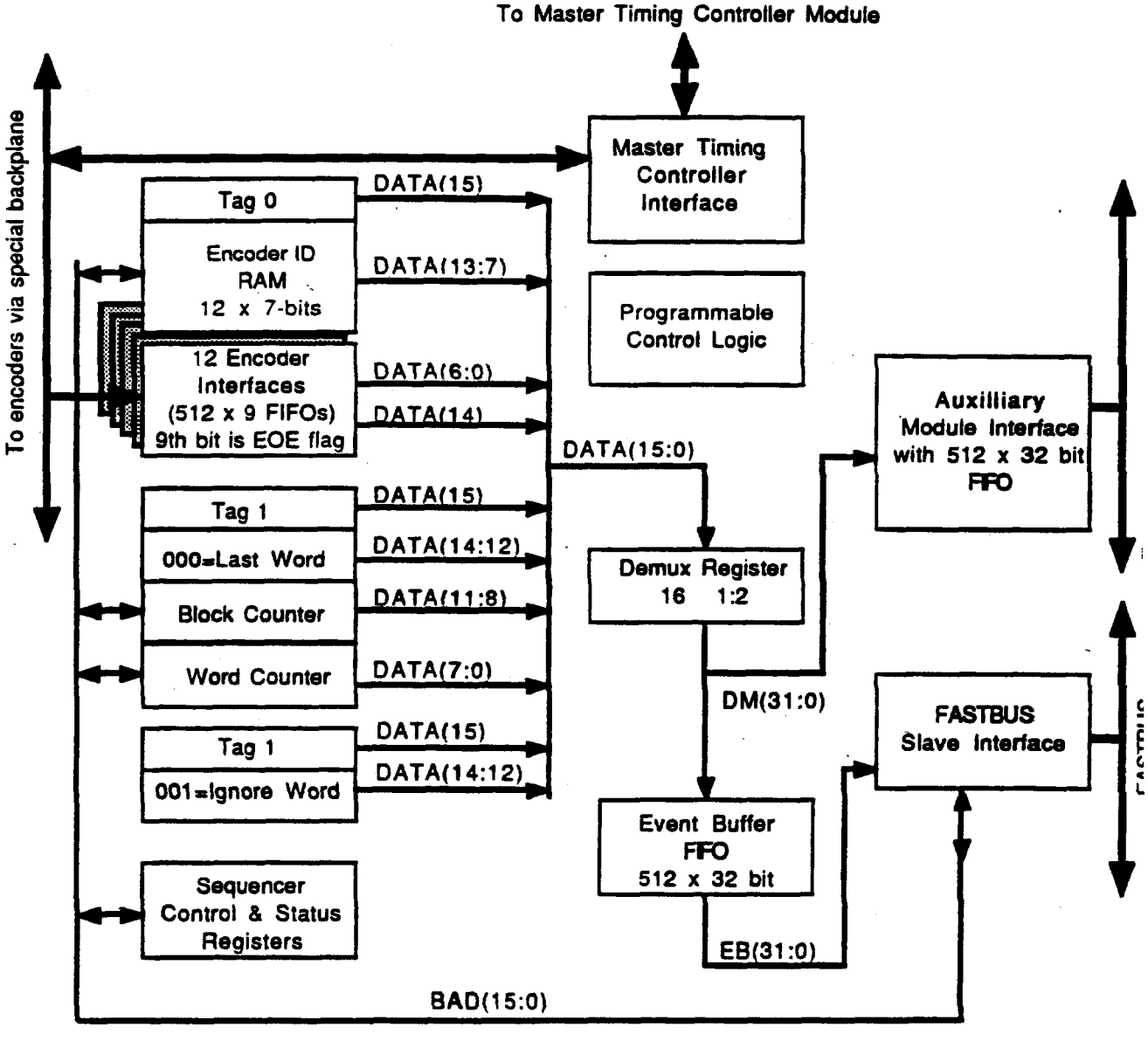
Preliminary Specification - Always check header date and use latest update - Direct all questions, comments and observations to Bob DeMaat.

****

1. GENERAL INFORMATION

1.1. Purpose

This document describes a device to be known as the "Silicon Strip Detector Readout System Sequencer Module" hereafter referred to as the sequencer. As one component of a larger system, this module will accept silicon strip detector data from the 12 encoder modules in its FASTBUS crate, combine the data and output it to other system components via FASTBUS or via an auxiliary link such as RS-485, ECLine or fiber optics. The general block diagram of the sequencer is as follows:

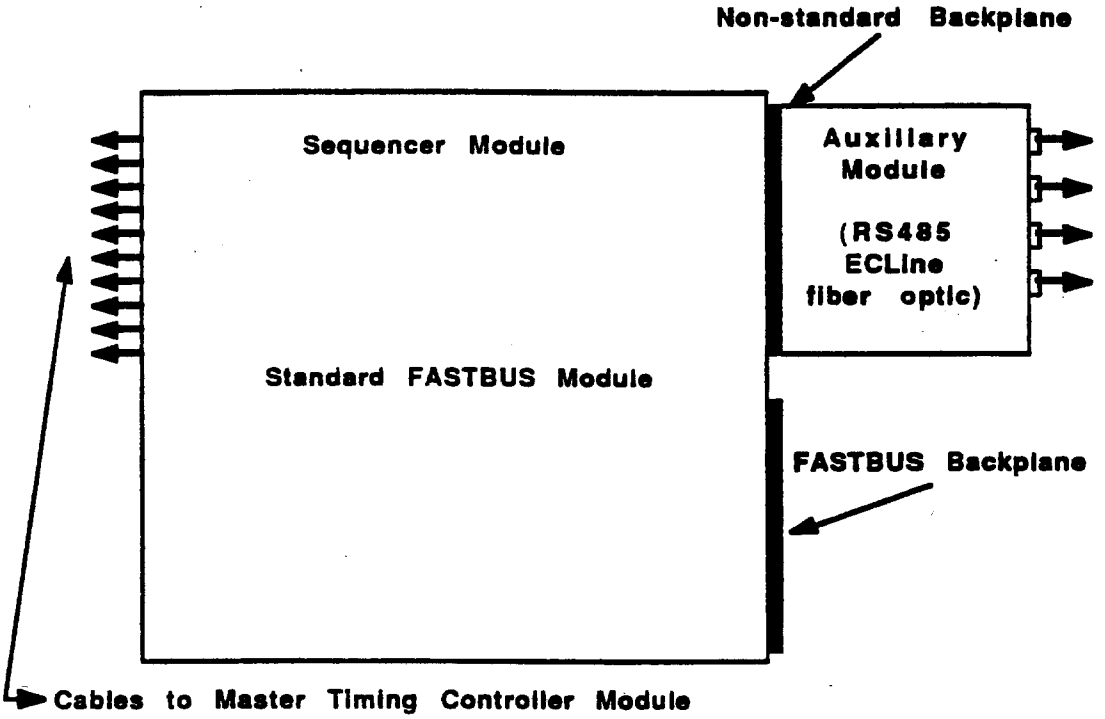


1.1.1. Standard Bus System Used

The sequencer is a single width FASTBUS module and includes FASTBUS slave capability. The FASTBUS slave is used for the initialization of some programmable parameters on the module and may be used to read out the encoded silicon strip detector data.

In addition to the standard FASTBUS interface, the sequencer uses a non-standard backplane on the FASTBUS auxiliary connector position to communicate with the encoder and postamp/discriminator modules. This backplane is optimized for high speed parallel transfers.

Some of these FASTBUS auxiliary connector pins are not used by the special backplane to communicate with the other modules. These pins are used instead by the sequencer to communicate with an I/O link driver such as RS-485, ECLine or fiber-optics. A special auxiliary interface module plugs into the back of the FASTBUS crate for this function. The figure describes this pictorially.

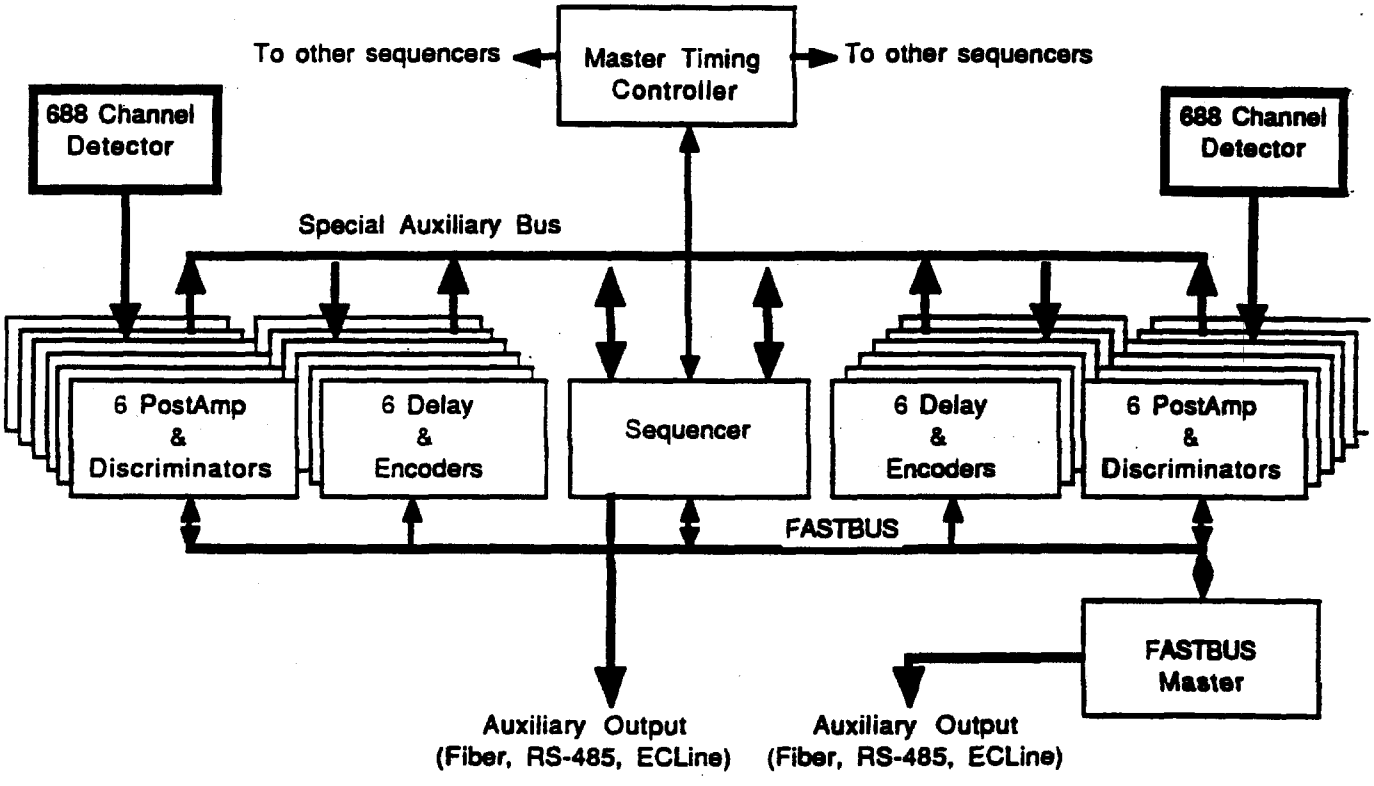


1.1.2. Number of Channels

Data comes into the sequencer from 12 encoder modules in the FASTBUS crate. The data is concatenated and may leave the sequencer via FASTBUS or auxiliary link (RS-485, ECLine or fiber optics).

1.2. Application

The sequencer module sits in the middle of the FASTBUS crate and communicates with the FASTBUS encoder modules via the special auxiliary backplane. Front panel connectors on the module connect via cables to the master controller to provide system clock, trigger commands, system RESET, errors and etc. The sequencer sends its data out via a rear mounted auxiliary card. Data may also be read via FASTBUS transfers. The sequencer may be used at Fermilab in experiment E771 and may be adaptable to other applications. In the E771 application, 24 planes of detector will be used requireing 12 crates of readout electronics. Each crate of electronics will be configured as pictured here:



1.3. Packaging

1.3.1. Module

The sequencer is implemented on a standard single width FASTBUS module which is 15.878" X 14.437" plus front panel. The module plugs into a FASTBUS crate which is then installed in an equipment rack along with the FASTBUS power and cooling system.

If the auxiliary port is used to output data, then a FASTBUS standard size auxiliary module is also used which contains logic and interface hardware to provide a high speed RS-485, ECLine or fiber optic data link. This auxiliary module plugs into the rear of the backplane at the sequencer module position in the slot provided for it by the FASTBUS crate.

1.3.2. Front and Rear Controls Connectors and Displays

1.3.2.1. CLK2 Delay Switch

The 53 Mhz clock received via a front panel connector is delayed by a programmable delay line and sent to the post amp/discriminator modules as a signal named CLK1. CLK1 is delayed by another programmable delay line and sent to the encoder modules as a signal named CLK2. The total amount of delay of CLK1 and CLK2 is also introduced on a signal called SYNC. The programmable delay line for CLK1 is programmed by FASTBUS, but the delay line for CLK2 is set by a 6-bit DIP switch. This switch provides a binary progressive delay from 0 to 31.5 nsec in .5 nsec increments.

1.3.2.2. Front Panel Displays

The sequencer module displays the following indicators of module operation and status on its front panel.

- FASTBUS SLAVE ACCESS; this LED indicator lights for a minimum of 100 msec each time a FASTBUS access to the sequencer occurs.

- SYNC ERROR, This LED indicator lights and remains lighted until RESET by FASTBUS any time an encoder module in the crate signals the sync error condition. Sync error is signalled by an encoder module if its write counter is not at the zero count when the sync signal is received from the Master Timing Controller via the sequencer.

- ENCODER FIFO OVERFLOW, This LED indicator lights and remains lighted until RESET by FASTBUS any time the encoder interface FIFOs are overfilled by data from the encoders. This is a system failure caused by the Master Timing Controller not using the SEQ_READY signal properly. When SEQ_READY is not asserted, the encoder data FIFOs are at least half full and the sequencer should not be given any new trigger addresses. At this time, there is room in the FIFOs to hold the data generated by a single trigger which may have been on its way to the sequencer before the Master Timing Controller received the SEQ_READY being deasserted. Any additional triggers sent by the Master Timing Controller after SEQ_READY is deasserted may generate enough data to overfill the FIFOs and generate the error.

- ENCODERS BUSY, This LED indicator is lighted by the logical 'OR' of the DATA VALID signals from each encoder in the crate. The signal is not 'stretched' or latched and normally occurs too fast to be seen. If this LED remains lighted, an encoder in the crate has failed and has effectively halted the system.

- SEQUENCER BUSY, This LED indicator is lighted by the logical 'OR' of the FIFO half-full signals from each encoder FIFO ion the sequencer module. The signal is not 'stretched' or latched and normally occurs too fast to be seen. If this LED remains lit, the sequencer is not processing the FIFO data out to FASTBUS and/or the auxiliary interface and has effectively halted the system. This may be caused by data not being removed from the FASTBUS event buffer or by the auxiliary link not being ready, or by PERMIT IN not being asserted.

- EVENT IN BUFFER, This LED indicator is lighted by the presence of data in the FASTBUS event FIFO. It will stay lit as long as that FIFO is not empty.

- EVENT SIZE OVERFLOW, this LED indicator is lighted by the occurance of an event which produces more that 255 'hits'. The sequencer does not 'know' this error has occurred until the data has already been received into its encoder FIFOs and is being counted as it is read out to FASTBUS and/or the auxiliary port. The occurrance of this error does not justify having to initialize the system yet a problem exists in how to dispose gracefully of the extra data. In normal operation the control logic will truncate an event at 255 'hits', properly insert the 'LAST WORD' into the data stream, and remove all extra data from the encoder FIFOs without FASTBUS intervention. Optionally, under the control of a bit in CSR0, the control logic will output the additional data while allowing the word counter to wrap around. This will be useful for system tests.

- +5 VOLT STATUS, this LED indicates that the board's +5 volt bus is powered up.

- -5.2 VOLT STATUS, this LED indicates that the board's -5.2 volt bus is powered up.

1.3.2.3. Front Panel Connectors

Signals marked '+' are 'party lined' in a single flat cable. One end connects to the Master Timing Controller (MTC) which terminates the signals. The sequencers attach at various places along the cable. The last sequencer module on the cable provides signal terminations.

- 53 Mhz CLOCK input, terminated NIM, from MTC.

+ WRITE ENABLE input, differential ECL, asynchronous to clock, from MTC. This signal is bussed to each encoder module over the auxiliary backplane. Enables encoders to accept hit data.

- SYNC input, terminated NIM, synchronous to clock, from MTC. This signal gets delayed by the same amount as CLK1 plus the same amount as CLK2 before being bussed to the encoder modules over the auxiliary backplane to test synchronization at each zero count.

+ RESET input, differential ECL, asynchronous to clock, resets control logic and fifos, bussed over the auxiliary backplane to the encoders where it resets write counters.

+ EVENT ADDRESS input, differential ECL, 8-bits, asynchronous to clock, bussed to encoder modules over the auxiliary backplane to designate a stored event. Note that this is actually the address of the event previous to the one requested by the trigger system. This is because the encoders are required to evaluate the previous RF bucket for 'hits' when encoding an event.

+ ADDRESS VALID input, differential ECL, asynchronous to clock, bussed to encoder modules over the auxiliary backplane to initiate encoding of a stored event.

+ SEQ_READY output, wire OR'd ECL, active low, signals that the sequencer has room in its fifos for events and thus can accept "read addresses".

+ ENC_READY output, wire OR'd ECL, active low, this is the sum of all encoder data valid signals. It signals to the master controller that the encoders are ready to accept another trigger.

+ ERROR output, wire OR'd ECL, active high, signals that the sequencer fifos have been overfilled by the encoders or that an encoder has lost synchronization. A fatal error requiring system RESET.

- PERMIT IN - Single ended TTL. A signal from the upstream module indicating that the sequencer may transmit. It's used to allow multiple sequencers to feed a party line.

- PERMIT OUT - Single ended TTL. A signal to a downstream module used to allow the next sequencer in a daisy chain to use the auxiliary party line if applicable.

1.3.2.4. Special Auxiliary Backplane Signals

- CLK1, delayed from the front panel 53 Mhz clock. Individually routed to each post amp/discrimminator module. The delay between the 53 Mhz clock received via the front panel, and this signal, is programmable via FASTBUS from 0 to 32 nsec in .5 nsec increments.

- CLK2, delayed from CLK1, individually routed to each encoder module. The delay between CLK1 and this signal is programmable via a 6-bit DIP switch from 0 to 32 nsec in .5 nsec increments.

- CLK3, 26.5 Mhz, CLK2 divided by two, bussed to all encoder modules for use as a data clock.

- RESET, Asynchronous, Bussed to all discriminators and encoder modules.

- WRITE ENABLE, Synchronized to 53 Mhz CLK2, bussed to all encoder modules, active low.

- EVENT ADDRESS, Asynchronous , 8-bits. Used to designate which memory location in the encoders to read out and encode. Note that this is actually the address of the event previous to the one requested by the trigger system. This is because the encoders are required to evaluate the previous RF bucket for 'hits' when encoding an event.

- ADDRESS VALID, Asynchronous , derived from the Master Timing controller signal . Used to validate Event Address.

- EVENT DATA, Synchronized to 26.5 Mhz CLK3, validated by data valid, 8-bits from each encoder module. Not bussed.

- DATA VALID, Synchronized to 26.5 Mhz CLK3, validates the event data, 1-bit from each encoder module. Not bussed.

- SYNC, This signal is received at the front panel, then delayed by the total of CLK1 delay plus CLK2 delay, then sent to the encoder modules. It is used to verify that all encoder module write counters are at the same count (zero) at sync time. Active low.

- SYNC ERROR, Asynchronous bussed signal from the encoder modules, signifies that one or more of the encoders is out of sync with the system.

1.4. Power Requirements

+5 Volts @ ?? Amps

-5.2 Volts @ ?? Amps

-2 Volts @ ?? Amps

1.4.1. Control and Monitoring Requirements

No requirement exists for a special control or monitoring of the power supply to the sequencer module. The normal protection provided by the FASTBUS crate enviremental system is sufficient.

1.5. Cooling Requirements

The sequencer represents a heat load of ?? watts which must be absorbed and carried away by the FASTBUS cooling system.

2. THEORY OF OPERATION AND OPERATING MODES

The sequencer module sits in the middle of the FASTBUS crate and communicates with the FASTBUS encoder modules via a special auxiliary backplane. Front panel connectors on the module connect via cables to the master controller to provide system clock, trigger commands, system RESET, errors and etc. The sequencer sends its data out via a rear mounted auxiliary card. Data may also be read via FASTBUS transfers.

2.1. Basic Operation

At power up and system reset time (i.e., the reset generated by the Master Timing Controller) the Sequencer Module will go into the initialization mode. In this mode CLK1 is not being driven onto the Auxiliary Backplane. The control logic waits for the Master Timing Controller to assert Write_Enable and then looks for the Master Timing Controller to assert Sync. When Sync is received the Sequencer will start driving CLK1 onto the Auxiliary Backplane. At this point the module is in normal running mode.

On receipt of an event address from the Master Timing Controller, the sequencer generates the event address-1 and broadcasts it to the encoder modules via the special auxiliary backplane. Each encoder module extracts from memory, the 128 channels of hit information from that event and encodes it into an ordered list of 8-bit binary numbers. Each number in the hit list is comprised of a 7-bit binary number identifying the strip which had been driven above threshold by an ionizing particle and 1-bit to indicate if that microstrip had also registered a hit during the previous bucket (thus indicating that the current hit might actually be a residual from that bucket). All encoders do their encoding concurrently and send the data over individual data paths on the auxiliary backplane to FIFOs in the sequencer module.

As the sequencer begins receiving data from the encoder modules, it commences delivering it in order to its I/O ports. Beginning with the lowest numbered encoder FIFO, the data is transferred into the auxiliary card for transmission and into the FASTBUS buffer FIFO for readout. All the data from the 12 encoders are concatenated into a single ordered list. The ordering of the list is set by the encoder modules as lowest hit address to highest.

As each encoder FIFO produces an 'end of event token', the control logic moves on to the next. When the last FIFO has been emptied, the control logic add the contents of the BLOCK COUNT/WORD COUNT and an IGNORE WORD if necessary to fill out the last 32-bit word in the data stream.

Referring to the block diagram, the sequencer consists of seven functional blocks.

2.1.1. Master Timing Controller Interface

- Receive 53 Mhz Clock. A function of the sequencer is to distribute clocks to the discriminator and encoder modules. A front panel connector on the sequencer receives a 53 Mhz clock from the master control module. This clock is exactly the same phase at all the sequencers. This clock has a fixed phase relationship to the signals arriving from the silicon microstrip detector. The sequencer derives two 53 Mhz clocks from this signal, CLK1 and CLK2. CLK1 is delayed from the input clock by a FASTBUS programmable amount and is driven to the discriminator modules via transmission lines on the auxilary backplane. CLK2 is delayed from CLK1 by a switch selectable amount and is driven to the encoder modules via transmission lines on the auxilary backplane. A divide-by-two version of CLK2 called CLK3 is bussed by the sequencer to the encoder modules to be used as a data readout clock.

- Receives system WRITE ENABLE signal from the Master Timing Controller.

- Receives SYNC signal from the Master Timing Controller.

- Receives RESET signal from the Master Timing Controller.

- Receives EVENT ADDRESSES and ADDRESS VALID from the Master Timing Controller.

- Sends 'sequencer ready' signal to the Master Timing Controller (SEQ_READY).

- Sends 'encoders ready' signal to the Master Timing Controller (EN_READY).

- Sends ERROR signal to the Master Timing Controller. This signal alerts to an encoder FIFO overflow, encoder out of sync and others to be added. FASTBUS can interrogate the Error Status Register to evaluate the error. Event size greater than 255 is not an error condition.

2.1.2. Encoder Interface

- Contains the FIFOs which receive data from each of the 12 encoder modules.
- Deposits 'end of event' token into each FIFO after all data has been received from the encoder.
- Sends RESET signal to the Encoder modules on command of the front panel signal.
- Sends WRITE ENABLE signal to the Encoder modules.
- Sends EVENT ADDRESSES and ADDRESS VALID to the Encoder modules.
- Sends SYNC to the Encoder modules delayed by CLK1 plus CLK2 delay.
- Receives SYNC ERROR from the Encoder modules.
- Receives EVENT DATA and DATA VALID from each Encoder module.

2.1.3. FASTBUS Interface

- Allows a FASTBUS master to read/write the control and status register.
- Allows a FASTBUS master to read/write the PLANE/ENCODER RAM.
- Allows a FASTBUS master to read/write the BLOCK COUNT/WORD COUNT.
- Allows a FASTBUS master to read/write the CLK1 delay value and read the CLK2 delay value.
- Allows a FASTBUS master to read the EVENT BUFFER FIFO.

2.1.4. Auxiliary Interface

The auxiliary interface is a high speed outlet for event data. Unless disabled by a CSR0 bit, the sequencer will drive data out this port directly, bypassing the need for software intervention to read out data over FASTBUS. This interface is designed so that it can drive an RS-485, ECLine or fiber optic auxiliary module. The auxiliary module provides the special interface requirements of the link. The signals provided by the sequencer to the fiber optic interface are listed below. The other types of interface modules ie. RS-485 and ECLine will have to be designed to utilize these same signals.

The auxiliary port interface includes the following signals. See the appendixes for the actual pin numbers.

- 40 data and control lines to the auxiliary module.
- Control/data strobe to the auxiliary module.
- Acknowledge from the auxiliary module.
- 6 status lines from the auxiliary module.
- Status Strobe from the auxiliary module.
- Link Error signal from the auxiliary module

The sequencer will be capable of initializing the data link by outputting a simple stream of control nibbles in response to a FASTBUS INITIALIZE command. The control/data lines from the auxiliary module will be used to indicate link errors. They will also be used to set/reset a latch indicating that the destination of the sequencer data has enough buffer space available to accept a maximum sized event. The sequencer will check this latch before sending an event out the auxiliary port.

2.1.5. Block/Word Counters

The BLOCK COUNT/WORD COUNT is a 16-bit entity which is normally used as the last valid word of a concatenated event. It consists of a 4-bit type code, a 4-bit counter for use as a BLOCK COUNT and an 8-bit counter for use as a WORD COUNT. The 4-bit type code is a binary '1000' placed in bit position 15-12 to identify the data as being the BLOCK COUNT/WORD COUNT of an event as opposed to data. The 4-bit BLOCK COUNT is intended to be used as a processed event identifier to provide parallel synchronization among the 12 FASTBUS crates. The 8-bit WORD COUNT is intended to be used to count the number of 'hits' in each event. The counters are cleared and incremented individually by the control logic, but their combined contents are always output as a 16-bit word. The counters may be written and read via FASTBUS for diagnostic purposes.

2.1.6. FASTBUS Event FIFO

FASTBUS output will be via a single event FIFO (32 X 512). Unless disabled by a CSR0 bit, a full event including LAST WORD and IGNORE WORD if necessary will be loaded into this FIFO as it is being sent out the auxiliary port. Output to FASTBUS will be in 32 bit words. A 32 bit word count will be generated and inserted at the end of the event record exactly as is done for the auxiliary port. If the FASTBUS port is being used for readout, the sequencer will not start to read a second event out through the auxiliary port until the event in the FASTBUS FIFO has been completely read out. A CSR0 bit will disable the FASTBUS event FIFO to allow data output via the auxiliary port without regard to the status of the event FIFO.

2.1.7. Control Logic

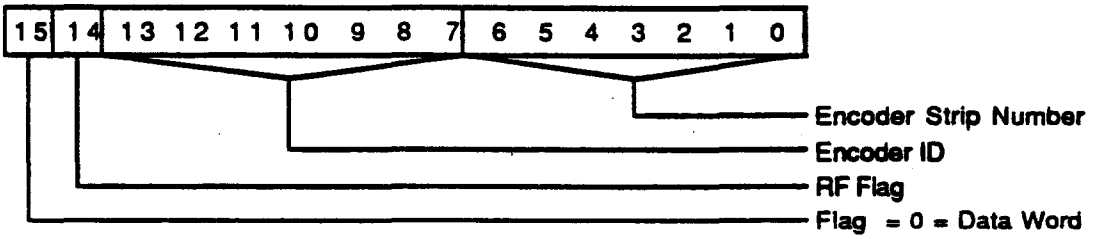
The control logic controls the input fifos, block and word counters, data identifier RAM, auxilliary port interface FIFO and FASTBUS event FIFO. This logic is implemented with programmable array logic devices (PALs).

2.2. Addressing Modes

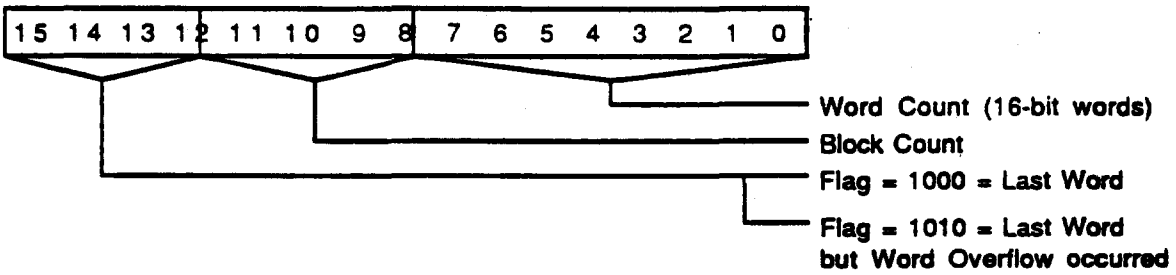
2.2.1. Data Transfer Description and Rates

The sequencer is designed to transfer silicon strip detector data over the auxiliary fiber optic link at peak instantaneous rates approaching 80 nsec per 32-bit word. The sequencer is designed to support block transfer data reads from its FASTBUS event buffer at peak instantanaeous rates approaching 150 nsec per 32-bit word. Whether data is output via the auxiliary port or read via the FASTBUS slave interface, it will always be in 32-bit words and each 32-bit word will always be comprised of two 16-bit words. The two 16-bit words words will each be of one of three types. The three types are illustrated below:

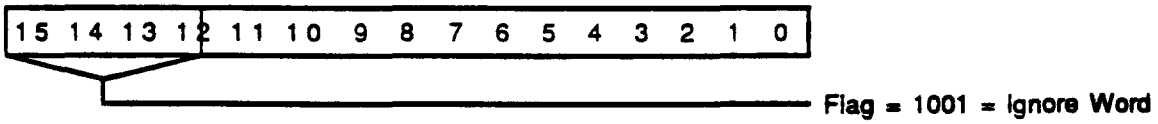
Data Word Format



Last Word Format



Ignore Word Format

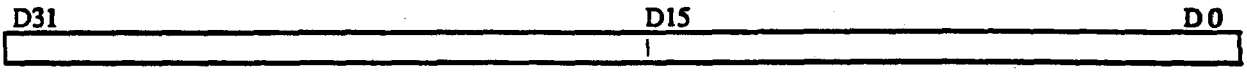


2.2.2. Internal Control, Status Registers, and Bit Descriptions

There are several registers on the Sequencer that are accessible via FASTBUS for initialization, performance monitoring, or diagnostics. These registers are described in the following subsections.

2.2.2.1. FASTBUS mandatory CSR0

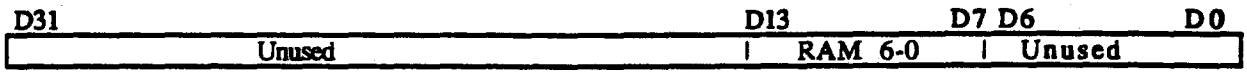
CSR Space Hex Address		Description
0000 0000	CSR 0	Bit descriptions:
	Read , Write	
	Bit <6>,<22>	Disable/Enable sequencer loading of FASTBUS event FIFO.
	Bit <7>,<23>	Disable/Enable sequencer loading of auxiliary interface FIFO.
	Bit <8>,<24>	Disable/Enable overflow truncation.
	Bit ,<30>	Reset sequencer.
	Bits<31:16>,	Manufacturer's ID and device type.



2.2.2.2. Read/Write The PLANE/ENCODER RAM

CSR Space Hex Address	Description
C000 0000 - C000 000B	There are 16 RAM locations accessible by FASTBUS of which 0 thru 11 are used to hold the 7-bit plane/encoder identifying data. 12 locations are

required, one for each encoder FIFO. Although FASTBUS read/writes utilized 32-bits, only 7-bits are used for this operation and the rest are undefined. This memory may by written and read via FASTBUS not only for setting up the module but for diagnostics as well. This is only a 7-bit RAM. The MSB will always read '0' because this is a tag that differentiates DATA WORDs from LAST WORD or IGNORE WORD in the data stream.

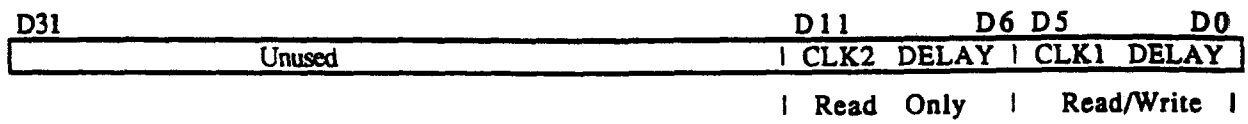


2.2.2.3. Read/Write The BLOCK COUNT/WORD COUNT

CSR Space Hex Address	Description
C000 0010	The BLOCK COUNT/WORD COUNT counters may be written and read via FASTBUS for diagnostic purposes.

2.2.2.4. Read/Write The CLK1 Delay Value, Read The CLK2 Delay Value

CSR Space Hex Address	Description
C000 0011	The CLK1 delay value is FASTBUS read/writeable, 0.5 ns per step, 64 steps. The 53 Mhz clock received via the front panel is delayed by this value to produce CLK1. This value is also added to the switch settable delay for CLK2 to produce the total delay value for the SYNC signal. The CLK2 delay value is set by hardware DIP switches on the card itself. The setting of those switches are read back in bits D11 through D6.



2.2.2.5. Read-Only Error Status Register

CSR Space Hex Address — Description

C000 0012 — FASTBUS readable, bit D0 is set if a Sync Error has been detected by a Delay/Encoder. Bits D1 through D12 correspond to encoder FIFO channels 1 through 12. If one of these bits is set then its corresponding FIFO has overflowed.

D31	D12	D1	D0
Unused	Encoder Fifo Overflow Flags		Sync Err Flag

2.2.2.6. Read The EVENT BUFFER

In order to minimize overhead, the FASTBUS master may attempt to read from the event buffer even before it recieves a trigger and will be forced to WAIT only until data begins streaming into the FIFO. The master must not allow itself to time out while waiting for data. At the end of event processing, the sequencer will load the LAST WORD which contains the BLOCK COUNT/WORD COUNT; and an IGNORE WORD if necessary into the FIFO. After this word is read by the FASTBUS master, SS=2 would terminate the transfer.

The EVENT BUFFER is a 32-bit wide X 512 deep FIFO that holds a single event. The 32-bit word is made by packing two 'hits' which are normally 16-bits. To speed up FASTBUS transfers, this FIFO will be read out by a FASTBUS master using block transfer reads. There is a CSR0 bit to disable the event buffer if the event buffer is not to be used. This bit prevents the control logic from waiting for the buffer to be ready or strobing data into the buffer. This feature allows a FASTBUS master to sample events for histogramming or other reasons.

Data Space Hex Address — Description

0000 0000 — Block transfer read of the event FIFO. Generate a primary address cycle to DATA space followed by block transfer reads. There is no secondary address cycle. The data transfers will be stalled by WAIT during the data cycle until event data begins streaming into the FIFO. The overhead involved with a 68020 powered FASTBUS Smart Crate Controller is as follows: 1) 300 nsec for the move instruction that executes a FASTBUS primary address cycle. 2) 300 nsec for the move instruction that starts the execution of a block transfer. 3) About 150 nsec per 32-bit word of data during the block transfer. The overhead from 1 and 2 is not incurred if the master executes the primary address cycle before a trigger is even received. The last word in the FIFO will have the 'LAST WORD' flag, the BLOCK COUNT and the WORD COUNT. If the number of words is odd, the last word will end up in the LS byte position of the 32-bit longword and an IGNORE WORD with a binary flag of '1100' will end up in the MS word. If the number of 32-bit words in the FIFO is even, the LAST WORD will end up in the MS word of the last 32-bit longword.

Data words:

D31	D15	D0
n+1 Data Word		n Data Word

Last word (if odd number of words):

D31	D15	D0
Ignore Word		Last Word

Last word (if even number of words):

D31	D15	D0
Last Word		n+x Data Word

2.2.3. Error Responses

The ERROR signal is sent to the Master Timing Controller when a Sync Error is detected of when and encoder FIFO overflows. The system software can then read the Error Status Register to determine specifically which of the encoder FIFOs overflowed or if a Sync Error had been detected by this sequencer.

2.2.4. Diagnostic Software

To be added

3. INPUT/OUTPUT SPECIFICATIONS

3.1. Communication Interfaces

3.1.1. Fiber Optic Auxiliary Port

The sequencer auxiliary port can be fitted with an auxiliary module which provides a high speed fiber optic link. Details about this auxiliary module is found in another document by the Detector Electronic Systems Group. Data delivered by the sequencer fiber optic port will have the same format as data delivered by the sequencer FASTBUS event buffer interface. However, the fiber optic link protocol requires some additional control operations to support handshaking, error reporting, etc.

There is a CSR0 bit which may be used by a FASTBUS master to disable the auxiliary port if the port is not to be used. This bit prevents the control logic from waiting on a port ready signal and from strobing data into the auxiliary port.

3.1.1.1. Communication Protocol

The sequencer communicates with the auxiliary card in the following fashion:

The sequencer will send an INITIALIZE command and then wait for the auxiliary module to return RECEIVED INITIALIZE status.

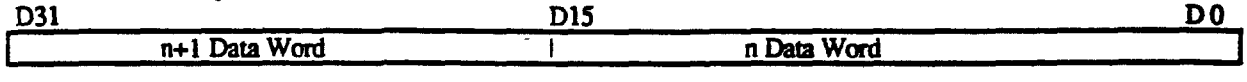
When the sequencer has data to transmit it issues INQUIRE commands until the auxiliary module returns READY status indicating that it can take the data.

The sequencer then transmits its data synchronized to the clock driven by the auxiliary card. as long as the auxiliary card is sending READY status.

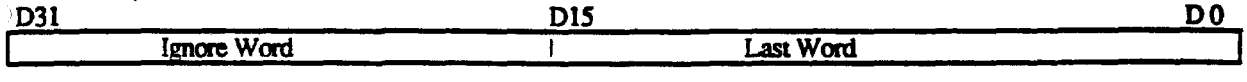
If the auxiliary card sends NOT READY status while the sequencer is transmitting data the sequencer will revert to the state where it transmits INQUIRE commands until the auxiliary card again issues READY status and data transmission will pick up where it left off.

The sequencer will return to the INITIALIZE state when it runs out of data to transmit, when it times out waiting for status after issueing an INQUIRE command, or when it receives a LINK ERROR signal from the auxiliary card.

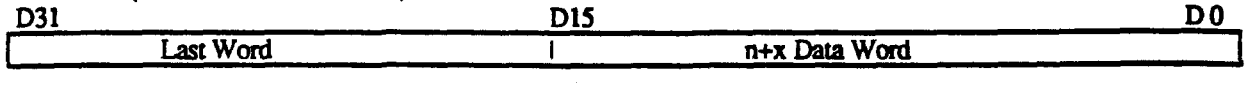
First and all subsequent normal data words:



Last word (if odd number of words):



Last word (if even number of words):



4. SYSTEM SOFTWARE DESCRIPTION

4.1. Initialization Description Including Documented Code

Upon initialization, FASTBUS must be used to load the 'encoder identification RAMs', and to set the CLK1 delay. FASTBUS may also be used to write and read all internal registers for diagnostic purposes.

4.2. System Software

5. SYSTEM, MODULE, CIRCUIT, OR CHIP DIAGNOSTICS

5.1. Hardware

5.1.1. Special Test Modules and/or Test Setup Descriptions

5.1.2. Operating Instructions

5.2. Software

5.2.1. Diagnostic Test Description

Software to test the funtionality to the sequencer will include the following tests:

Write and read the CLK1 delay value via FASTBUS.

Read the CLK2 delay value via FASTBUS.

Write and read the twelve encoder identification RAM locations from FASTBUS.

Write and read the block counter from FASTBUS.

Write and read the word counter from FASTBUS.

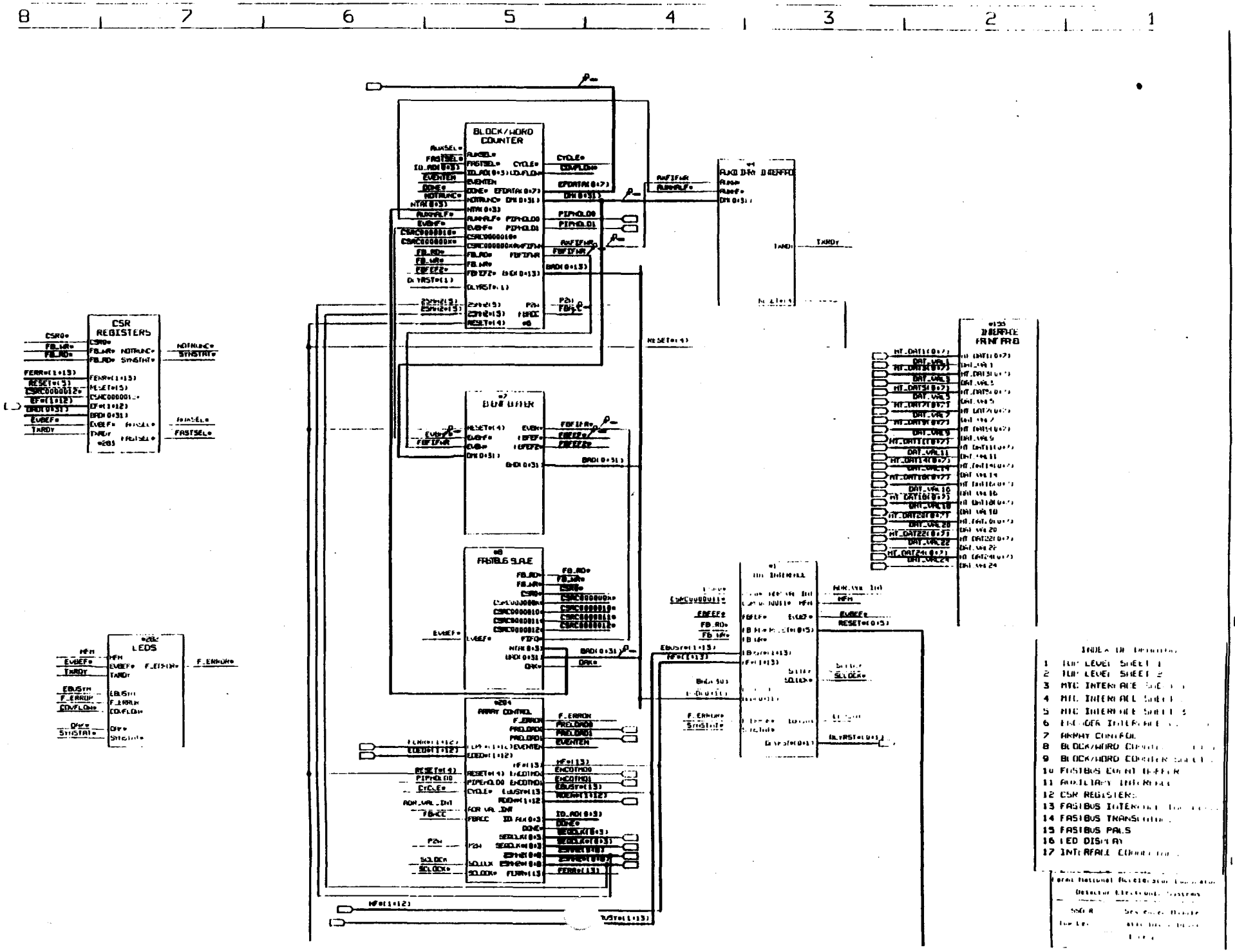
Write and read CSR 0 from FASTBUS.

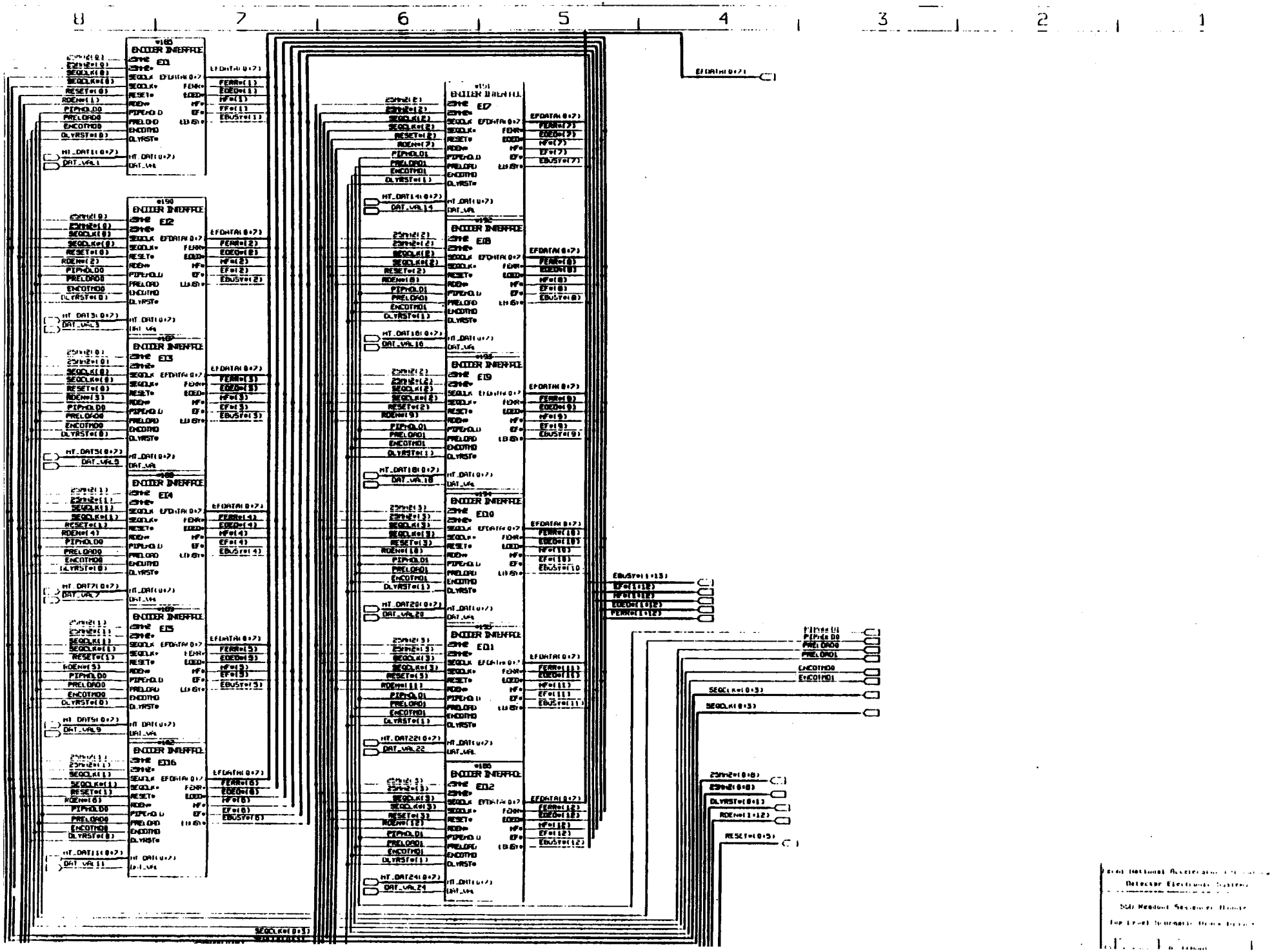
Read the Error Status Register from FASTBUS.

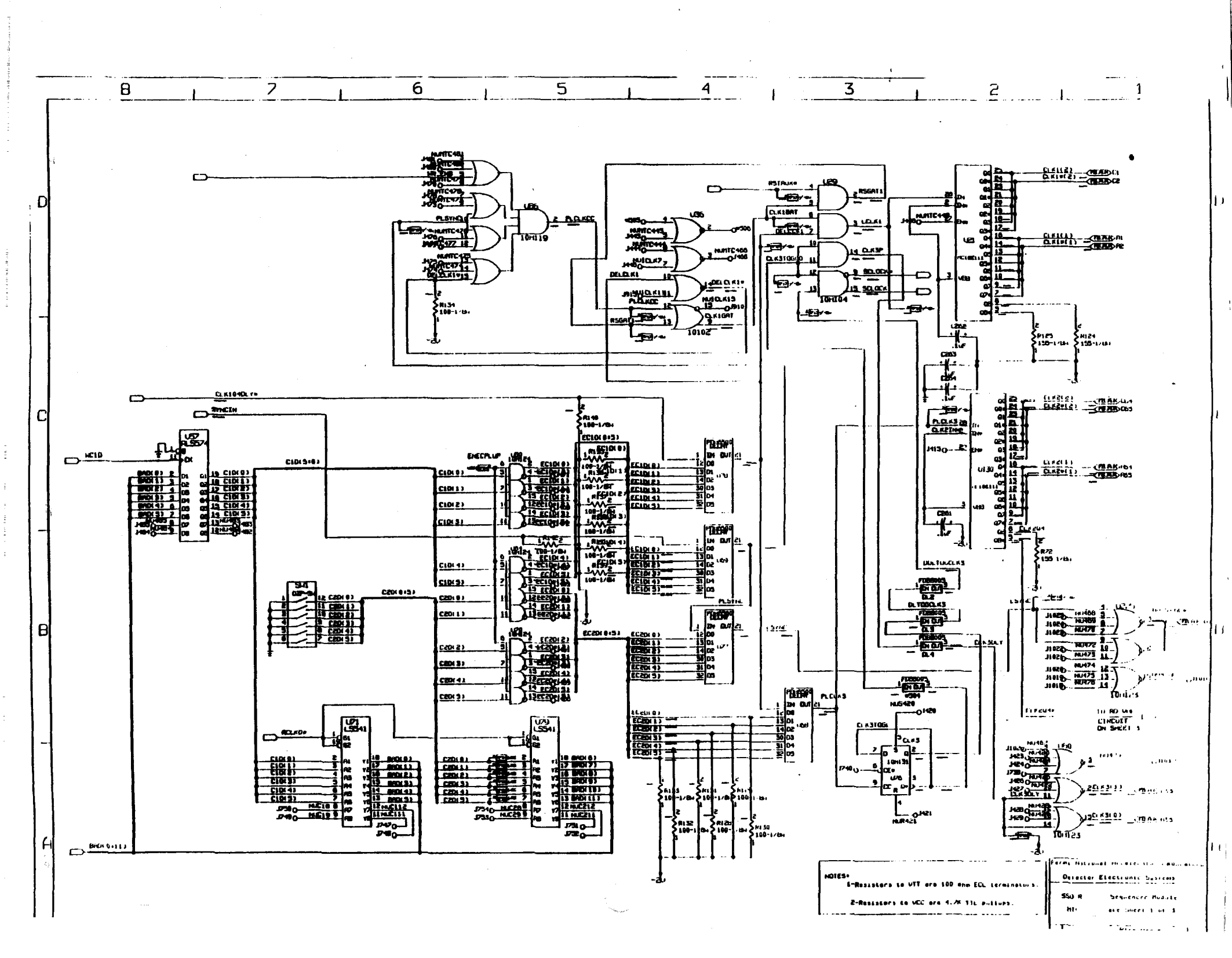
5.2.2. Documented Listing of Each Test Software Module

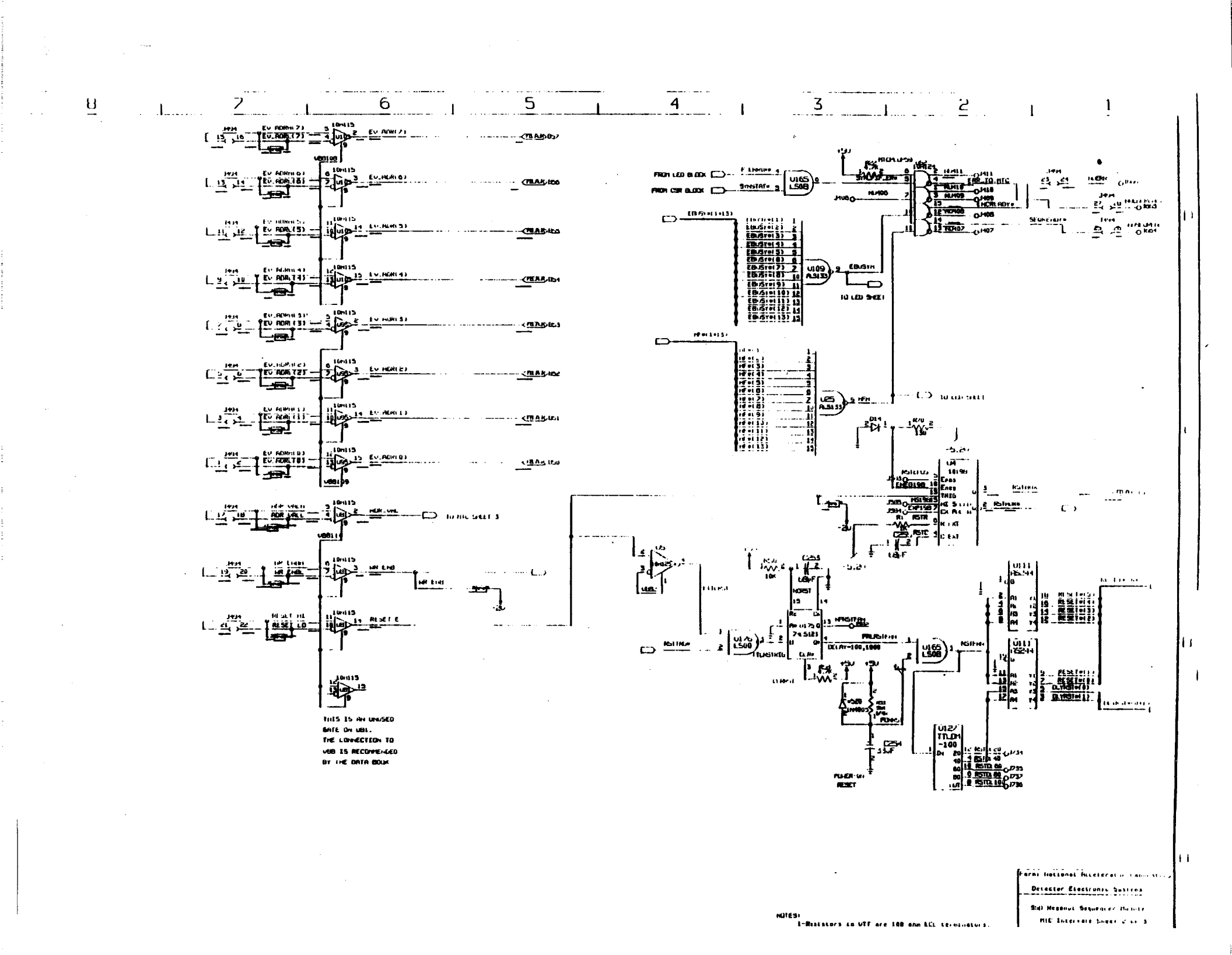
APPENDIX A

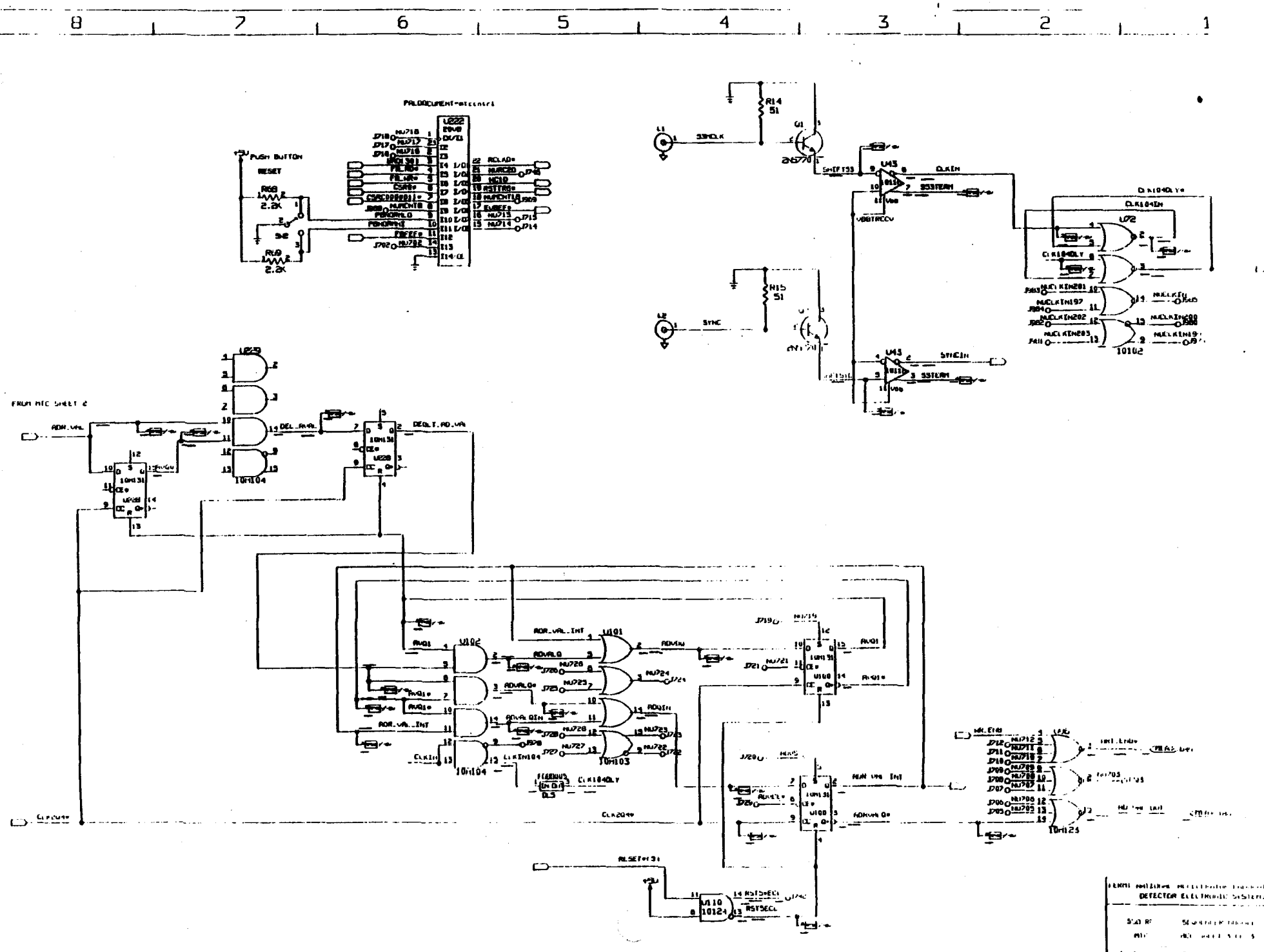
CIRCUIT DIAGRAMS

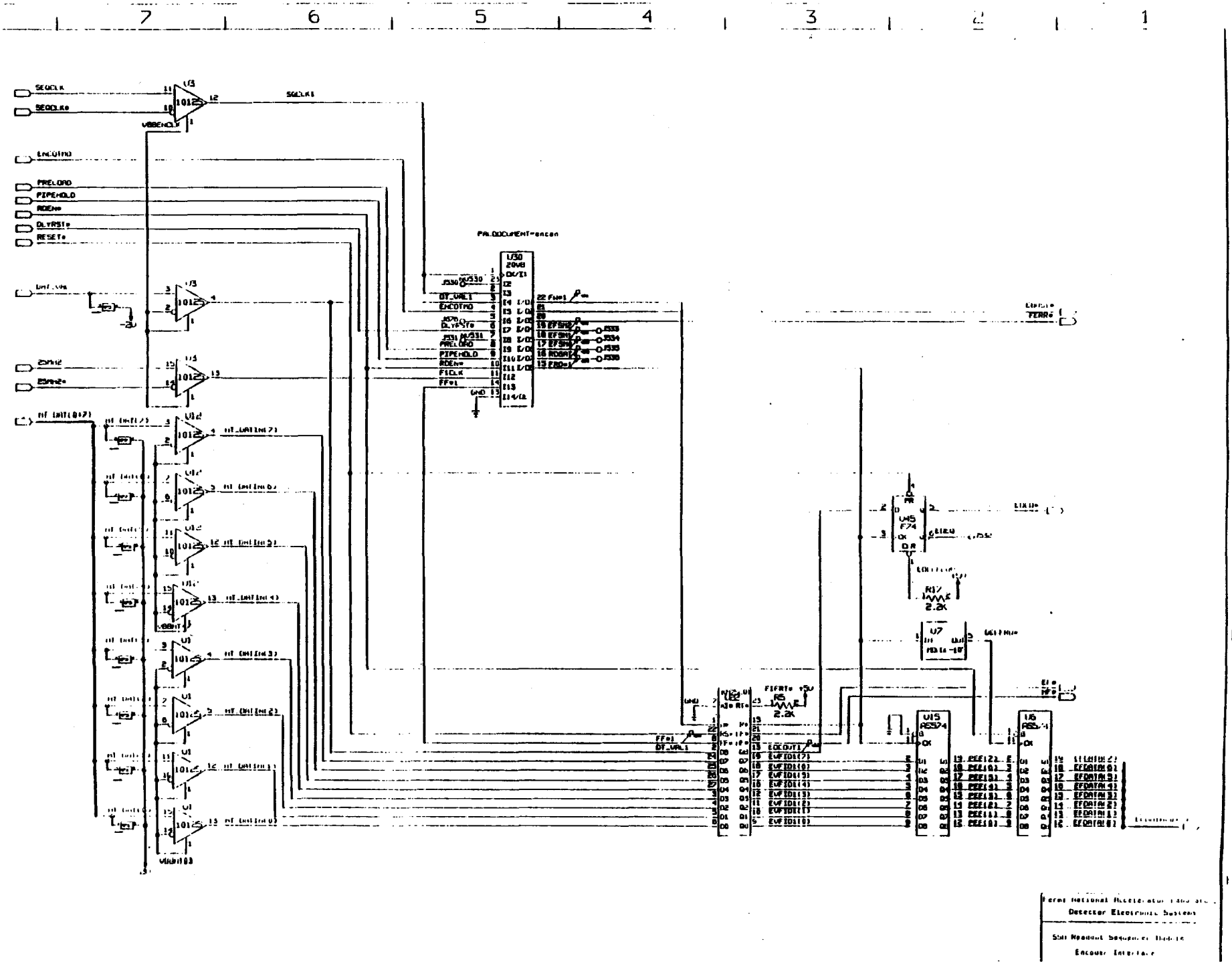

BLOCK/WORD COUNTER
CSR REGISTERS
FASTBUS SLAVE
ARRAY CONTROL
LEDS
INDEX OF DRAWINGS
1 TOP LEVEL SHEET 1
2 TOP LEVEL SHEET 2
3 MTC INTERFACE SHEET 3
4 MTC INTERFACE SHEET 4
5 MTC INTERFACE SHEET 5
6 ENCODER INTERFACE
7 ARRAY CONTROL
8 BLOCK/WORD COUNTER
9 BLOCK/WORD COUNTER SHEET 2
10 FASTBUS EVENT BUFFER
11 AUXILIARY INTERFACE
12 CSR REGISTERS
13 FASTBUS INTERFACE
14 FASTBUS TRANSCEIVER
15 FASTBUS PALS
16 LED DISPLAY
17 INTERFACE CONNECTOR

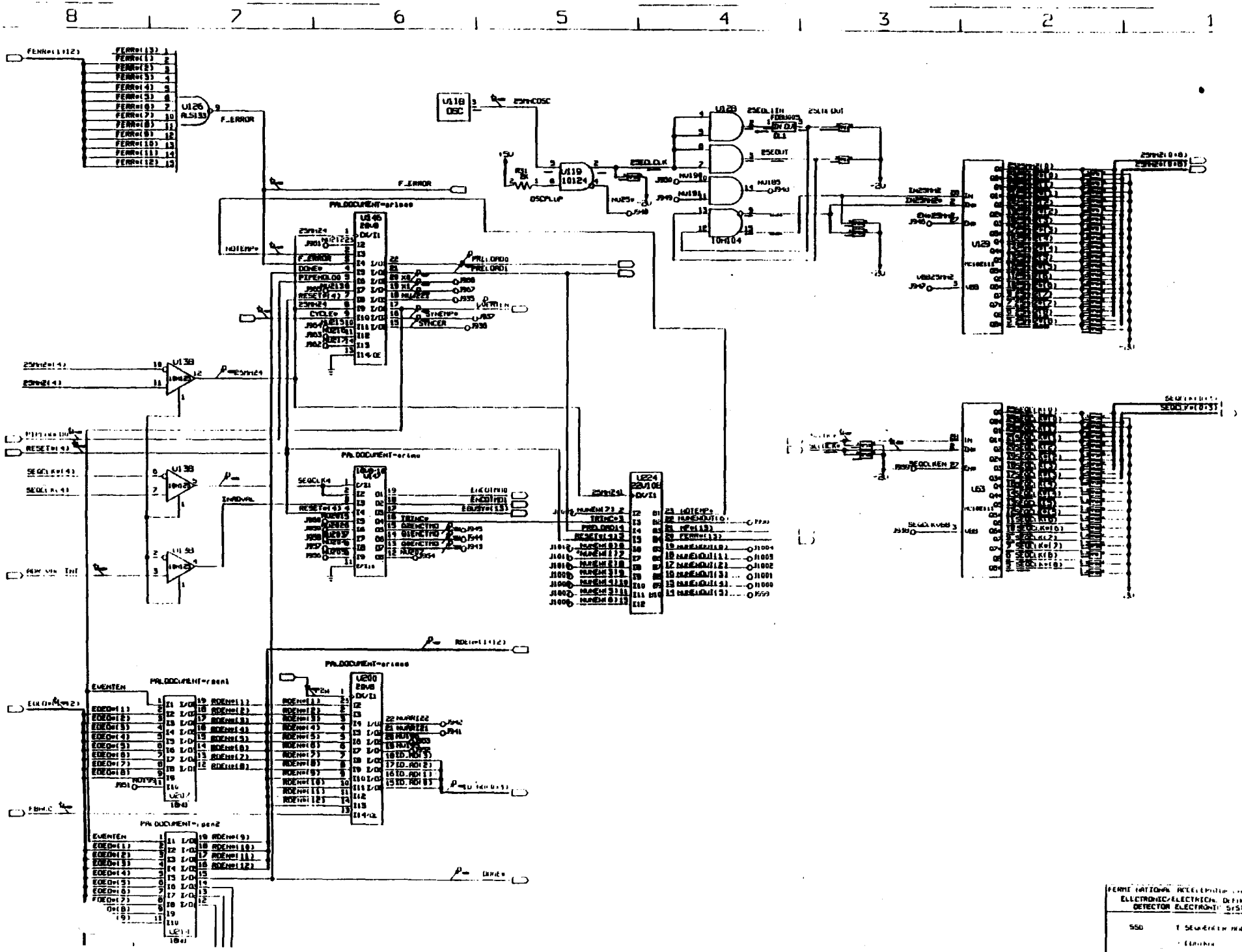

8
7
6
5
4
3
2
1
ENCODER INTERFACE
EI1
EI2
EI3
EI4
EI5
EI6
EI7
EI8
EI9
EI0
EI1
EI2
EFDATA[0:7]
FERR
EOED
HF
EF
EBUSY
SEQCLK[0:3]
RESET
RDEN
PIPEHOLD
PRELOAD
ENCOTMD
DLYRST
HT_DAT[0:7]
DAT_VAL

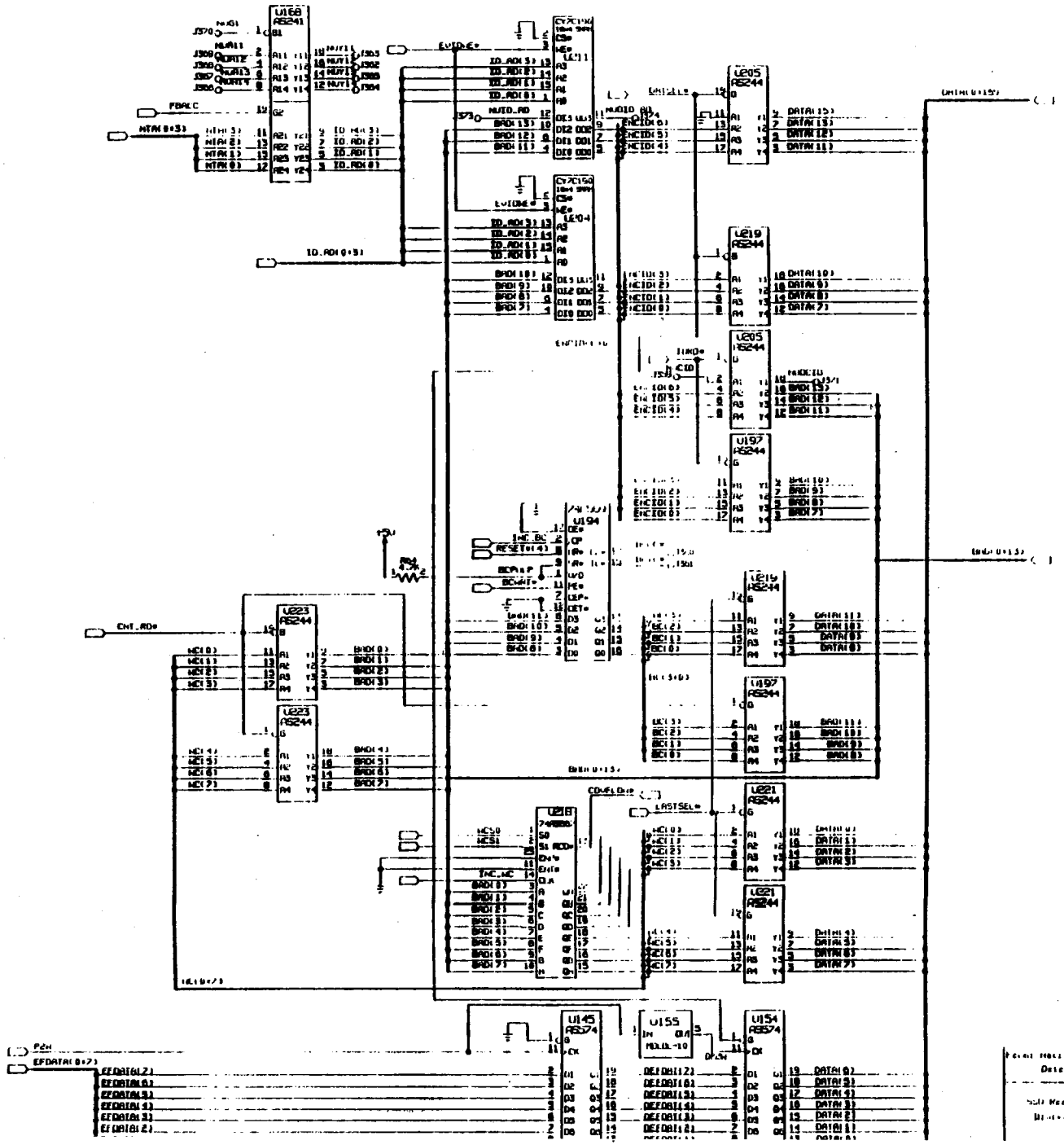

8
7
6
5
4
3
2
1
D
C
B
A
10H119
10H102
10H104
10H125
NOTES:
1-Resistors to VTT are 100 ohm ECL terminators.
2-Resistors to VCC are 1.2K TTL pullups.
Detector Electronic Systems
SSD R
Sequencer Module

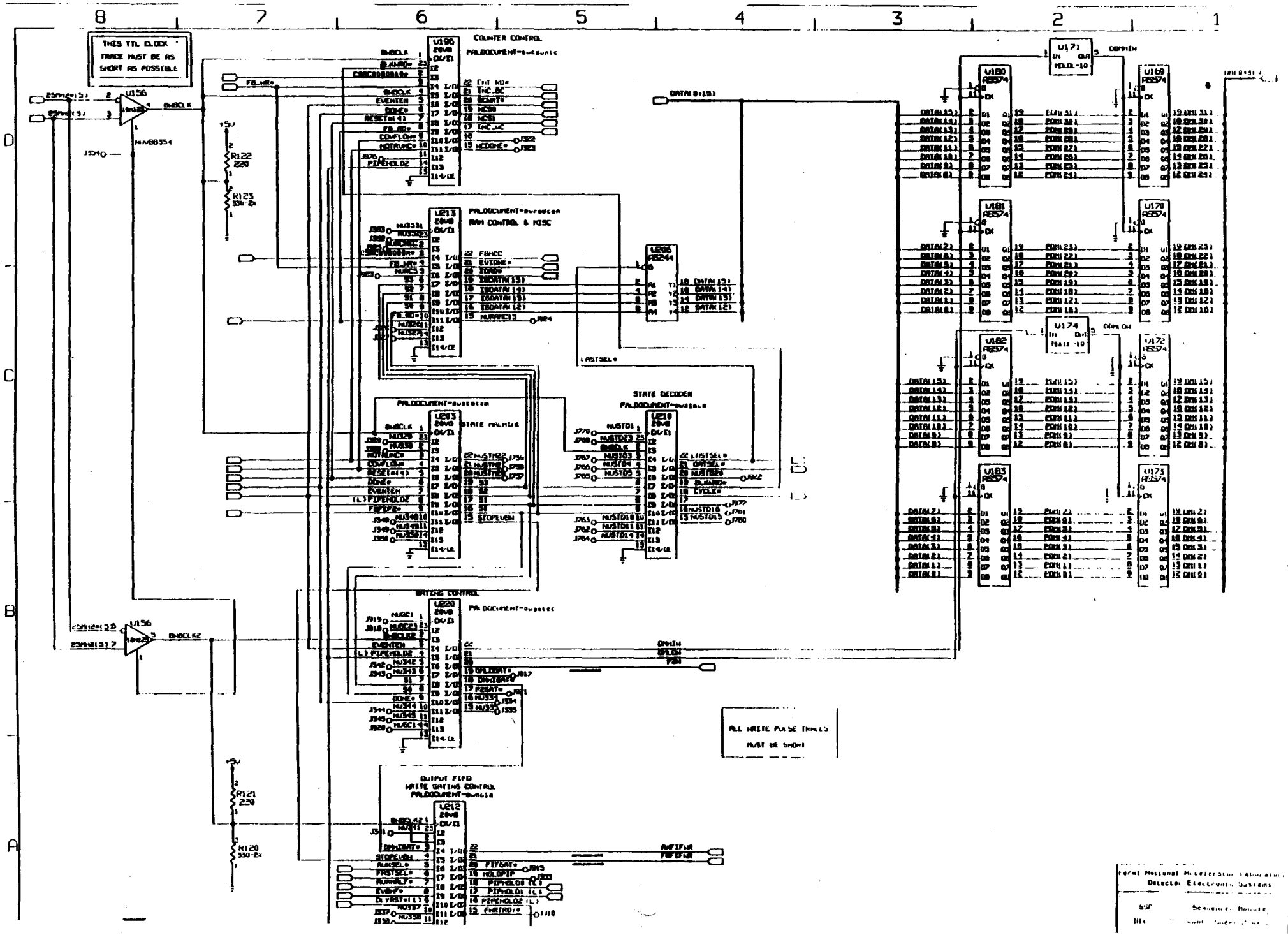

8
7
6
5
4
3
2
1
FROM LED BLOCK
FROM CSR BLOCK
U165 LS08
U109 ALS133
U5 ALS133
TO LED SHEET
THIS IS AN UNUSED GATE ON U81. THE CONNECTION TO VBB IS RECOMMENDED BY THE DATA BOOK
NOTES:
1-Resistors in UTF are 100 ohm ECL terminators.
Detector Electronics Systems

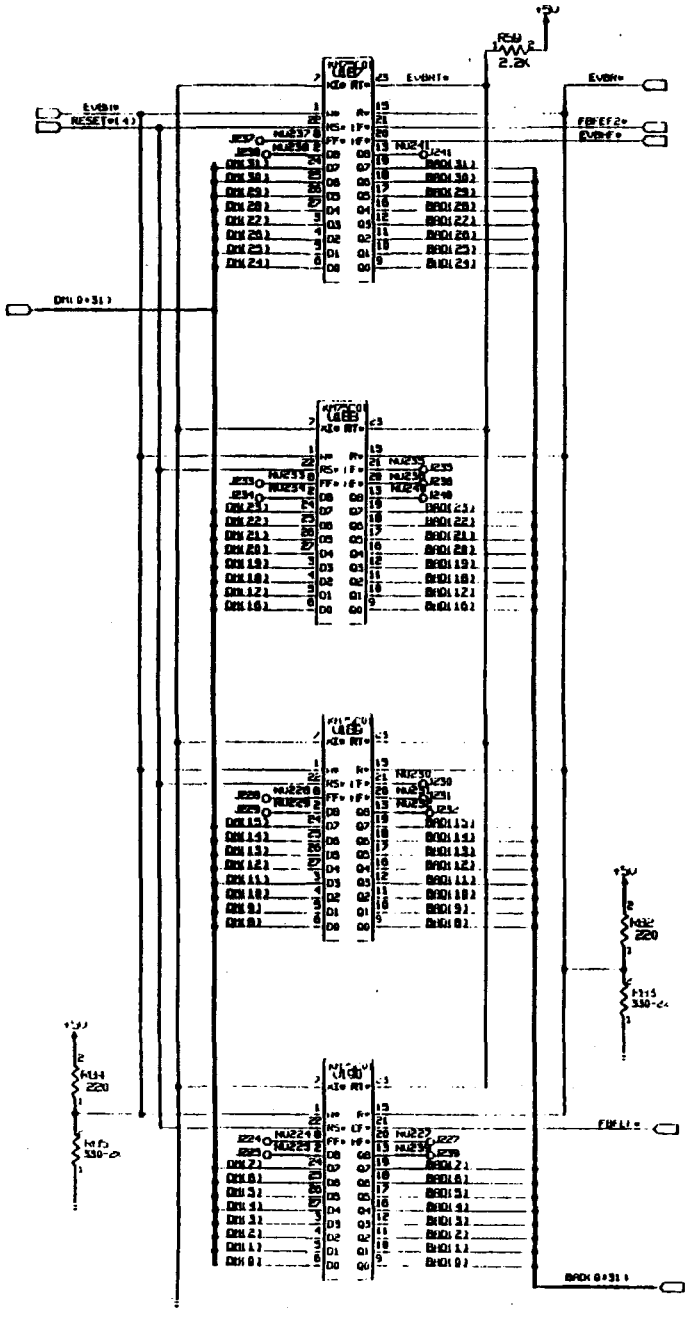

8
7
6
5
4
3
2
1
D
C
B
A
PUSH BUTTON
RESET
R68
2.2K
R69
2.2K
R14
51
R15
51
SYNCLK
SYNC
2N5770
SHIFT33
CLKIN
SYSTEM
VBBTRECV
SYNCIN
CLK104IN
CLK104DLY
NUCLKIN201
NUCLKIN197
NUCLKIN202
NUCLKIN203
10102
FROM HTC SHEET 2
ADR.VAL
DEL.ADR.VAL
DELT.ADR.VAL
10H104
10H134
10H103
10H125
ADR.VAL.INT
ADRIN
ADVALO
ADVALOIN
CLKIN
CLK104IN
CLK204IN
RESET33
RSTSECL
10124

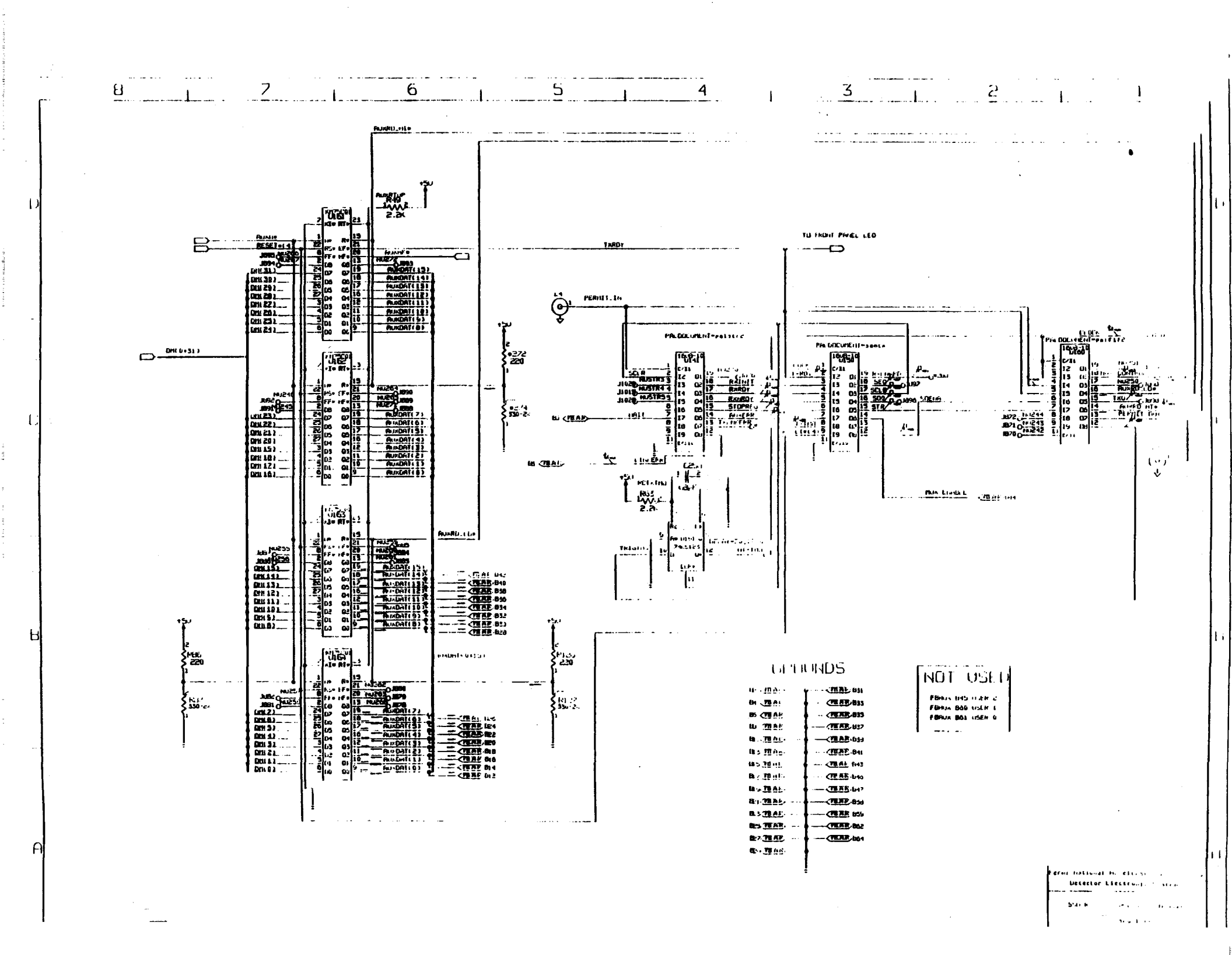

8
7
6
5
4
3
2
1
SEQCLK
SEQCLKo
ENCOTPO
PRELOAD
PIPEHOLD
RDENo
DLYRSTo
RESETo
SEQCLK1
PRELOADTPenable
DT_VAL1
ENCOTPO
DLYRSTo
PRELOAD
PIPEHOLD
RDENo
FCLK
FFo1
GND
FERRo
EFDATA(7)
EFDATA(6)
EFDATA(5)
EFDATA(4)
EFDATA(3)
EFDATA(2)
EFDATA(1)
EFDATA(0)
R17
2.2K
FIFRTo +5V
R5
2.2K
U50
U5
U6
U7
Detector Electronics System
Encoder Interface

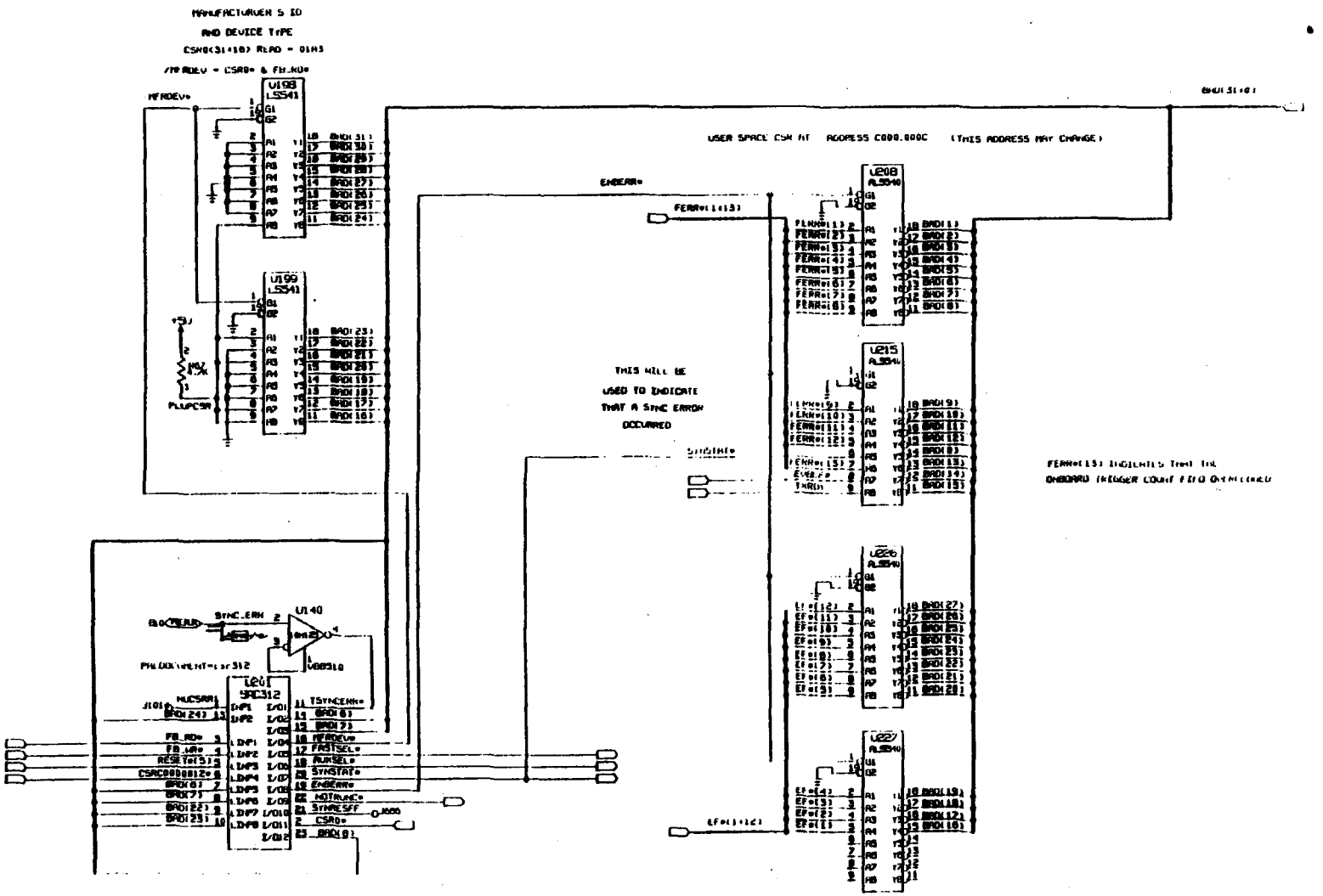

8
7
6
5
4
3
2
1
D
C
B
A
F_ERROR
PRELOPDO
PRELOPDI
RESET
SEQCLK
ENEVAL
EVENTEN
RDEN
DRDEN
PRELOAD
SYNCER
ESCOUT
ESCLK OUT
OSCPLUP
PRLOCUMENT=arlaaa
PRLOCUMENT=arlaaa
PRLOCUMENT=arlaaa
PRLOCUMENT=arlaaa
PRLOCUMENT=arlaaa
FERMI NATIONAL ACCELERATOR LABORATORY
ELECTRONICS/ELECTRICAL DEPARTMENT
DETECTOR ELECTRONIC SYSTEMS



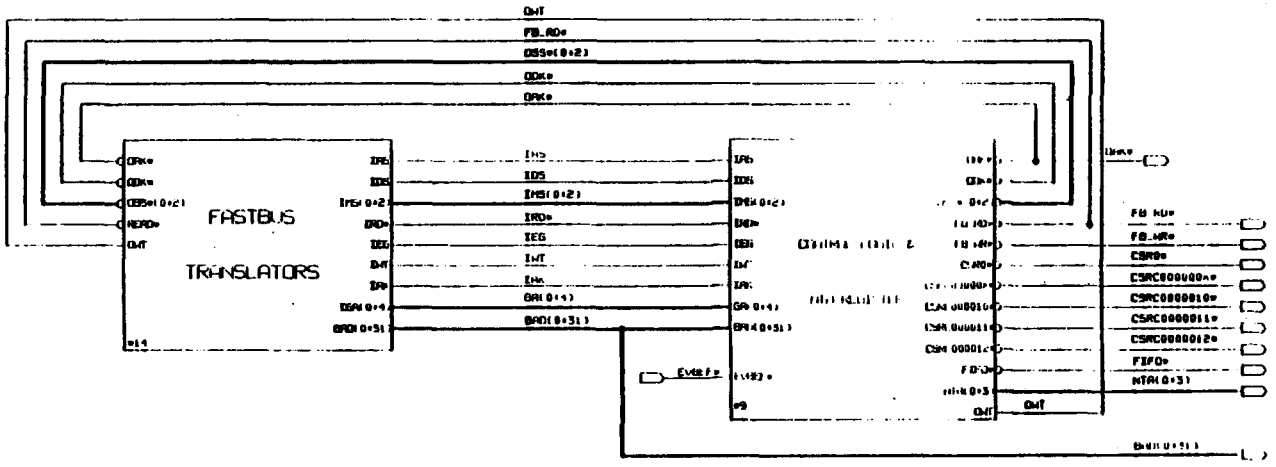

THIS TTL CLOCK TRACE MUST BE AS SHORT AS POSSIBLE
COUNTER CONTROL
STATE DECODER
DATING CONTROL
OUTPUT FIFO WRITE DATING CONTROL
ALL WRITE PULSE TIMES MUST BE SHOWN
8
7
6
5
4
3
2
1
D
C
B
A

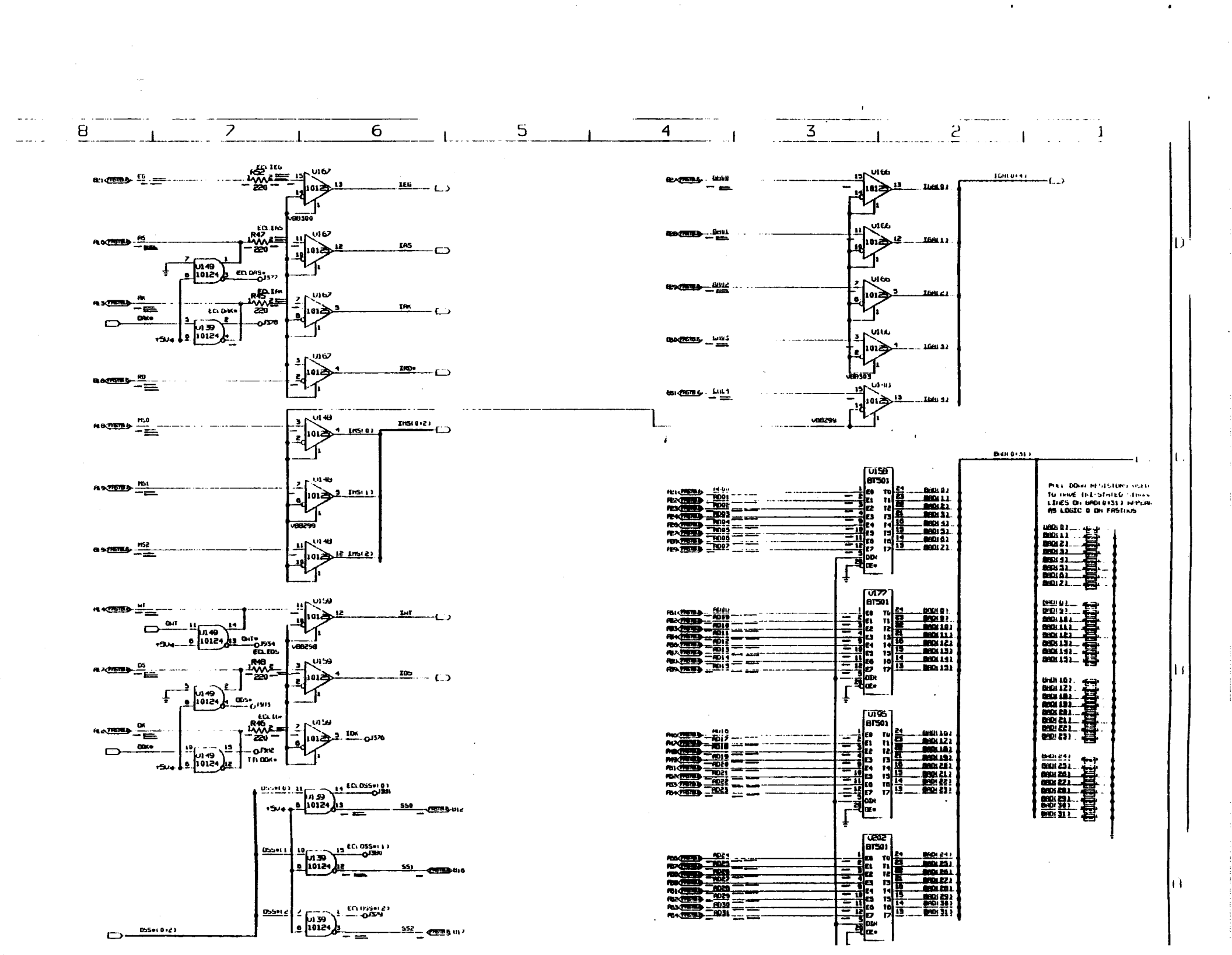

+5V
R50
2.2K
EVENT+
EVBK+
EVBK0
RESET(4)
FIFEF2+
EVBF+
DIN(0:31)
FIFLL+
BADX(0:31)

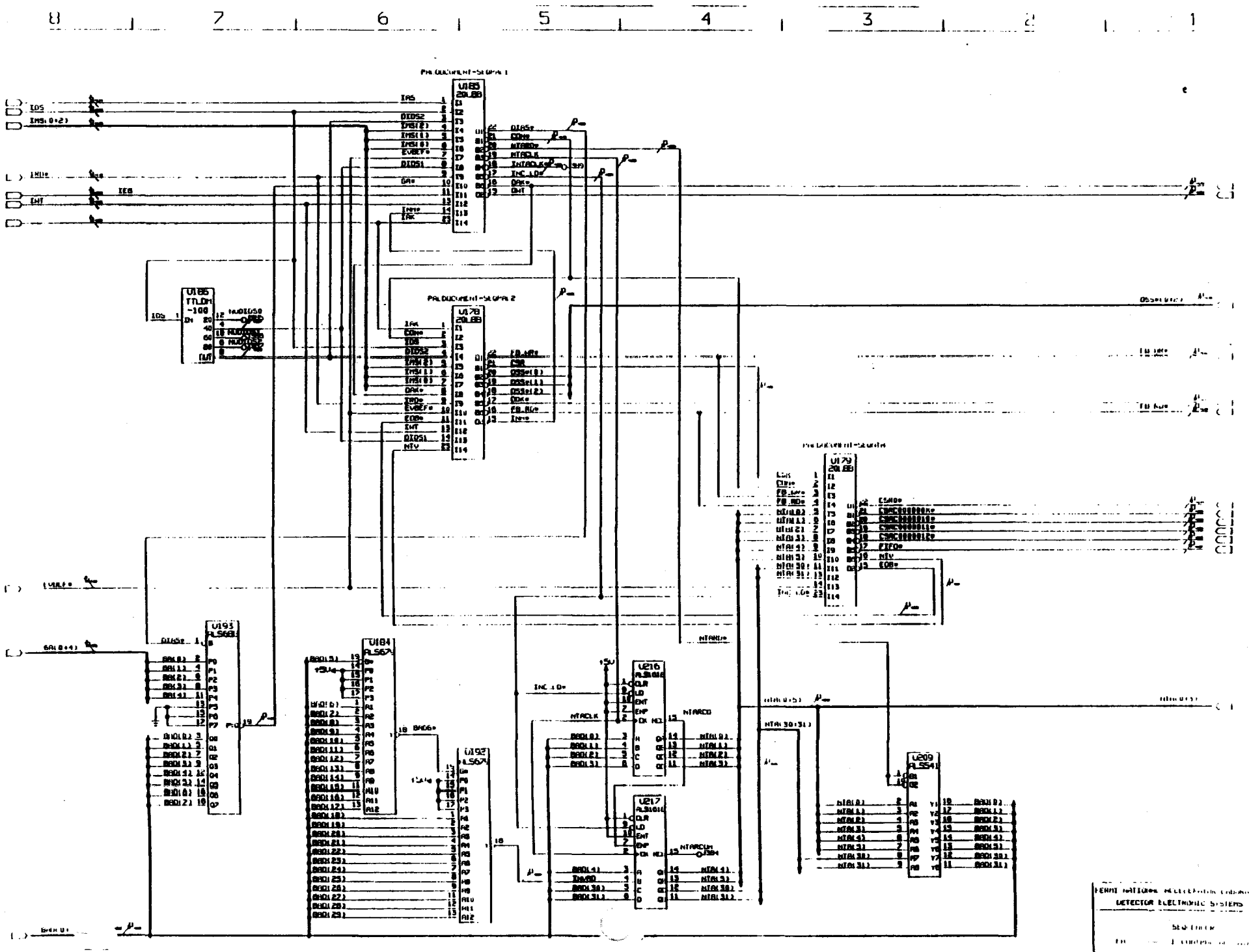

8
7
6
5
4
3
2
1
D
C
B
A
TO FRONT PANEL LED
GROUNDS
NOT USED

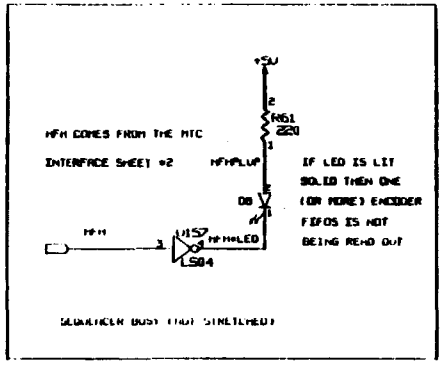


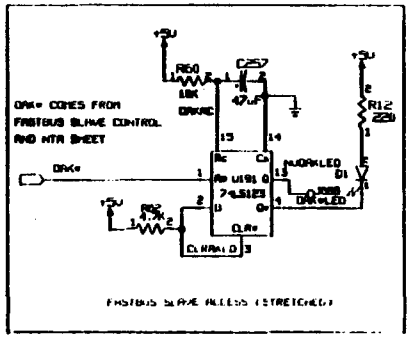
FERMI NATIONAL ACCELERATOR LABORATORY ELECTRONICS/ELECTRICAL DEPARTMENT DETECTOR ELECTRONIC SYSTEMS	
SSD READOUT SEQUENCER BOARD CONTROL AND STATUS REGISTERS	
D 1/19/90	SHEET

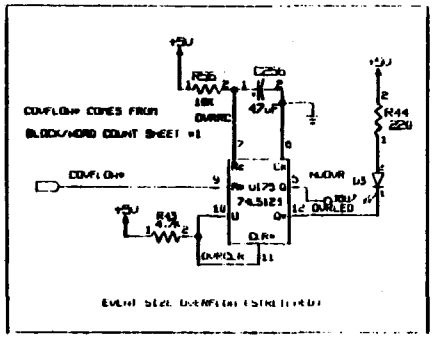

FASTBUS
TRANSLATORS

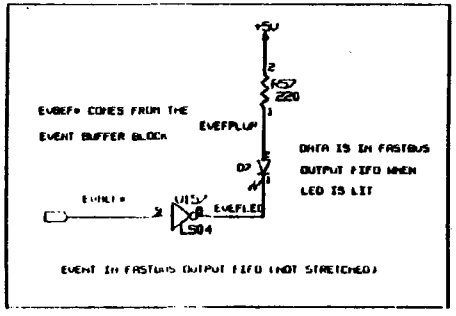


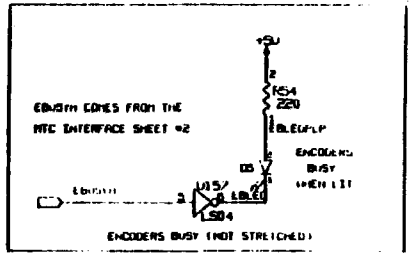

8
7
6
5
4
3
2
1
U185
20L8B
U186
TTLDM
-100
U178
20L8B
U179
20L8B
U193
RLS681
U184
RLS679
U192
RLS679
U216
RLS6818
U217
RLS6818
U209
RLS541
IOS
INHIBIT
EMT
IOS
DIOS2
DIOS1
INTARCO
INTARCLK
INC_LOS
DAK
INH
TAK
FB_WE
CSR
DSSel(0)
DSSel(1)
DSSel(2)
FB_RE
EOB
NTV
FIFO
CSAD
NTARCLK
NTARCO
NTARCLK
NTARCLK
INC_LOS
NTAR30:31
NTAR0:29
BAD0:31
BAD0:15
BAD16:31
DIAS
DIAS2
DIAS1
+5V
+5Vdc
BAD6
BAD16
BAD17
BAD18
BAD19
BAD20
BAD21
BAD22
BAD23
BAD24
BAD25
BAD26
BAD27
BAD28
BAD29
BAD30
BAD31
NTAR0
NTAR1
NTAR2
NTAR3
NTAR4
NTAR5
NTAR30
NTAR31
DETECTOR ELECTRONIC SYSTEMS

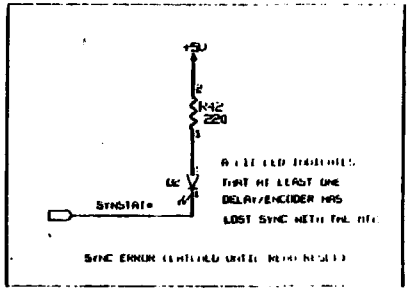


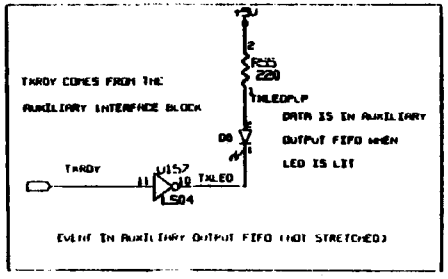


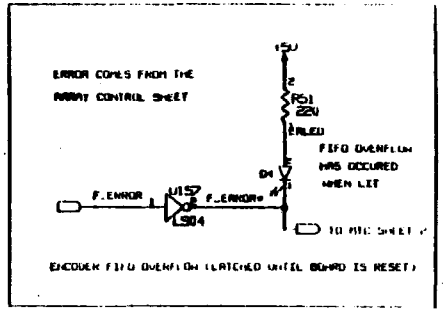




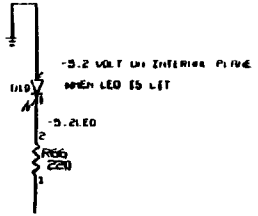


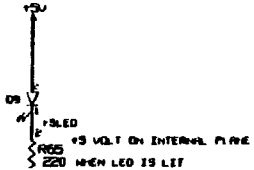






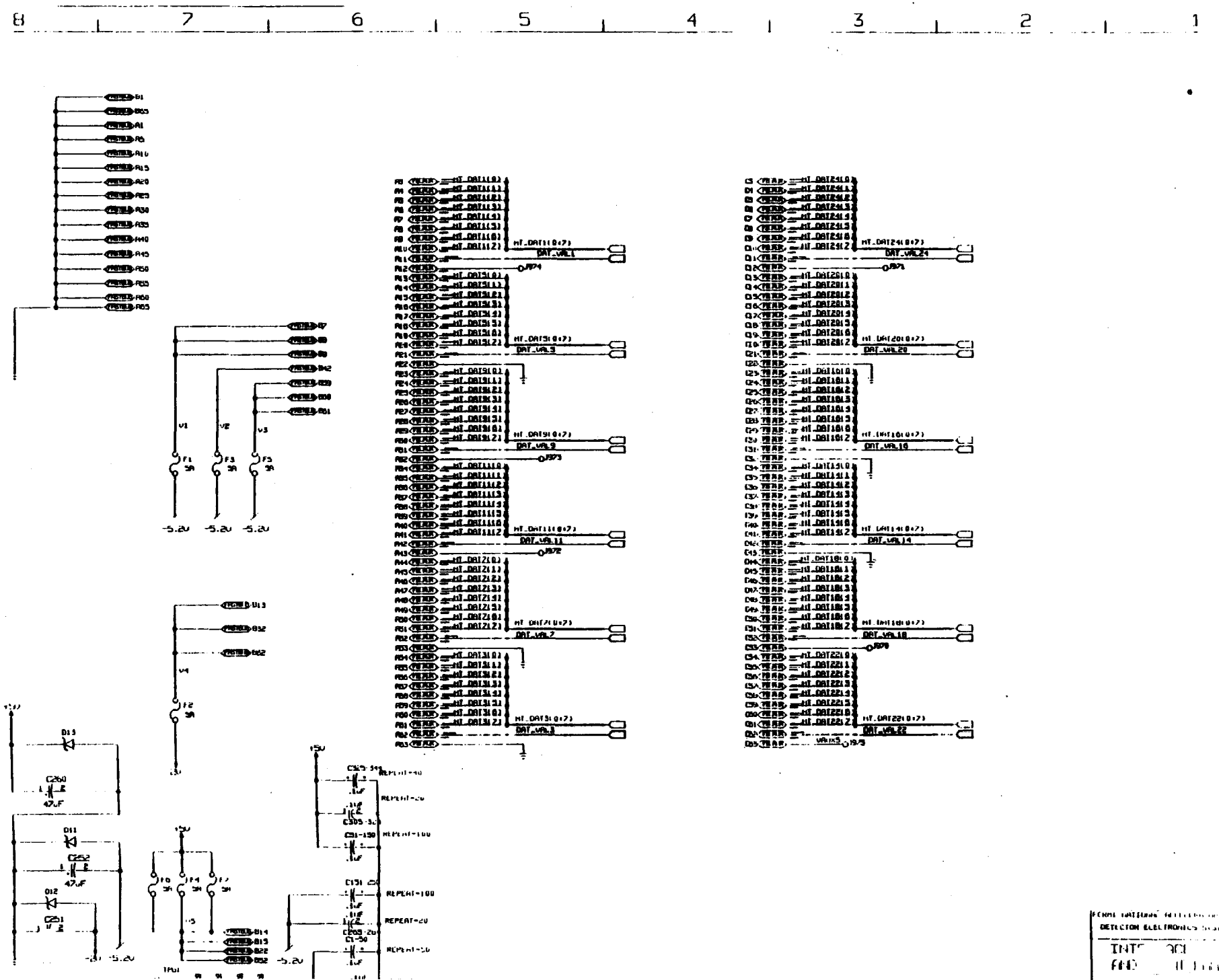
ENCODER FIFO OVERFLOW (LATCHED UNTIL BOARD IS RESET)





FERMI NATIONAL ACCELERATOR LABORATORY
ELECTRONIC/ELECTRICAL ENGINEERING
DETECTOR ELECTRONICS SYSTEMS

SSO READOUT SEQUENCER MODULE
LED DESIGN CIRCUITS


8
7
6
5
4
3
2
1
-5.2V
+5V
REPEAT=100
REPEAT=20
REPEAT=50

APPENDIX B

PROGRAMMABLE LOGIC EQUATIONS

ARIDAD.abl ID ADDRESS ENCODER

```
MODULE ID_ADDRESS_ENCODER
FLAG '-R7'
TITLE 'ANDY ROMERO RD/EE CONTROLS 2-6-90'

ARIDAD DEVICE 'P20V8R';

CLK,RDEN1,RDEN2,RDEN3,RDEN4,RDEN5,RDEN6    PIN  1,23,2,3,4,5,6;
RDEN7,RDEN8,RDEN9,RDEN10,RDEN11,RDEN12     PIN  7,8,9,10,11,14;
IDAD3,IDAD2,IDAD1,IDAD0                    PIN  18,17,16,15;
FBACC                                      PIN  13;

"Standard Symbol Declarations"
X=.X.;
C=.C.;
P=.P.;
Z=.Z.;
"End Standard Symbol Declarations"

VECIN =
[RDEN1,RDEN2,RDEN3,RDEN4,RDEN5,RDEN6,RDEN7,RDEN8,RDEN9,RDEN10,RDEN11,RDEN12];
ID_AD_VEC = [IDAD3,IDAD2,IDAD1,IDAD0];

TRUTH_TABLE ([VECIN] :> [ID_AD_VEC])

[[0,1,1,1,1,1,1,1,1,1,1,1]] :> [[0,0,0,0]];
[[1,0,1,1,1,1,1,1,1,1,1,1]] :> [[0,0,0,1]];
[[1,1,0,1,1,1,1,1,1,1,1,1]] :> [[0,0,1,0]];
[[1,1,1,0,1,1,1,1,1,1,1,1]] :> [[0,0,1,1]];
[[1,1,1,1,0,1,1,1,1,1,1,1]] :> [[0,1,0,0]];
[[1,1,1,1,1,0,1,1,1,1,1,1]] :> [[0,1,0,1]];
[[1,1,1,1,1,1,0,1,1,1,1,1]] :> [[0,1,1,0]];
[[1,1,1,1,1,1,1,0,1,1,1,1]] :> [[0,1,1,1]];
[[1,1,1,1,1,1,1,1,0,1,1,1]] :> [[1,0,0,0]];
[[1,1,1,1,1,1,1,1,1,0,1,1]] :> [[1,0,0,1]];
[[1,1,1,1,1,1,1,1,1,1,0,1]] :> [[1,0,1,0]];
[[1,1,1,1,1,1,1,1,1,1,1,0]] :> [[1,0,1,1]];
[[1,1,1,1,1,1,1,1,1,1,1,1]] :> [[1,1,1,1]];

TEST_VECTORS

([CLK,FBACC,VECIN] -> [ID_AD_VEC])

"[C,1,[X,X,X,X,X,X,X,X,X,X,X,X]] -> [[Z,Z,Z,Z]]; Zs cause unisite to barf"
[C,0,[0,1,1,1,1,1,1,1,1,1,1,1]] -> [[0,0,0,0]];
[C,0,[1,0,1,1,1,1,1,1,1,1,1,1]] -> [[0,0,0,1]];
[C,0,[1,1,0,1,1,1,1,1,1,1,1,1]] -> [[0,0,1,0]];
[C,0,[1,1,1,0,1,1,1,1,1,1,1,1]] -> [[0,0,1,1]];
[C,0,[1,1,1,1,0,1,1,1,1,1,1,1]] -> [[0,1,0,0]];
[C,0,[1,1,1,1,1,0,1,1,1,1,1,1]] -> [[0,1,0,1]];
[C,0,[1,1,1,1,1,1,0,1,1,1,1,1]] -> [[0,1,1,0]];
[C,0,[1,1,1,1,1,1,1,0,1,1,1,1]] -> [[0,1,1,1]];
[C,0,[1,1,1,1,1,1,1,1,0,1,1,1]] -> [[1,0,0,0]];
[C,0,[1,1,1,1,1,1,1,1,1,0,1,1]] -> [[1,0,0,1]];
[C,0,[1,1,1,1,1,1,1,1,1,1,0,1]] -> [[1,0,1,0]];
[C,0,[1,1,1,1,1,1,1,1,1,1,1,0]] -> [[1,0,1,1]];
[C,0,[1,1,1,1,1,1,1,1,1,1,1,1]] -> [[1,1,1,1]];

END ID_ADDRESS_ENCODER
```

```
[C, X,X, 1,1,X,1, [X,X]]->[[0,1]];
[C, X,X, X,X,X,X, [X,X]]->[[1,1]];
[C, X,X, X,X,1,1, [X,X]]->[[1,1]];
[C, X,X, X,X,1,0, [X,X]]->[[0,0]];

END FACU_CONTROL
```

ARTMO.abl GLOBAL TIMEOUT CHIP

```
MODULE GLOBAL_TIMEOUT_CHIP
FLAG '-R7'
TITLE 'ANDY ROMERO RD/EE CONTROLS 2-2-90'

"6-4-90 Because the use of the encoder fifo empty flags for
"       encoder fifo readout initiation was a stupid idea,
"       we will have to modify this chip as part a new encoder
"       fifo readout cycle initiation scheme. See the design note
"       for details on the operation of this new scheme.
"       -Add the signal TRINC (trigger count increment) to pin 16
"        -Move Q2,Q1,Q0 from pins 16,15,14 to pins 15,14,13
"       -TRINC will be the MSB of the state vector


ARTMO DEVICE 'P16V8R';

PIN1,PIN2,PIN3,PIN4    PIN   1,2,3,4;
PIN19,PIN18,PIN17,PIN16,PIN15,PIN14,PIN13  PIN  19,18,17,16,15,14,13;

"Declaration of active levels"

SEQCLK  =  PIN1;
SEQCLKB =  PIN2;
ADR_VAL  =  PIN3;
RESET  =  !PIN4;

ENCOTMO0 = !PIN19;
ENCOTMO1 = !PIN18;
EBUSY   = !PIN17;
TRINC   = PIN16;
Q2      = PIN15;
Q1      = PIN14;
Q0      = PIN13;


"Standard Symbol Declarations"

X=.X.;
C=.C.;
P=.P.;


STATEVEC = [TRINC,Q2,Q1,Q0];

TRINC_HOLD = [0,0,0,0];
TRAP1 = [0,0,0,1];
TRAP2 = [0,0,1,0];
TRAP3 = [0,0,1,1];
TRAP4 = [0,1,0,0];
TRAP5 = [0,1,0,1];
TRAP6 = [0,1,1,0];
TIMEOUT = [0,1,1,1];
WAIT = [1,0,0,0];
S1 = [1,0,0,1];
S2 = [1,0,1,0];
S3 = [1,0,1,1];
S4 = [1,1,0,0];
```

ARLOAD.abl ARRAY CONTROL UNIT STATE MACHINE

```
MODULE FACU_CONTROL
FLAG '-R7'
TITLE 'ANDY ROMERO RD/EE CONTROLS 2-8-90'

"This Chip needs the special reduction technique"

ARLOAD DEVICE 'P20V8R';

PIN1,PIN2,PIN3,PIN4,PIN5,PIN7,PIN8,PIN9  PIN  1,2,3,4,5,7,8,9;
PIN15,PIN16,PIN17,PIN19,PIN20,PIN21,PIN22  PIN 15,16,17,19,20,21,22;

"Declaration of Active Levels"

CLK = PIN1;          PRELOAD0 = !PIN22;
NOTEMP = !PIN2;      PRELOAD1 = !PIN21;
ERROR = PIN3;        X0 = PIN20;
DONE = !PIN4;        X1 = PIN19;
PIPEHOLD = !PIN5;    EVENTEN = !PIN17;
RESET = !PIN7;       SYNCD_NOTEMP = !PIN16;
CLKB = PIN8;         SYNCD_ERROR = PIN15;
CYCLE = !PIN9;

STATEVEC = [X1,X0];

HOLD = [0,0];
FULL_ERROR = [1,0];
PRE_LOAD = [0,1];
EVENT_READOUT = [1,1];


"Standard Symbol Declarations"

X=.X.;
C=.C.;
P=.P.;
Z=.Z.;

"End Standard Symbol Declarations"

EQUATIONS

PRELOAD0 = (
        ((STATEVEC == PRE_LOAD) & !CLKB)
       #(PRELOAD0 & (STATEVEC == PRE_LOAD))
        #(PRELOAD0 & CLKB)
       );

PRELOAD1 = (
        ((STATEVEC == PRE_LOAD) & !CLKB)
       #(PRELOAD1 & (STATEVEC == PRE_LOAD))
        #(PRELOAD1 & CLKB)
       );

EVENTEN = (
        ((STATEVEC == EVENT_READOUT) & !CLKB)
       #(EVENTEN & (STATEVEC == EVENT_READOUT))
        #(EVENTEN & CLKB)
      );
```

```
SYNCD_NOTEMP := NOTEMP;

SYNCD_ERROR := ERROR;


STATE_DIAGRAM STATEVEC

STATE HOLD:;

IF RESET THEN HOLD
ELSE
  IF SYNCD_ERROR THEN FULL_ERROR
  ELSE
    IF (SYNCD_NOTEMP & !PIPEHOLD & !CYCLE) THEN PRE_LOAD
    ELSE HOLD;

STATE FULL_ERROR:;

IF RESET THEN HOLD
ELSE FULL_ERROR;

STATE PRE_LOAD:;

GOTO EVENT_READOUT;

STATE EVENT_READOUT:;

IF (RESET # (DONE & !PIPEHOLD)) THEN HOLD
ELSE EVENT_READOUT;


TEST_VECTORS

"([CLK, ERROR,NOTEMP, PIPEHOLD,CYCLE,DONE,RESET, STATEVEC]->[STATEVEC])"

([PIN1, PIN3,PIN2, PIN5,PIN9,PIN4,PIN7, STATEVEC]->[STATEVEC])

[P, 0,1, 1,1,1,1, [0,0]]->[[X,X]];
[C, 0,1, 1,1,1,0, [X,X]]->[[0,0]];
[P, 0,1, 1,1,1,1, [0,0]]->[[X,X]];
[C, 0,1, 1,1,1,0, [X,X]]->[[0,0]];
[C, 1,1, 1,1,1,1, [X,X]]->[[0,0]];
[C, X,X, X,X,X,1, [X,X]]->[[1,0]];
[P, 0,1, 1,1,1,1, [0,0]]->[[X,X]];
[C, 0,1, 1,1,1,0, [X,X]]->[[0,0]];
[C, 0,0, X,X,X,X, [X,X]]->[[0,0]];
[C, X,X, 1,1,X,1, [X,X]]->[[0,1]];

[P, 0,1, 1,1,1,1, [0,0]]->[[X,X]];
[C, 0,1, 1,1,1,0, [X,X]]->[[0,0]];
[C, 0,0, X,X,X,1, [X,X]]->[[0,0]];
[C, X,X, 1,1,X,1, [X,X]]->[[0,1]];
[C, X,X, X,X,X,X, [X,X]]->[[1,1]];
[C, X,X, X,X,1,1, [X,X]]->[[1,1]];
[C, X,X, 0,X,0,1, [X,X]]->[[1,1]];
[C, X,X, 1,X,0,1, [X,X]]->[[0,0]];

[P, 0,1, 1,1,1,1, [0,0]]->[[X,X]];
[C, 0,1, 1,1,1,0, [X,X]]->[[0,0]];
[C, 0,0, X,X,X,1, [X,X]]->[[0,0]];
```

```
S5 = [1,1,0,1];
S6 = [1,1,1,0];
TRAP7 = [1,1,1,1];

EQUATIONS

EBUSY = (STATEVEC != WAIT);

ENCOTMO0 = (
        (!SEQCLKB & (STATEVEC == TIMEOUT))
       #(ENCOTMO0 & (STATEVEC == TIMEOUT))
        #(ENCOTMO0 & SEQCLKB)
        );

ENCOTMO1 = (
        (!SEQCLKB & (STATEVEC == TIMEOUT))
       #(ENCOTMO1 & (STATEVEC == TIMEOUT))
        #(ENCOTMO1 & SEQCLKB)
        );


STATE_DIAGRAM STATEVEC

STATE WAIT:;

IF RESET THEN WAIT
ELSE
IF ADR_VAL THEN S1
ELSE WAIT;

STATE S1:;

IF RESET THEN WAIT
ELSE S2;

STATE S2:;

IF RESET THEN WAIT
ELSE S3;

STATE S3:;

IF RESET THEN WAIT
ELSE S4;

STATE S4:;

IF RESET THEN WAIT
ELSE S5;

STATE S5:;

IF RESET THEN WAIT
ELSE S6;

STATE S6:;

IF RESET THEN WAIT
ELSE TIMEOUT;

STATE TIMEOUT:;
```

```
IF RESET THEN WAIT
ELSE TRINC_HOLD;


STATE TRINC_HOLD::;

GOTO WAIT;


STATE TRAP1::;
GOTO WAIT;

STATE TRAP2::;
GOTO WAIT;

STATE TRAP3::;
GOTO WAIT;

STATE TRAP4::;
GOTO WAIT;

STATE TRAP5::;
GOTO WAIT;

STATE TRAP6::;
GOTO WAIT;

STATE TRAP7::;
GOTO WAIT;

TEST_VECTORS

([PIN1,PIN3,PIN4,      PIN16,PIN15,PIN14,PIN13]->[PIN16,PIN15,PIN14,PIN13])


[P,0,0,   1,0,0,0]->[X,X,X,X];
[C,0,0,   1,0,0,0]->[X,X,X,X];
[C,1,1,   X,X,X,X]->[1,0,0,1];
[C,0,1,   X,X,X,X]->[1,0,1,0];
[C,0,1,   X,X,X,X]->[1,0,1,1];
[C,0,1,   X,X,X,X]->[1,1,0,0];
[C,0,1,   X,X,X,X]->[1,1,0,1];
[C,0,1,   X,X,X,X]->[1,1,1,0];
[C,0,1,   X,X,X,X]->[0,1,1,1];
[C,0,1,   X,X,X,X]->[0,0,0,0];
[C,0,1,   X,X,X,X]->[1,0,0,0];

END GLOBAL_TIMEOUT_CHIP
```

BWCOUNT.abl BLOCK COUNTER AND WORD COUNTER CONTROL

```
MODULE BLK_AND_WRD_COUNTER_CONTROL
FLAG '-R7'
TITLE 'ANDY ROMERO RD/EE CONTROLS 2-13-90'

BWCOUNTC DEVICE 'P20V8R';

PIN1,PIN23,PIN2,PIN3,PIN4,PIN5,PIN6,PIN7,PIN8,PIN9,PIN10,PIN14  PIN  1,23,2,3,4,5,6,7,8,9,10,14;
PIN15,PIN17,PIN18,PIN19,PIN20,PIN21,PIN22  PIN 15,17,18,19,20,21,22;

"Declaration of Active Levels"

CLK = PIN1;        CNT_RD = !PIN22;
BLKWRD = !PIN23;   INC_BC = !PIN21;
CSRC = !PIN2;      BCWRT = !PIN20;
FB_WR = !PIN3;     WCS0 = PIN19;
CLKB = PIN4;       WCS1 = PIN18;
EVENTEN = !PIN5;   INC_WC = !PIN17;
DONE = !PIN6;      WCDONE = !PIN15;
RESET = !PIN7;
FB_RD = !PIN8;
COVFLOW = !PIN9;
NOTRUNC = !PIN10;
PIPEHOLD =!PIN14;

"End Declaration of Active Levels"


"Standard Symbol Declarations"

X=.X.;
C=.C.;
P=.P.;
Z=.Z.;

"End Standard Symbol Declarations"

WCSELVEC = [WCS1, WCS0];
CLEAR_WC = [0,0];
COUNT_DOWN = [0,1];
LOAD_WC = [1,0];
COUNT_UP = [1,1];

EQUATIONS

CNT_RD = CSRC & FB_RD;

INC_BC = (
        (BLKWRD)
        #(CSRC & FB_WR)
       );

BCWRT = (
        (CSRC & FB_WR)
       #(BCWRT & INC_BC)
      );

WHEN (RESET # BLKWRD) THEN WCSELVEC = CLEAR_WC;
ELSE WHEN (CSRC & FB_WR) THEN WCSELVEC = LOAD_WC;
```

BWGATEC.abl STAGE 2 GATING CHIP

```
MODULE STAGE_2_GATING_CHIP
FLAG '-R1'
TITLE 'ANDY ROMERO RD/EE CONTROLS 2-13-90'

BWGATEC DEVICE 'P20V8R';

PIN2,PIN3,PIN4,PIN7,PIN8,PIN9  PIN  2,3,4,7,8,9;
PIN22,PIN21,PIN20,PIN19,PIN18,PIN17  PIN  22,21,20,19,18,17;

"Declaration of Active Levels"

CLKB = PIN2;        DMHIW = PIN22;
EVENTEN = !PIN3;    DMLOW = PIN21;
PIPEHOLD = !PIN4;   P2W = PIN20;
                    DMLOGAT = !PIN19;
                    DMHIGAT = !PIN18;
S1 = PIN7;          P2GAT = !PIN17;
S0 = PIN8;
DONE = !PIN9;

"End Declaration of Active Levels"


"Standard Symbol Declarations"

X=.X.;
C=.C.;
P=.P.;
Z=.Z.;

"End Standard Symbol Declarations"

EQUATIONS

P2W = P2GAT & CLKB & !PIPEHOLD;

DMHIW = DMHIGAT & CLKB & !PIPEHOLD;

DMLOW = DMLOGAT & CLKB & !PIPEHOLD;

P2GAT = (
      (EVENTEN & !CLKB & !DONE)
      #(P2GAT & CLKB)
      );

DMHIGAT = (
       (S1 & !CLKB)
       #(S1 & DMHIGAT)
       #(CLKB & DMHIGAT)
       );

DMLOGAT = (
       (S0 & !CLKB)
       #(S0 & DMLOGAT)
       #(CLKB & DMLOGAT)
       );

END STAGE_2_GATING_CHIP
```

```
FIFGAT = (
        (FWRTRDY & !CLKB)
       #(FIFGAT & FWRTRDY)
       #(FIFGAT & CLKB)
      );

FWRTRDY := (
        (DMHIGAT)
       #(FWRTRDY & PIPEHOLD0) "Save the state of FWRTRDY if Pipehold"
       );

HOLDPIP := (
        (AUXSEL & AUXHALF)
       #(FASTSEL & EVBHF)
       );

PIPEHOLD0 = (
        (((!AUXSEL & STOPEVBW) # HOLDPIP) & !CLKB)
       #(PIPEHOLD0 & ((!AUXSEL & STOPEVBW) # HOLDPIP))
       #(PIPEHOLD0 & CLKB)
       );

PIPEHOLD1 = (
        (((!AUXSEL & STOPEVBW) # HOLDPIP) & !CLKB)
       #(PIPEHOLD1 & ((!AUXSEL & STOPEVBW) # HOLDPIP))
       #(PIPEHOLD1 & CLKB)
       );

PIPEHOLD2 = (
        (((!AUXSEL & STOPEVBW) # HOLDPIP) & !CLKB)
       #(PIPEHOLD2 & ((!AUXSEL & STOPEVBW) # HOLDPIP))
       #(PIPEHOLD2 & CLKB)
       );

END OUTPUT_FIFO_WRITE_CONTROL
```

BWSTATE.abl PIPELINE CONTROL STATE MACHINE

```
MODULE PIPELINE_CONTROL_STATE_MACHINE

FLAG   '-r7'

TITLE  'Andy Romero  RD/EE Controls 2-13-90'


"Modification 7-24-90
"Modify this chip to include an output that will allow
"one event to be written to the Fastbus Event Buffer and then
"prevent further writes until the previous event has been read from the
"Event Buffer by FASTBUS. This circuit will only allow complete events to
"be writen to the event buffer. The following signals were added
"FBFEF,STOPEVBW.


BWSTATEM DEVICE        'P20V8R';

PIN1,PIN3,PIN4,PIN5,PIN6,PIN7,PIN8,PIN9  PIN  1,3,4,5,6,7,8,9;
PIN19,PIN18,PIN17,PIN16,PIN15  PIN  19,18,17,16,15;

"DECLARATION OF ACTIVE LEVELS"

CLK = PIN1;        S3 = PIN19;
NOTRUNC = !PIN3;   S2 = PIN18;
COVFLOW = !PIN4;   S1 = PIN17;
RESET = !PIN5;     S0 = PIN16;
DONE = !PIN6;      STOPEVBW = PIN15;
EVENTEN = !PIN7;
PIPEHOLD = !PIN8;
FBFEF = !PIN9;

"END DECLARATION OF ACTIVE LEVELS"

"STANDARD SYMBOL DECLARATIONS"

X=.X.;
C=.C.;
P=.P.;
Z=.Z.;

"END STANDARD SYMBOL DECLARATIONS"

@INCLUDE 'E:\\PROJECTS\\SSDSEQ\\PALS\\PIPSTATE.TBL'

"Declarations concerning STOPEVBW state machine
ALLOW = 0; "allow FASTBUS fifo writes
PREVENT = 1; "prevent  FASTBUS fifo writes

STATE_DIAGRAM STATEVEC

STATE WAIT:;

IF RESET THEN WAIT
ELSE IF DONE THEN LASTLO
ELSE IF (EVENTEN & !DONE) THEN DATLO
ELSE WAIT;
```

```
GOTO WAIT;

STATE TRAP3:;

GOTO WAIT;

STATE TRAP4:;

GOTO WAIT;

STATE TRAP5:;

GOTO WAIT;

STATE TRAP6:;

GOTO WAIT;

STATE_DIAGRAM STOPEVBW

STATE ALLOW:
IF ((STATEVEC == SET_CNT) & !FBFEF) THEN PREVENT
ELSE ALLOW;


STATE PREVENT:
IF ((STATEVEC == WAIT) & FBFEF) THEN ALLOW
ELSE PREVENT;

END PIPELINE_CONTROL_STATE_MACHINE
```

```
!CYCLE = (
        ((STATEVEC == WAIT) & !CLKB)
        #((STATEVEC == WAIT) & !CYCLE)
        #(CLKB & !CYCLE)
        );

WCNT_EN = (
        ((STATEVEC == FLUSH) & !CLKB)
        #((STATEVEC == FLUSH) & WCNT_EN)
        #(CLKB & WCNT_EN)
        );

END PIPELINE_STATE_DECODER
```

```
AUXSEL := 1; "KLUDGE FOR ABEL telling ABEL to make the cell registered.
AUXSEL.C = 1; "KLUDGE FOR ABEL telling ABEL to not clock with pin 1. (I really
        "only want to use the preset and clear functions and don't want
        "to clock the register at all.)


AUXSEL.RE = /CSR0 * /FB_WR * BAD7IN "Enable loading of Auxiliary output FIFO by
      + /RESET;                     " writing a 1 to CSR0<7>. Also enable at
                                    " reset time. /AUXSEL is active low.

AUXSEL.PR = /CSR0 * /FB_WR * BAD23IN;"Disable loading of FASTBUS output FIFO by
                                    " writing a 1 to CSR0<23>.
                                    " writing a 1 to CSR0<23>.

BAD7 = /AUXSEL.Q; " Status of Auxiliary Output FIFO loading is read
             " back on CSR0<7> via BAD(7)

BAD7.OE = /CSR0 * /FB_RD; " Enable active high status bit onto BAD(7)
                    " that indicates if the Auxiliary output FIFO is
                    " enabled/disabled as part of CSR0 read operation.

NOTRUNC := 1; "KLUDGE FOR ABEL telling ABEL to make the cell registered.
NOTRUNC.C = 1; "KLUDGE FOR ABEL telling ABEL to not clock with pin 1. (I really
        "only want to use the preset and clear functions and don't want
        "to clock the register at all.)


NOTRUNC.PR = /CSR0 * /FB_WR * BAD8   " Disable active low NOTRUNC which is
      + /RESET;                      " used to prevent truncating events at
                                     " 255 when it is not enabled.

NOTRUNC.RE = /CSR0 * /FB_WR * BAD24IN; " Enable active low NOTRUNC so that
                                     " events will NOT be truncated at 255.

BAD8      = NOTRUNC.Q;               " Status of NOTRUNC will be read back BAD(8)
                          " of CSR0.

BAD8.OE = /CSR0 * /FB_RD; " Enable active high status bit onto BAD(8)
                    " that indicates if the Overflow Truncation is
                    " enabled/disabled as part of CSR0 read operation.

SYNSTAT.C = /TSYNERR; " Clock the SYNC status F/F with TTL level Sync Error

/SYNSTAT := 1; " When clocked by TSYNERR, assert /SYNSTAT

SYNSTAT.PR = SYNRESFF.Q * FB_RD "Clear ACTIVE LOW /SYNSTAT, hence not .RE
      + /RESET;


SYNRESFF := 1; "The Sync Error Reset F/F will have a 1 clocked in when
           "FASTBUS reads the CSR at C000_0012. This is in preparation
           "for clearing the F/F at the end of the read.

SYNRESFF.C = /C000_0012 * /FB_RD; "Clock the Sync Error Reset F/F when FASTBUS
                          "reads from CSR at C000_0012.

SYNRESFF.RE = SYNRESFF.Q * /C000_0012 * FB_RD "Clear the Sync Error Reset F/F
      + /RESET;                       "when FASTBUS finishes that read
                                      "that set the F/F originally.
```

ENCEN.abl ENCODER FIFO CIRCUIT CONTROL

```
MODULE              Enc_Fifo_Ckt_Pal

FLAG                '-r7'

TITLE               ' Andy Romero  RD/EE Controls'

ENCEN               DEVICE          'P20V8R';

PIN1,PIN2,PIN3,PIN4,PIN6,PIN8,PIN9,PIN10,PIN11  PIN  1,2,3,4,6,8,9,10,11;
PIN14,PIN15,PIN16,PIN17,PIN18,PIN19  PIN  14,15,16,17,18,19;
PIN20,PIN21,PIN22   PIN  20,21,22;

"STANDARD SYMBOL DECLARATIONS

X=.X.;
C=.C.;
P=.P.;


"DECLARATION OF ACTIVE LEVELS AND DESCRIPTION OF PIN FUNCTIONS

SQCLK = PIN1;          FW = PIN22;
SQCLKB = PIN2;         EBUSY = !PIN21;
DT_VAL = PIN3;         FERR = !PIN20;
ENCOTMO = !PIN4;       EFSM2 = PIN19;
DLYRST = !PIN6;        EFSM1 = PIN18;
PRELOAD = !PIN8;       EFSM0 = PIN17;
PIPEHOLD = !PIN9;      RDGAT = !PIN16;
RDEN = !PIN10;         FRD = PIN15;
FCLK = PIN11;
FF = !PIN14;

EFSTATE = [EFSM2,EFSM1,EFSM0];

S0 = [0,0,0];
S1 = [0,0,1];
S2 = [0,1,0];
S3 = [0,1,1];
S4 = [1,0,0];
S5 = [1,0,1];
S6 = [1,1,0];
S7 = [1,1,1];


EQUATIONS

FW = (
     (
       ( SQCLKB & EBUSY ) "Normal data write
      # ( SQCLKB & ENCOTMO & ( EFSTATE==S0 ) ) "Timeout EOE write
     )
     & (!FF & !FERR) "Fifo not full"
    )
  # (DLYRST); "Hold high on reset

EBUSY =  (!SQCLKB & DT_VAL) "Clk qualify EBUSY for glitch free gating of FW"
     # (EBUSY & DT_VAL) "Latch gating while DT_VAL active"
     # (EBUSY & ((EFSTATE==S1) # (EFSTATE==S3)))"Insure that EOE gets written"
```

```
STATE S7::

Goto S0;


TEST_VECTORS

"Test of Basic State Machine Operation"

([PIN1,    PIN3,  PIN4,  PIN6,   EFSTATE] -> [EFSTATE])
"([SQCLK,   DT_VAL, ENCOTMO, DLYRST,   EFSTATE] -> [EFSTATE])

[P, 0,1,1,  S0] -> [X];
[C, 0,1,1,  X] -> [S0];
[C, 1,1,1,  X] -> [S1];
[C, 1,0,1,  X] -> [S3];
[C, X,0,1,  X] -> [S2];
[C, X,X,X,  X] -> [S0];
[C, 0,1,1,  X] -> [S0];
[C, 1,1,1,  X] -> [S1];
[C, 0,1,1,  X] -> [S5];
[C, 0,0,1,  X] -> [S4];
[C, X,X,X,  X] -> [S0];
[C, 1,1,X,  X] -> [S0];
[C, 1,1,1,  X] -> [S1];
[C, 0,0,1,  X] -> [S0];

"Test of all transitions out of all states

"([PIN1,    PIN3,  PIN4,  PIN6,   EFSTATE] -> [EFSTATE])
"([SQCLK,   DT_VAL, ENCOTMO, DLYRST,   EFSTATE] -> [EFSTATE])

"S0
[P, 0,1,1,  S0] -> [X];
[C, 0,1,1,  X] -> [S0];
[C, X,X,0,  X] -> [S0];
[C, 1,1,1,  X] -> [S1];

"S1
[P, 1,1,1,  S1] -> [X];
[C, 1,1,1,  X] -> [S1];
[P, 1,1,1,  S1] -> [X];
[C, 1,0,1,  X] -> [S3];
[P, 1,1,1,  S1] -> [X];
[C, X,X,0,  X] -> [S0];
[P, 1,1,1,  S1] -> [X];
[C, 0,0,1,  X] -> [S0];
[P, 1,1,1,  S1] -> [X];
[C, 0,1,1,  X] -> [S5];

"S2
[P, 0,1,1,  S2] -> [X];
[C, X,X,X,  X] -> [S0];

"S3
[P, 1,1,1,  S3] -> [X];
[C, 1,1,1,  X] -> [S3];
[P, 1,1,1,  S3] -> [X];
[C, X,X,0,  X] -> [S2];
[P, 1,1,1,  S3] -> [X];
[C, 0,1,1,  X] -> [S2];
```

"this because there is no maximum limit for the width of the active low Fifo
"write signal

"IMPORTANT NOTE xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

"When designing the chips that generate the following signals, insure that
"they are only allowed to change state when FCLK or SQCLK are low.
"ENCOTMO, FF (determined by fifo looks ok), PRELOAD, PIPEHOLD

"IMPORTANT NOTE xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

""Redundant (glitch prevent) terms were added to the equations for EBUSY and
"and RDGAT; these terms will be eliminated by the ABEL Reduce program if any
"reduction level greater than -r1 is used. This would not be a problem except
"that the state machine (EFSM <2:0>) requires extensive reduction (-r7
"is highly recomended). Here is how to get around this problem.

" 1)Compile this file, ENCFIFO.ABL, with -r7 reduction
" 2)Copy ENCFIFO.ABL to TEMPFILE.ABL
"
" DO NOT DELETE ENCFIFO.ABL IT IS THE MASTER SOURCE DOCUMENT !!!!
"
" 3)Open TEMPFILE.ABL, change the flag statement to activate -r1
" reduction. Delete the state machine description from TEMPFILE.ABL.
" Paste the reduced equations from ENCFIFO.DOC for the state terms
" (EFSM <2:0>) into the equations section of TEMPFILE.ABL.
"
" 4)Compile TEMPFILE.ABL to obtain a glitch free JED file
"
" 5)Delete TEMPFILE.*
"

"INPUTS

"SQCLK = PIN1;

"SQCLK is a clock signal derived from the Sequencer CLK3 signal; SQCLK is used
"to clock a small state machine within this pal.

"SQCLKB = PIN2;

"SQCLKB is the same signal as SQCLK; it provides a waveform of suitable duty
"cycle for writing the Encoder Fifos.

"DT_VAL = PIN3;

"(ACTIVE HIGH) DT_VAL is the Data Valid signal from the Delay Encoder
"Module.

"ENCOTMO = !PIN4;

"(ACTIVE LOW) The Sequencer module sends the Delay encoder module an Address
"Valid signal the Delay Encoder module responds, if it has data to send
"to the Sequencer module, by activating (and holding while it is sending data)
"the Data Valid signal. If the Delay Encoder has no data to send it does not
"issue a Data Valid. 300ns (8 SQCLK cycles) after Address Valid is sent, the
"FACU activates the ENCOTMO signal. If this pal has not received a Data Valid
"before ENCOTMO becomes active an EOE word is written into the Encoder Fifo

"DLYRST = !PIN6;

```
"read signal.

"FRD = PIN15;

"(ACTIVE HIGH) Encoder Fifo read
"signal


END Enc_Fifo_Ckt_Pal
```

PALFIF2.abl FIFO CONTROL FOR AUXILIARY INTERFACE

```
MODULE _PALFIF2 flag '-r3'

TITLE
'FIFO CONTROL FOR AUXILIARY INTERFACE TO TAXI '

    PALFIF2         DEVICE          'P16V8R'; "Use a 16R6B for 16 MHZ. 386

" CONSTANTS:

    ON      =       1;
    OFF     =       0;
    H       =       1;
    L       =       0;
    X       =       .X.;    " ABEL  don't care symbol
    C       =       .C.;    " ABEL clocking input symbol

" Pin names:
                                "INPUT PINS
    ACK     pin     2;          "ACKNOWLEDGE FROM TAXI
    CLOCK           pin     3;          "CLOCK
    AUXOR           pin     4;          "AUXILIARY output READY
    TXRDY           pin     5;          "TXRDY

    SOENA           pin     6;          "SHIFT OUT ENABLE
    STR     pin     7;          "STROBE FROM SEQUENCER
    NU8     pin     8;          "NOT USED 8
    NU9     pin     9;          "NOT USED 9

    CLK     pin     1;          "Clock
    NU11    pin     11;         "NOT USED 11
    TRIG    pin     12;         "TRIGGER ACK ONESHOT
    NU19    pin     19;         "NOT USED 19

                                "OUTPUT PINS
    STROBE          pin     18;         "STROBE TO TAXI
    NU17    pin     17;         "NOT USED 17
    NU16    pin     16;         "NOT USED 16
    TXQ     pin     15;         "OUTPUT READY STATE Q
    AUXRNOT         pin     14;         "READ OF FIFO
    NU13    pin     13;         "NOT USED 13

equations

STROBE = ((STR & !CLOCK) # (!CLOCK & !AUXRNOT & SOENA)
     # (STROBE & !CLOCK));
TXQ = ((CLOCK & TXQ) # (TXRDY & CLOCK));
AUXRNOT =!((TXRDY & SOENA & TXQ) # (!AUXRNOT & TXQ));
TRIG = (TXRDY & SOENA & TXQ);

END _PALFIF2
```

RDEN1.abl SMALL END-OF-EVENT DECODER

```
MODULE SMALL_EOE_DECODER
FLAG '-R7'
TITLE 'ANDY ROMERO RD/EE CONTROLS 2-6-90'

RDEN1 DEVICE 'P18N8';

EVENTEN,EOE1,EOE2,EOE3,EOE4,EOE5,EOE6,EOE7,EOE8  PIN   1,2,3,4,5,6,7,8,9;
READEN1,READEN2,READEN3   PIN   19,18,17;
READEN4,READEN5,READEN6   PIN   16,15,14;
READEN7,READEN8           PIN   13,12;

"Standard Symbol Declarations"

X=.X.;
C=.C.;
P=.P.;

"End Standard Symbol Declarations"

VECTOR = [EVENTEN,EOE1,EOE2,EOE3,EOE4,EOE5,EOE6,EOE7,EOE8];

EQUATIONS

!READEN1 = (VECTOR==[0,1,X,X,X,X,X,X,X]);
!READEN2 = (VECTOR==[0,0,1,X,X,X,X,X,X]);
!READEN3 = (VECTOR==[0,0,0,1,X,X,X,X,X]);
!READEN4 = (VECTOR==[0,0,0,0,1,X,X,X,X]);
!READEN5 = (VECTOR==[0,0,0,0,0,1,X,X,X]);
!READEN6 = (VECTOR==[0,0,0,0,0,0,1,X,X]);
!READEN7 = (VECTOR==[0,0,0,0,0,0,0,1,X]);
!READEN8 = (VECTOR==[0,0,0,0,0,0,0,0,1]);

TEST_VECTORS

([VECTOR] ->
[READEN1,READEN2,READEN3,READEN4,READEN5,READEN6,READEN7,READEN8])

[[1,X,X,X,X,X,X,X,X]] -> [1,1,1,1,1,1,1,1];
[[0,1,X,X,X,X,X,X,X]] -> [0,1,1,1,1,1,1,1];
[[0,0,1,X,X,X,X,X,X]] -> [1,0,1,1,1,1,1,1];
[[0,0,0,1,X,X,X,X,X]] -> [1,1,0,1,1,1,1,1];
[[0,0,0,0,1,X,X,X,X]] -> [1,1,1,0,1,1,1,1];
[[0,0,0,0,0,1,X,X,X]] -> [1,1,1,1,0,1,1,1];
[[0,0,0,0,0,0,1,X,X]] -> [1,1,1,1,1,0,1,1];
[[0,0,0,0,0,0,0,1,X]] -> [1,1,1,1,1,1,0,1];
[[0,0,0,0,0,0,0,0,1]] -> [1,1,1,1,1,1,1,0];


END SMALL_EOE_DECODER
```

SAMTX.abl TRANSMIT LOGIC FOR AUXILIARY INTERFACE

```
MODULE SAMTX_EMULATOR flag '-r3'

TITLE
'SEQUENCER AUXILIARY SAMTX PAL '

    SAMTX       DEVICE      'P16V8R'; "Use a SAM CHIP IN REAL BOARD

" CONSTANTS:

    ON      =       1;
    OFF     =       0;
    H       =       1;
    L       =       0;
    X       =       .X.;    " ABEL  don't care symbol
    C       =       .C.;    " ABEL clocking input symbol

" State definitions :

    ACTIVE              =       ^b000;  "Check for Stopreq and TXRDY
    NU1         =       ^b001;  "NOT USED
    NU2         =       ^b010;  "NOT USED
    NU3         =       ^b011;  "NOT USED
    INIT0       =       ^b100;  "Setup INIT and clear error status
    INIT1       =       ^b101;  "Send INIT and load loop count 20us.
                                "Waiting for RXINIT or timeout
    INQ0        =       ^b110;  "CLR status and setup INQ
    INQ1        =       ^b111;  "Load wait timer and send INQ
                                "Waiting for RXRDY,/RXNRDY or timeout


" Pin names:
                                "INPUT PINS
    TXRDY       pin     2;      "TRANSMITER READY data in fifo
    RXRDY       pin     4;      "READY from reciever
    RXNRDY      pin     5;      "NOT READY from reciever
    AUXERR      pin     7;      "ERROR on link

    STOPREQ     pin     6;      "STOP request form RX
    RXINIT      pin     3;      "Reciever has been initialized
    TMO   pin   8;      "20 microsec Timout
    NRESET      pin     9;      "RESET onmain board

    CLK2  pin   1;      "Clock
    OE    pin   11;     "output enable

                                "OUTPUT PINS
    STRout      pin     15;     "Control strobe
    SOENA       pin     12;     "Shift out and strobe enable
    SCLR  pin   17;     "Status register clear
    LINKRDY     pin     19;     "ready for data transfer
    SEQ   pin   18;     "SEQUENCE state control
    S0    pin   16;     "STATE code bit 0
    C1    pin   14;     "Control code bit 1
    C0    pin   13;     "Control code bit 0

"MAINSTATE definition.

    MAINSTATE = [C1, C0, S0];
```

```
        state NU1:                      "unused state - should not occur
         SEQ := OFF;
          SCLR := ON;
          STRout := OFF;
          SOENA := OFF;
          LINKRDY := OFF;
          if !NRESET then INIT0         "reset to INITstate
          else NU2;                     "if entered go to INITstate

        state NU2:                      "unused state - should not occur
         SEQ := OFF;
          SCLR := ON;
          STRout := OFF;
          SOENA := OFF;
          LINKRDY := OFF;
          if !NRESET then INIT0         "reset to INITstate
          else NU3;                     "if entered go to INITstate

        state NU3:                      "unused state - should not occur
         SEQ := OFF;
          SCLR := ON;
          STRout := OFF;
          SOENA := OFF;
          LINKRDY := OFF;
          if !NRESET then INIT0         "reset to INITstate
          else INIT0;                   "if entered go to INITstate

    end SAMTX_EMULATOR;
```

```
" CHANGE HISTORY:
" 4-30-90    : Pin 14 was changed from EMPTY to EVBEF.
" 4-30-90    : The device type was changed from 20L8 to 20V8.
" 6-05-90    : FB_RD & !EVBEF was added to FIFO equation.
" 6-20-90    : EOB was modified to react to a change on the EVBEF signal
"            : only when FB_RD is inactive in order to prevent EOB being
"            : issued DURING the read of the last word in the FIFO.
" 6-20-90    : EOB before change:      EOB = CON & !CSR & EVBEF
"                                          # CON & CSR & INC & NTV  ;
" 6-20-90    : The p-term (FIFO * FB_RD) was added to FIFO to latch it.

END SEQ_NTA
```

```
            "Block transfer DS dn
        # CON & INC_LD & !INH & IRD & !IMS2 & IMS1 & !IMS0
            & IDS & !DIDS2 & !NTACLK   "NTA read cycle
        # CON & INC_LD & !INH & !IMS2 & !IMS1 & !IMS0
            & IDS & !DIDS2 & !NTACLK;  "Single R/W cycle

"INCrement or LoaD the NTA

 INC_LD = !(!(CON & !INH & IMS0 & IDS & DIDS2) & !INC_LD "Set if blk
     # !IRD & !IMS2 & IMS1 & !IMS0 & IDS & !DIDS2 "Reset on NTA write
     # !IMS2 & !IMS0 & NTACLK  "Reset on NTA or Single & after NTACLK
     # !CON ) ;        "Reset when disconnected

"Output AK

 OAK = CON & !IAK & !IWT                    "Send AK if not broadcast address
     " Removed !BCADD from the first p-term on 4-30-90. The Sequencer
     " doesn't support broadcast
     # OAK & CON                    "Latch until CON goes away
     # OAK & IWT                    "Hold if WaiT is asserted
     # OAK & DIDS2  ;                    "Stretch until DK is off

OWT = DIAS & CON & !IMS2 & !IMS1 & !IMS0 & !OAK & EVBEF
   + OWT & DIAS & EVBEF; "Assert WAIT when the FIFO is empty at address
                         "strobe time and data space is being addressed.
                         "Latch until FIFO goes not empty or master
                         "disconnects by removing AS.

" CHANGE HISTORY:
" 4-30-90    : OWT added to originally unused pin 15 and set to 0.
" 4-30-90    : BCADD was changed to EVBEF.  BCADD is for broadcast
"             which the Sequencer doesn't support.  EVBEF is for WAIT
" 4-30-90    : Device type changed from 20L8 to 20V8
" 5-2-90     : Polarity of IRD changed to active low.
" 7-18-90    : OWT is implemented to assert WAIT at AS time.
" 7-18-90    : The device was changed from 20V8 to 22V10. Needed feedbacks.
" 7-23-90    : Change polarity of input EVBEF to !EVBEF.
END SEQ_PAL1
```

```
"Output Slave Status bit 0

 OSS0 = CON & BSY  & !IMS2 & !IMS1 & !IMS0 & DIDS1 & !DIDS2 & !CSR "BuSY & single xfer
    # CON & NTV & !IMS2 & IMS1 & !IMS0 & DIDS1 & !DIDS2 "NTV & 2nd Address
    # OSS0 & IDS & DIDS2 & CON                    "Latch while DS=DK=1
   # OSS0 & !IMS2 & IMS0 & !IDS & !DIDS2 & CON ; "Latch if DS=DK=0 & MS=1or3

"Output Slave Status bit 1
 OSS1 = CON & IMS2 & !CSR  "any MS=4-7 on DS up or down
    # CON & !IMS2 & NTV & DIDS1 & !DIDS2
    "any Not Valid address if not BuSY on DS up
    # CON & !IMS2 & IMS0 & NTV & !EOB & !DIDS1 & DIDS2
    "any Not Valid address, not BuSY, not End Of Block, & MS=1or3 on DS down
    # CON & !IMS2 & IMS0 & EOB & DIDS1 & !DIDS2
    "End Of Block if not BuSY on DS up
    # CON & !IMS2 & IMS0 & EOB & !DIDS1 & DIDS2
    "End Of Block if not BuSY on DS down
    # OSS1 & IDS & DIDS2 & CON                    "Latch while DS=DK=1
    # OSS1 & IMS0 & !IDS & !DIDS2 & CON; "Latch if DS=DK=0 & MS=1,3,5,or 7

"Slave Status bit 2
 OSS2 = CON & IMS2   "any MS=4-7 on DS up or down
    # CON & !IMS2 & NTV & !EOB & DIDS1 & !DIDS2
    "any Not Valid address if not BuSY and not End Of Block on DS up
    # CON & !IMS2 & IMS0 & NTV & !EOB & !DIDS1 & DIDS2
    "any Not Valid address, not BuSY, not End Of Block, & MS=1or3 on DS down
    # OSS2 & IDS & DIDS2 & CON                    "Latch while DS=DK=1
    # OSS2 & IMS0 & !IDS & !DIDS2 & CON  ; "Latch if DS=DK=0 & MS=1,3,5,or 7

"Output Data acKnowlege generates DK

 ODK = !IWT & CON & OAK & DIDS2  "set if DS (delayed) and attached and not WaiT
    # IMS1 & IMS0 & CON & OAK & DIDS2
    "set if DS (delayed) and attached and MS=3 (pipeline) even if WaiT
   # ODK & CON & OAK & DIDS2  "transition hold while DS (delayed)
    # IWT & !IMS1 & ODK   "hold if WaiT and not MS1
    # IWT & !IMS0 & ODK  ;          "hold if WaiT and not MS0
    "i.e. hold if not MS=3 (pipeline) AND WaiT, release it otherwise

"FastBus ReaD

 FB_RD = CON & !INH & IRD & !IMS2 & !IMS1 & !IMS0 & DIDS1 & !DIDS2
          "set on DS up, MS=0 random data read
    # CON & !INH & IRD & !IMS2 & IMS0 & DIDS1 & !DIDS2
          "set on DS up, MS=1 block read
          "set on DS up, MS=3 pipeline read
    # CON & !INH & IRD & !IMS2 & IMS0 & !DIDS1 & DIDS2
          "set on DS dn, MS=1 block read
          "set on DS dn, MS=3 pipeline read
    # CON & IRD & !IMS2 & !IMS1 & !IMS0 &IDS & DIDS2 & FB_RD
          "latch while MS=0 read and DS,DK up
          "latch while MS=1,3 read
    # CON & IDS & DIDS2 & FB_RD                   "latch while DS,DK up
    # CON & !IDS & !DIDS2 & FB_RD  ;     "latch while DS,DK down
                                  "i.e. latch until new MS or WR cycle

"Output INHibit data transfers

 OINH = INH
    # CON & !CSR ;          "This stops NTA Incrementing for the FIFO in Data Space
```

TRIGCNT.abl TRIGGER COUNTER

```
MODULE TRIGGER_COUNTER
FLAG '-R7'
TITLE 'ANDY ROMERO RD/EE CONTROLS 6-14-90'


TRIGCNT DEVICE 'P22V10';

PIN1,PIN3,PIN4,PIN5   PIN  1,3,4,5;

PIN23,PIN22,PIN21,PIN20,PIN19  PIN  23,22,21,20,19;
PIN18,PIN17,PIN16,PIN15,PIN14  PIN  18,17,16,15,14;

"Declaration of active levels"

CLK25MHZ    = PIN1;
TRINC       = !PIN3;
DEC         = !PIN4;
RESET       = !PIN5;

NOTEMP      = !PIN23;
SYNCINC     = PIN22;
HF          = !PIN21;
ERROR       = !PIN20;
TRCQ5       = PIN19;
TRCQ4       = PIN18;
TRCQ3       = PIN17;
TRCQ2       = PIN16;
TRCQ1       = PIN15;
TRCQ0       = PIN14;

"Standard Symbol Declarations"

X=.X.;
C=.C.;
P=.P.;

TRIG_COUNT = [TRCQ5,TRCQ4,TRCQ3,TRCQ2,TRCQ1,TRCQ0];
WAIT_FOR_TRINC = 1;
SYNCINC_ACTIVE = 0;
OVERFLOW = [1,1,1,1,1,1];

EQUATIONS

WHEN RESET THEN TRIG_COUNT := [0,0,0,0,0,0];
ELSE WHEN (TRIG_COUNT == OVERFLOW) THEN TRIG_COUNT := OVERFLOW;
ELSE WHEN ((SYNCINC == SYNCINC_ACTIVE) & DEC) THEN TRIG_COUNT := TRIG_COUNT;
ELSE WHEN ((SYNCINC == SYNCINC_ACTIVE) & !DEC) THEN TRIG_COUNT := (TRIG_COUNT + 1);
ELSE WHEN ((SYNCINC == WAIT_FOR_TRINC) & DEC) THEN TRIG_COUNT := (TRIG_COUNT - 1);
ELSE TRIG_COUNT := TRIG_COUNT;


HF := ((TRIG_COUNT >= 31) & !RESET);

ERROR := ((TRIG_COUNT == 63) & !RESET);

NOTEMP = (TRIG_COUNT != 0);
```

```
[C,  0,1,1]->[0,0,1,1,  19];
[C,  X,1,1]->[0,1,1,1,  20];
[C,  1,0,1]->[0,1,1,1,  19];
[C,  1,0,1]->[0,1,1,1,  18];
[C,  1,0,1]->[0,1,1,1,  17];
[C,  1,0,1]->[0,1,1,1,  16];

[C,  0,1,1]->[0,0,1,1,  16];
[C,  X,1,1]->[0,1,1,1,  17];
[C,  0,1,1]->[0,0,1,1,  17];
[C,  X,1,1]->[0,1,1,1,  18];
[C,  0,1,1]->[0,0,1,1,  18];
[C,  X,1,1]->[0,1,1,1,  19];
[C,  0,1,1]->[0,0,1,1,  19];
[C,  X,1,1]->[0,1,1,1,  20];

"[CLK      /TRNC /DEC /RST ]-> [/ NTMP  SINC  /HF  ERR  TR_CNT]

[C,  0,1,1]->[0,0,1,1,  20];
[C,  X,1,1]->[0,1,1,1,  21];
[C,  0,1,1]->[0,0,1,1,  21];
[C,  X,1,1]->[0,1,1,1,  22];
[C,  0,1,1]->[0,0,1,1,  22];
[C,  X,1,1]->[0,1,1,1,  23];
[C,  0,1,1]->[0,0,1,1,  23];
[C,  X,1,1]->[0,1,1,1,  24];
[C,  0,1,1]->[0,0,1,1,  24];
[C,  X,1,1]->[0,1,1,1,  25];
[C,  0,1,1]->[0,0,1,1,  25];
[C,  X,1,1]->[0,1,1,1,  26];
[C,  0,1,1]->[0,0,1,1,  26];
[C,  X,1,1]->[0,1,1,1,  27];
[C,  0,1,1]->[0,0,1,1,  27];
[C,  X,1,1]->[0,1,1,1,  28];
[C,  0,1,1]->[0,0,1,1,  28];

"[CLK      /TRNC /DEC /RST ]-> [/ NTMP  SINC  /HF  ERR  TR_CNT]

[C,  X,1,1]->[0,1,1,1,  29];
[C,  0,1,1]->[0,0,1,1,  29];
[C,  X,1,1]->[0,1,1,1,  30];
[C,  0,1,1]->[0,0,1,1,  30];
[C,  X,1,1]->[0,1,1,1,  31];
[C,  0,1,1]->[0,0,0,1,  31];
[C,  X,1,1]->[0,1,0,1,  32];
[C,  0,1,1]->[0,0,0,1,  32];
[C,  X,1,1]->[0,1,0,1,  33];
[C,  0,1,1]->[0,0,0,1,  33];
[C,  X,1,1]->[0,1,0,1,  34];
[C,  0,1,1]->[0,0,0,1,  34];
[C,  X,1,1]->[0,1,0,1,  35];
[C,  0,1,1]->[0,0,0,1,  35];
[C,  X,1,1]->[0,1,0,1,  36];
[C,  0,1,1]->[0,0,0,1,  36];
[C,  X,1,1]->[0,1,0,1,  37];
[C,  0,1,1]->[0,0,0,1,  37];
[C,  X,1,1]->[0,1,0,1,  38];
[C,  0,1,1]->[0,0,0,1,  38];
[C,  X,1,1]->[0,1,0,1,  39];
[C,  0,1,1]->[0,0,0,1,  39];
[C,  X,1,1]->[0,1,0,1,  40];
```

```
[C,  0,1,1]->[0,0,0,1,  49];
[C,  X,1,1]->[0,1,0,1,  50];
[C,  0,1,1]->[0,0,0,1,  50];
[C,  X,1,1]->[0,1,0,1,  51];
[C,  0,1,1]->[0,0,0,1,  51];
[C,  X,1,1]->[0,1,0,1,  52];
[C,  0,1,1]->[0,0,0,1,  52];
[C,  X,1,1]->[0,1,0,1,  53];
[C,  0,1,1]->[0,0,0,1,  53];
[C,  X,1,1]->[0,1,0,1,  54];
[C,  0,1,1]->[0,0,0,1,  54];
[C,  X,1,1]->[0,1,0,1,  55];
[C,  0,1,1]->[0,0,0,1,  55];
[C,  X,1,1]->[0,1,0,1,  56];
[C,  0,1,1]->[0,0,0,1,  56];
[C,  X,1,1]->[0,1,0,1,  57];
[C,  0,1,1]->[0,0,0,1,  57];
[C,  X,1,1]->[0,1,0,1,  58];
[C,  0,1,1]->[0,0,0,1,  58];
[C,  X,1,1]->[0,1,0,1,  59];
[C,  0,1,1]->[0,0,0,1,  59];
[C,  X,1,1]->[0,1,0,1,  60];
[C,  0,1,1]->[0,0,0,1,  60];
[C,  X,1,1]->[0,1,0,1,  61];
[C,  0,1,1]->[0,0,0,1,  61];
[C,  X,1,1]->[0,1,0,1,  62];
[C,  0,1,1]->[0,0,0,1,  62];
[C,  X,1,1]->[0,1,0,1,  63];
[C,  0,1,1]->[0,0,0,0,  63];
[C,  0,X,1]->[0,1,0,0,  63];
[C,  1,X,1]->[0,1,0,0,  63];
[C,  X,X,0]->[1,1,1,1,  0];

"[CLK       /TRNC /DEC /RST ]-> [/ NTMP  SINC  /HF   ERR   TR_CNT]

END TRIGGER_COUNTER
```

APPENDIX C

PARTS LIST

BROOKTREE BT501KC	$15.00	4	$60.00
CYPRESS CY7C190-15PC	$9.25	2	$18.50
CON/WIN S15R8 25 MHZ OSC	$7.00	1	$7.00
1N4005	$0.04	1	$0.04
1N914	$0.03	1	$0.03
2N5770	$0.17	2	$0.34
51 OHM 1/8W RES	$0.18	2	$0.36
100 OHM 1/8W RES	$0.16	18	$2.88
130 OHM RES	$0.06	1	$0.06
220 OHM RES	$0.07	12	$0.84
330 OHM RES	$0.07	7	$0.49
1K RES	$0.06	11	$0.66
2K RES	$0.06	1	$0.06
2.2K RES	$0.06	29	$1.74
4.7K RES	$0.07	6	$0.42
10K RES	$0.06	3	$0.18
51K RES	$0.07	1	$0.07
DALE CSC08A-03-101G sipp	$0.30	51	$15.30
DALE CSC10A-01-101G sipp	$0.35	10	$3.50
DALE CSC06A-01-471G sipp	$0.22	2	$0.44
DALE CSC10A-01-102G sipp	$0.22	4	$0.88
33 uf CAP	$0.45	1	$0.45
47 uf CAP	$0.98	5	$4.90
68 pf CAP	$0.17	3	$0.51
SPRAGUE 923CZ5U104M050B 0.1uF CAP	$0.11	274	$30.69
GRAYHILL 76-RSB06S DIP SWITCH	$3.50	1	$3.50
3M 34PIN HEADER 3431-1303	$1.60	1	$1.60
5A PICOFUSE	$0.48	7	$3.36
MIL MAX 0665-0-15-15-30-27-10-2	$0.22	14	$3.08
Rgt Angle Coax LEMO conn. EPL. 00.250.NTN	$4.46	4	$17.84
LEDTRONICS RED PC120TR4 HI-EFFIC.	$0.45	10	$4.50
CK SWITCH EP12D1ABE	$5.36	1	$5.36
GEN. SEMI. ICTE-5 TRANSORB	$0.90	3	$2.70
AUXILIARY CONN			$0.00
FASTBUS CONN.			$0.00
FRONT PANEL			$0.00
HARDWARE			$0.00
ROB. NUGNT. ICT-243-S-TG 24 PIN IC SKT.	$0.88	26	$22.88
ROB. NUGNT. ICT-203-S-TG 20 PIN IC SKT.	$0.88	6	$5.28
ROB. NUGNT. ICA-143-SCO-TG30 OSC. SKT	$0.45	1	$0.45
BURNDY QILE 28P-410T 28 PIN PLCC SKT	$1.84	4	$7.36

Fermilab

March 6, 1991

TO: Distribution

FROM: Carl Swoboda

SUBJECT: FSCC Documentation

Enclosed you will find pages: 75 & 76; 77 & 78; 79 & 80; and 97 & 98 to the FSCC documentation in your SSD binder. Please discard the old pages and replace with the new pages.

Distribution:
Dave Christian, MS 222
Brad Cox, MS 219
Peter Garbincius, MS 208
Hector Gonzalez, MS 222
Franco Grancagnolo, MS 219
Bill Haynes, MS 127
Rick Kwarciany, MS 222
Wolfgang Kowald, MS 219 (2 sets)
Mark Larwill, MS 220
Cash McManus, MS 219
Garry Moore, MS 222
Tom Nash, MS 127
John Peoples, MS 105
Ruth Pordes, MS 120
Matteo Racagni, MS 219
Carl Swoboda., MS 222
Robert Trendler, MS 220
Ken Treptow, MS 222

Fermilab

February 5, 1991

TO: Distribution

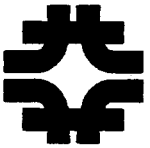
FROM: Carl Swoboda

SUBJECT: FSCC Documentation

The enclosed document is an "as built" upgrade to the FSCC documentation. Additions include expanded PAL Equations, PROM Designations, PROM Placement and Module Front Panel information. Please replace the present FSCC document in your binder with the enclosed document.

Distribution:
Dave Christian, MS 222
Brad Cox, MS 219
Peter Garbincius, MS 208
Hector Gonzalez, MS 222
Franco Grancagnolo, MS 219
Bill Haynes, MS 127
Rick Kwarciany, MS 222
Wolfgang Kowald, MS 219 (2 sets)
Mark Larwill, MS 220
Cash McManus, MS 219
Garry Moore, MS 222
Tom Nash, MS 127
John Peoples, MS 105
Ruth Pordes, MS 120
Matteo Racagni, MS 219
Carl Swoboda., MS 222
Robert Trendler, MS 220
Ken Treptow, MS 222

DAE/0023



Fermilab

FASTBUS Smart Crate Controller

Mark Bernett, Mark Bowden, Gustavo Cancelo, Rick Kwarciany, John Urish

Version PC 3
Manual

HN 96

October 22, 1990

Table of Contents

1 General Information

1.1 Purpose

The FSCC is designed as a simple readout controller for low occupancy front-end modules. It performs most basic FASTBUS operations but is not intended to be a "general-purpose" FASTBUS master.

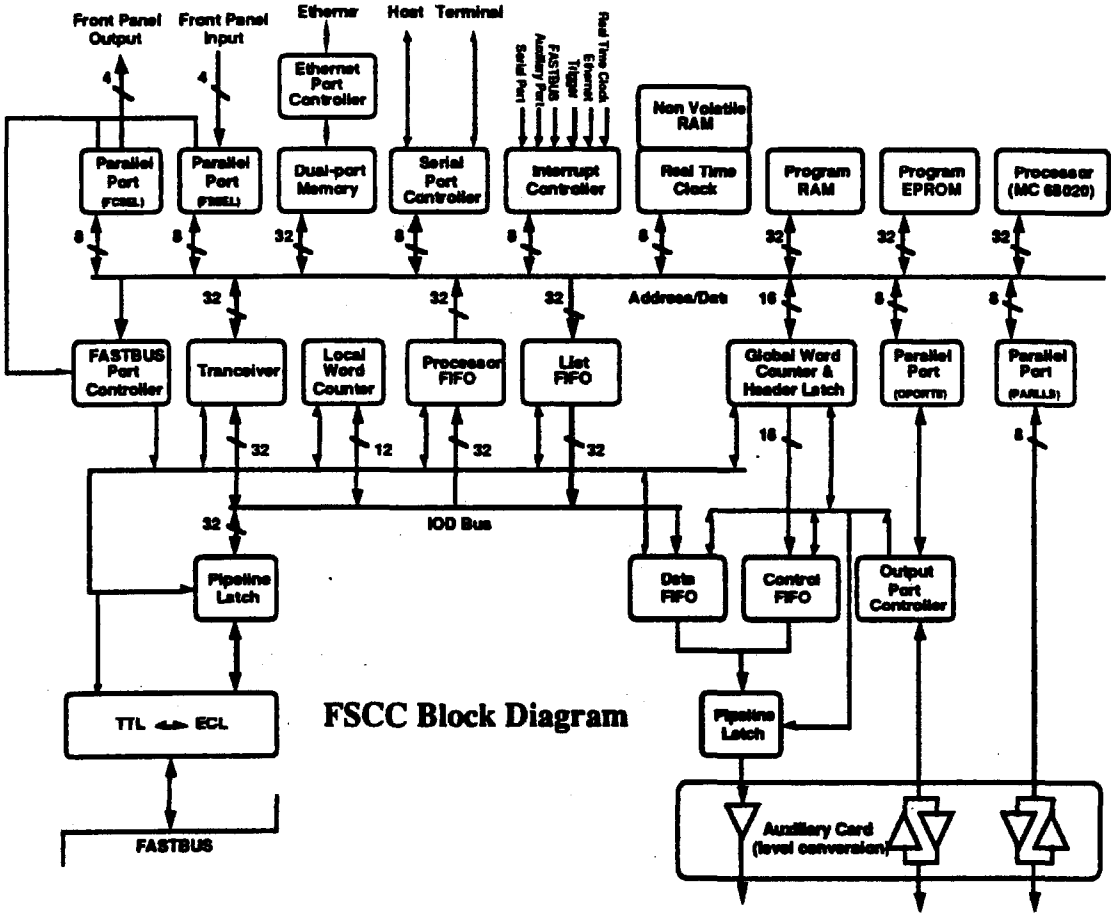
A Motorola 68020 processor is used to control operation of the module and any features which are not time-critical have been allocated to software.

The design goals are as follows;

a) a typical readout time of 1 µsec for a single slave module with a few words of data, including primary address and address release,

b) ability to execute most standard FASTBUS Master operations, and

c) design simplicity such that a working prototype module can be assembled in 6-9 months.



FSCC Block Diagram

Figure 1

1.2 Standard Bus Connections (FASTBUS)

The FSCC operates as a master on both the FASTBUS crate segment port and the (non-FASTBUS) auxiliary port. It also supports FASTBUS slave operations on the crate segment, but at a very reduced rate.

1.3 Packaging

The FSCC is a single-width FASTBUS module containing approximately 100 integrated circuits. Physical dimensions of the module are per the FASTBUS specification.

1.3.1 Module Pinout (Backplane Connections)

a) FASTBUS Crate Segment (130 pin FASTBUS standard connector) Refer to the FASTBUS specification.

b) Auxiliary Port (195 pin FASTBUS standard 3 row connector), see Figure 2

1.3.2 Front Panel

FASTBUS Master	Green LED
FASTBUS Slave	Yellow LED
Reset Pushbutton	Hard processor reset.
Remote Reset	Hard processor reset. This is an active low TTL input. Shorting the connetor or applying a TTL low will cause a reset. This input may be Daisy-Chained.
Trigger Input	The Trigger Input consists of five differential RS 485 pairs. A 4 bit Trigger ID is latched on the leading edge of the Trigger Input Strobe. This value can be read by the processor through a parallel port. The Trigger Input Strobe generates a processor interrupt if enabled. The processor should clear this interrupt to allow further trigger recognition. State of the Trigger Strobe signal is also available to the FASTBUS Port Controller for fast triggering. The Trigger ID should be valid for at least 60 nsec prior to the leading edge of the Trigger Input Strobe and remain valid for at least 20 nsec after the leading edge of the strobe. The Trigger Strobe pulse width should be at least 150 ns to ensure recognition by the FASTBUS Port Controller.

Trigger Input (20 pin IDC)

Pin#	
1-10	Reserved
11	Trigger Strobe +
12	Trigger Strobe -
13	Trigger ID0 +
14	Trigger ID0 -
15	Trigger ID1 +
16	Trigger ID1 -
17	Trigger ID2 +
18	Trigger ID2 -
19	Trigger ID3 +
20	Trigger ID3 -

Front Panel Output: The Front Panel Output (20 pin IDC) consists of four latched differential RS485 pairs, which are driven by the processor via a parallel port. These are user defined.

Pin#	
1	Front Panel 0 +
2	Front Panel 0 -
3	Front Panel 1 +
4	Front Panel 1 -
5	Front Panel 2 +
6	Front Panel 2 -
7	Front Panel 3 +
8	Front Panel 3 -
9-20	Reserved

2 Serial Ports: RS232 signal levels (4 pin LEMO connectors) one for Host connection and one for Terminal connection.

Terminal port	Pin #	
	1	TXD (transmit data)
	2	(no connection)
	3	RXD (receive data)
	4	Ground
Host port (Null Modem)	Pin #	
	1	RXD (receive data)
	2	CTS (+10V Reference)
	3	TXD (transmit data)RXD (receive data)
	4	Ground

Ethernet Port: Cheapernet signal levels, Isolated BNC Connector.

Permit_In/Permit_Out: Serial "daisy-chain" signals for FSCC auxiliary port bussing, LEMO connectors.

Auxilliary Port Connector

Pin	Signal	Function	
B65	VTT	-2.0 Volts	
B64	GND	Power Return	
B63	AC15	NC	
B62	AC14	NC	
B61	AC13	Auxiliary Control Bit	13
B60	AC12	Auxiliary Control Bit	12
B59	AC11	Auxiliary Control Bit	11
B58	AC10	Auxiliary Control Bit	10
B47	D31	Output Data Bus Bit	31
B46	D30	Output Data Bus Bit	30
B45	D29	Output Data Bus Bit	29
B44	D28	Output Data Bus Bit	28
B43	D27	Output Data Bus Bit	27
B42	D26	Output Data Bus Bit	26
B41	D25	Output Data Bus Bit	25
B40	D24	Output Data Bus Bit	24
B39	D23	Output Data Bus Bit	23
B38	D22	Output Data Bus Bit	22
B37	D21	Output Data Bus Bit	21
B36	D20	Output Data Bus Bit	20
B35	D19	Output Data Bus Bit	19
B34	D18	Output Data Bus Bit	18
B33	D17	Output Data Bus Bit	17
B32	D16	Output Data Bus Bit	16
B31	D15	Output Data Bus Bit	15
B30	D14	Output Data Bus Bit	14
B29	D13	Output Data Bus Bit	13
B28	D12	Output Data Bus Bit	12
B27	D11	Output Data Bus Bit	11
B26	D10	Output Data Bus Bit	10
B25	D09	Output Data Bus Bit	9
B24	D08	Output Data Bus Bit	8
B23	D07	Output Data Bus Bit	7
B22	D06	Output Data Bus Bit	6
B21	D05	Output Data Bus Bit	5
B20	D04	Output Data Bus Bit	4
B19	D03	Output Data Bus Bit	3
B18	D02	Output Data Bus Bit	2
B17	D01	Output Data Bus Bit	1
B16	D00	Output Data Bus Bit	0
B15	AC09	Auxiliary Control Bit	9
B14	AC08	Auxiliary Control Bit	8
B13	AC07	Auxiliary Control Bit	7
B12	AC06	Auxiliary Control Bit	6
B11	AC05	Auxiliary Control Bit	5
B8	AC04	Auxiliary Control Bit	4
B7	AC03	Auxiliary Control Bit	3
B6	AC02	Auxiliary Control Bit	2
B5	AC01	Auxiliary Control Bit	1
B4	AC00	Auxiliary Control Bit	0
A12,A32,C63	VEE	-5.2 Volts	
A43,C53,C12	VCC	+5.2 Volts	
A20,A22,A53,A63,C22,C32,C43,C44		Power Return	

Figure 2

1.4 Power Requirements

This module dissipates approximately 40 watts.

+5.0 volts	6 amps
-5.2 volts	2 amp
-2.0 volts	0.2 amps

2 Theory of Operation and Operating Modes

Figure 1 shows a general block diagram of the FSCC. The individual blocks are explained in detail below.

2.1 Basic Operation

All module operations are controlled by memory-mapped instructions from the **68020 Processor**. Processor operation is described in section 2.2.

Low level management of the various ports is handled by dedicated controllers. In the case of the **Serial** and **Ethernet Ports**, the controllers are commercial integrated circuits. Controllers for the **FASTBUS Port** (FPORT) and Auxiliary **Output Port** (OPORT) are implemented using PLD state machines.

To maximize throughput, high speed block transfer data from FASTBUS to the Auxiliary Output Port is routed through the **Data FIFO** which serves two functions; it 1) decouples the input and output data rates, and 2) provides buffering of one full crate-event of front-end data for insertion of leading word counts on output. The buffer is implemented using commonly available integrated circuit FIFO's which provide a depth of 2K, 32-bit words. This is expandable by substitution of pin-compatible higher density parts.

A 256X32-bit **Processor FIFO** connects the FASTBUS port to the Processor. Input to this FIFO can be enabled or disabled by the Processor. When enabled, any data which is written to the Data FIFO will simultaneously be written to the Processor FIFO. The Processor FIFO provides a high-speed path through which the processor can sample data in the event stream.

A 256X32-bit **List FIFO** also connects the Processor to the FASTBUS data port. This FIFO is written by the Processor and contains a data list for use by application-specific FPORT Controller microcode.

To further improve the readout speed of the FSCC, a **Local Word Counter** is implemented. In a typical application, this counter is loaded directly from the first word of a Block or Pipelined transfer, with a fixed word count position (bits 0-11). The counter can also be preset by the processor for use with slave modules which do not supply a leading word count. It is implemented in PLD's and can be adapted for inclusive or non-inclusive word counts.

For slave modules capable of producing leading word counts, the local counter eliminates the extra delay the FPORT Controller normally incurs in checking for the SS=2 "end-of-block" condition. It allows better pipelining of the data since the controller will not be required to read beyond end-of-block and then back-up it's internal counters and pipeline registers. The Local Word Counter can also be used to produce intermediate word counts, at user-selected boundaries for insertion in the main data stream. Maximum word count is 4095.

A **Global Word Counter and Header Latch** is also implemented. This is normally used to provide a total count of all data read from a group of modules. It is clocked by the FPORT Controller. The value of this counter is inserted into the output data stream along with 5 bits of header information on command of the OPORT Controller. The FPORT Controller provides both an immediate and delayed "end-of-event" signal for this purpose. The word count is limited to 12 bits. The 5 bit header field is loaded directly by the processor. Register definitions and operating modes for the counter/header latch and OPORT controller are outlined in section 2.3.3.

A small **Control FIFO** connects the word count / header output and Data FIFO outputs to allow overlapping of events in the Data FIFO. This is necessary for variations in event size and input-output transfer rates. The immediate end-of-event signal is normally used to transfer the current word count/header to the Control FIFO. The delayed end-of-event signal, which tracks the data through the Data FIFO, is used to insert the word count/header into the output data stream.

2.2 On-Board Processor

The processor is a Motorola 68020 running at 20 MHz. The 68020 was selected strictly on the basis of software compatibility with existing Fermilab modules.

The processor address map is decoded as follows;

Base Address	Transfer Type	Name	Purpose
0000 0000	byte, word, long word	ROM1S*	Program EPROM (Bank 1, 256K)
0004 0000	byte, word, long word	ROM2S*	Program EPROM (Bank 2, 256K)
0020 0000	byte, word, long word	RAM1S*	Program RAM (Bank 1, 128K)
0022 0000	byte, word, long word	RAM2S*	Program RAM (Bank 2, 128K)
0030 0000	byte	NVDS*	Real-time clock / Non-volatile memory
0040 0000	byte, word, long word	ETHS*	Ethernet Dual-Port RAM
0044 0000	long word	CAS	Ethernet Channel Attention
0046 0000	byte	UARTS*	Serial Ports (68681)
0048 0000	byte	TMRS*	Interrupt Vector/Timer Logic (MC68901)
004A 0000	byte	OPORTS*	OPORT Controller (EPB1400)
004C 0000	byte	PARLLS*	Auxiliary Parallel Port
004E 0000	word	H&CSEL*	Header/Counter Registers
0050 0000	byte, word, lon ord	LPBKS	Ethernet loopback mode set
0052 0000	byte, word, lon ord	LPBKC	Ethernet loopback mode clear
0054 0000	byte, word, lon ord	ETHRES	Ethernet reset set
0056 0000	byte, word, lon ord	ETHRESC	Ethernet reset clear
0060 0000	long word	FB1S*	FPORT Controller (Fast cycle instructions)
0062 0000	long word	FB2S*	FPORT Controller (Slow cycle instructions)
0064 0000	byte	FCSEL*	FASTBUS Parallel Port 1 (MC68230)
0066 0000	byte	FSSEL*	FASTBUS Parallel Port 2 (MC68230)
0068 0000	long word	FIFO1S*	Processor FIFO
006A 0000	long word	FIFO2S*	List FIFO

* Active low signal.

The processor memory consists of 256 KBytes of 0-wait state static RAM and 512 KBytes of 1-wait state EPROM. Byte, word and long-word accesses are supported. Minimal configuration is 128 KBytes of RAM and 256 KBytes of EPROM.

A DS1286 real-time clock with interrupt capability and 50 bytes of RAM is provided for system use. Both functions are maintained by a built-in battery. The RAM holds module-specific information (e.g., Ethernet address).

Processor reset occurs at power-up, by pressing the front-panel Reset Pushbutton or through the front panel reset input Lemo. A reset can also be generated by a "watchdog" timer contained in the FASTBUS Parallel Port 1 controller (68230). If enabled, this timer must be reset by software periodically. The timer can be set for any period from 4 µsec up to approximately 50 seconds. A module reset can be forced by software which drives the parallel port watchdog timeout bit low.

A Processor bus response timer is contained in the other 68230 port controller. When enabled, this timer will generate a BUS ERROR exception if the Processor fails to complete a bus cycle within the specified time. The bus response timer also serves as the FASTBUS "long timer". In this application however, a FASTBUS timeout will only be detected on a subsequent Processor to FPORT Controller access. When the FPORT Controller is operating in a standalone mode for extended periods (as in a microcode readout loop), the Processor should occasionally access the FPORT Controller using the NULL instruction to determine if a timeout has occurred.

2.2.1 Control and Status Registers

The FSCC has no control or status registers which are independent of the associated controllers. Refer to the appropriate controller description for register definitions.

2.2.2 Error Responses

The FPORT Controller normally monitors the FASTBUS exception logic. When a FASTBUS error or FASTBUS Reset is detected, the controller will terminate its current operation and return to an idle state. If another FASTBUS instruction is pending (Processor pipelined mode), the FPORT Controller will return Processor DSACK immediately to clear the bus and allow interrupt processing. The instruction which was pending will not be executed. If the FPORT Controller fails to recognize an error and return control to the Processor, the Processor will eventually timeout with a BUS ERROR exception. This can happen, for example, if the address/data cycle timer is disabled and no acknowledge is received.

FASTBUS exceptions can be cleared by asserting the parallel port FASTBUS clear error (**FCLERR**) or sequencer clear error signals. The FASTBUS interrupt vector must be programmed and the interrupt enabled to allow recognition of FASTBUS exceptions.

Because FASTBUS operations can be queued, a FASTBUS error interrupt may not apply to the current processor data cycle. For example, a FASTBUS block transfer can be initiated and then followed by any number of non-FASTBUS processor operations while the block transfer takes place. A FASTBUS error during the block transfer generates an interrupt which may be unrelated to the current processor activity. For any FASTBUS interrupt, the parallel port status lines must be examined to determine the cause of the interrupt.

2.2.3 Software

The pSOS operating system and pROBE resident monitor, module diagnostics and control software for the Ethernet and Serial Ports is implemented in EPROM. User code may be placed in the second EPROM bank or in RAM.

2.2.4 Interrupts

A 68901 multifunction peripheral provides the vectored interrupt control. Refer to the Motorola 68901 manual for register definitions. Interrupts for the following inputs are provided;

Interrupt Line	Name	Function
7	TSTRB	Trigger Input Strobe
6	RTCREQ*	Real-time Clock
5	ETHREQ	Ethernet
4	SERREQ*	Serial Port
3	FBERR*	FASTBUS error
2	AUXREQ*	OPORT Controller
1	FBREQ*	FASTBUS request
0	FPCREQ*	FASTBUS Port Controller

* Active low signal.

Interrupt vectors are programmable and interrupts can be separately enabled, disabled or masked in the 68901 controller. The 68020 interrupt mask is hardwired to IPL2, which corresponds to a interrupt level of 4, and is not used except to enable or disable all interrupts. A typical FSCC readout program will have an inner loop of less than 100 µsec, so for many applications, polling of the serial ports, slave mode and trigger inputs may be more efficient than use of interrupts.

TSTRB-IRQ7: The trigger input interrupt is generated by the rising edge of the front panel trigger strobe. Four trigger ID bits are latched in parallel port 1.

RTCREQ*-IRQ6: The real-time clock (DS1286-U38) can be programmed to generate periodic interrupts at a rate of 10 msec to 100 seconds. It can also be programmed to interrupt on a specific date or/and time.

ETHREQ-IRQ5: This is the Ethernet message interrupt. Ethernet messages are buffered by the controller in Dual-Port memory so immediate interrupt response is not required.

SERREQ*-IRQ4: The serial ports can be programmed to generate interrupts when the receive buffer is loaded or the transmit buffer is empty.

FBERR*-IRQ3: FASTBUS error conditions result in a processor interrupt when enabled. The conditions are:

1)FASTBUS timeout-failure of a slave module to respond within 1500 ns on an address or data cycle. This error can be disabled through the short timer enable bit (STEN) in parallel port 2.

2)FASTBUS SS errors-SS responses of 1,2,3,4,5,6 or 7 on a Address cycle or 1,3,4,5,6, or 7 on a Data cycle will cause a FASTBUS error interrupt. The last non-zero value of SS is latched at parallel port 2 (FLSS0, FLSS1, FLSS2).

3)Data FIFO overflow.

AUXREQ*-IRQ2: OPORT request to processor, see section 2.3.3

FBREQ*-IRQ1: (External) FASTBUS request-This interrupt is generated when the FSCC is accessed as a slave or when a FASTBUS Service Request (SR) or FASTBUS Reset (RB) is issued. RB does not directly reset the processor. A processor reset can be generated by software in the RB interrupt handler. RB does not cause an interrupt if it is being driven by the FSCC itself.

FPCREQ*-IRQ0: The FPORT controller can assert an interrupt request at any time. In the standard FASTBUS instruction set this interrupt is generated at the End of Block in Block and Pipelined transfers.

2.3 Communication Interfaces

The Ethernet interface physical connection will be through a front-panel "cheapernet" BNC connector. Software for this interface must support connection to VAX systems. The 82586 coprocessor interfaces to the 68020 through a 2K by 32 bit dual-port memory. This configuration allows both processors to operate independently and is necessary to avoid buffer overrun if the 68020 is unable to process interrupts for extended periods. It also improves the speed of FASTBUS operations since there is no contention on the processor bus.

Two standard RS-232 serial ports (Signetics 68681) are provided for development and diagnostic use. One of these will be connected to a terminal or PC. The other will typically serve as a link to a host machine. Data rates to 9600 baud will be supported. Operation of these ports is controlled by on-board software. At module initialization, the ports are configured for 9600 baud using XON / XOFF protocol. Refer to the 68681 data sheet for register definitions. The Host port is a null-modem connection.

FASTBUS is accessed through the crate segment backplane. All FASTBUS operations are performed by a microsequencer as directed by the processor. A sequence of FASTBUS transfers may be performed to the Data FIFOs or to the Processor FIFOs without processor intervention.

The Output Port is controlled by a programmable microsequencer which is directed by the processor. A transfer may be initiated and proceed unattended. The Output Port transfers data through the Auxillary connector of the backplane. Personality cards, which plug into the auxillary connector from the rear of the FASTBUS crate are available for standard protocols.

2.3.1 Addressing Modes (FASTBUS)

Two 68230 parallel ports provide status communication between the processor and the FASTBUS control logic. Parallel port lines are defined in Figure 3 ("I" in the first column indicates an input signal, "O" indicates an output, "S" indicates a special function pin)

Port data direction and signal states are individually programmable. The software should avoid defining or driving INPUT pins as OUTPUTs, since multiple drives on the same signal line can cause circuit damage. If in doubt, define all unused signals as INPUTs or leave ports in the normal power-on reset configuration.

The FASTBUS Port Controller provides most of the low-level control of FASTBUS operations, based on simple memory-mapped instructions from the processor. It consists of three parallel EPS448 programmable sequencers plus assorted PLD's. The EPS448 is limited to 256 states, of which only 64 support conditional branching. Therefore, the sequencer will directly execute only FASTBUS primitives plus a few generally used compound operations.

Parallel Port 1

Input/Output	Bit	Signal	Function
I	A0	FRRD	FASTBUS received RD
I	A1	FRDS	FASTBUS received DS
I	A2	FRMS0	FASTBUS received MS0
I	A3	FRMS1	FASTBUS received MS1
I	A4	FRMS2	FASTBUS received MS2
I	A5	FRAK	FASTBUS received AK
I	A6	FRDY	FASTBUS master
I	A7	FSLV*	FASTBUS slave mode
I	B0	TRIG0	Trigger Vector 0 All Port B inputs are latched
I	B1	TRIG1	Trigger Vector 1 on the rising edge of the
I	B2	TRIG2	Trigger Vector 2 trigger input strobe.
I	B3	TRIG3	Trigger Vector 3
O	B4	FP0	Front Panel 0 Front Panel differential ECL
O	B5	FP1	Front Panel 1 Outputs
O	B6	FP2	Front Panel 2
O	B7	FP3	Front Panel 3
O	C0	PRS*	Processor FIFO reset
I	C1	PEF*	Processor FIFO empty
O	C2	COPYEN	Processor FIFO copy enable
O,S	C3	WDTO*	"Watchdog" timeout
O	C4	DRS*	Data FIFO reset
O	C5	DRT*	Data FIFO retransmit
O	C6	SRS*	Sequencer List FIFO reset
I	C7	DFF*	Data FIFO overflow

Parallel Port 2

Input/Output	Bit	Signal	Function
O	A0	FDGK	FASTBUS drive GK
O	A1	FDRB	FASTBUS drive RB
O	A2	FDSS0	FASTBUS drive SS0
O	A3	FDSS1	FASTBUS drive SS1
O	A4	FDSS2	FASTBUS drive SS2
	A5		
	A6		
O	A7	LCEN	Local Counter Enable
I	B0	SSTAT0	Sequencer status 0
I	B1	SSTAT1	Sequencer status 1
I	B2	SSTAT2	Sequencer status 2
I	B3	SSTAT3	Sequencer status 3
I	B4	FRSR	FASTBUS SR
	B5		
O	B6	FCLERR	FASTBUS Clear Errors
O	B7	SNRESET*	FPORT Controller Reset
O	C0	STEN	Short timer enable
I	C1	STO*	Short timeout
I,S	C2	LTEN	Long timer enable
O,S	C3	LTO*	Long timeout
I	C4	FLSS0	FASTBUS latched SS0
I	C5	FLSS1	FASTBUS latched SS1
I	C6	FLSS2	FASTBUS latched SS2
I	C7	FRESET*	FASTBUS reset

Figure 3

The EPS448 is EPROM programmable and the instruction set cannot be modified by the processor. FPORT Controller output signals are as follows:

	Bit	Name	Function
EPS448-3	F00	SMUX0	Select sequencer condition code set 0
	F01	SMUX1	Select sequencer condition code set 1
	F02	SMUX2	Select sequencer condition code set 2
	F03	FDSACK*	Processor data strobe acknowledge
	F04	SSTAT0	Sequencer status bit 0
	F05	SSTAT1	Sequencer status bit 1
	F06	SSTAT2	Sequencer status bit 2
	F07	SSTAT3	Sequencer status bit 3
	F08	CCLERR	Controller Clear Error
	F09	TIMER	Short timer enable
	F10	CEOE	Control "End-of-Event"
	F11	DEOE	Data "End-of-Event"
	F12	FCLK	Global word counter clock
	F13	STATCLK	Sequencer status clock
	F14	SR*	Sequencer List FIFO read
	F15	PFIFOEN	Processor FIFO input enable
EPS448-2	F00	FSAS	FASTBUS set AS
	F01	FCAS	FASTBUS clear AS
	F02	FSDS	FASTBUS set DS
	F03	FCDS	FASTBUS clear DS
	F04	FSDK	FASTBUS set DK
	F05	FCDK	FASTBUS clear DK
	F06	FDWT	FASTBUS drive WT
	F07	FDMS0	FASTBUS drive MS0
	F08	FDMS1	FASTBUS drive MS1
	F09	FDMS2	FASTBUS drive MS2
	F10	FDRD	FASTBUS drive RD
	F11	FDEG	FASTBUS drive EG
	F12		
	F13	FEOBA	FASTBUS EOB Acknowledge
	F14	FREQ	FASTBUS request bus
	F15	FREL	FASTBUS release bus
EPS448-1	F00	DFIFOEN	Data FIFO input enable
	F01	FPCREQ*	Sequencer Interrupt Request (TRAP)
	F02	FCOE*	FASTBUS control output enable
	F03	FDOE*	FASTBUS data output enable
	F04	SDG*	Data pipeline latch enable
	F05	DSBA	Data pipeline latch B->A mode control
	F06	DCPBA	Data pipeline latch B->A clock
	F07	DSAB	Data pipeline latch A->B mode control
	F08	SDCPAB	Data pipeline latch A->B clock
	F09	DDIR	Data pipeline latch direction
	F10	SPOE*	Processor tranceiver output enable
	F11	SDW	Data FIFO write
	F12	SRT*	Sequencer List FIFO retransmit
	F13	SLCOE	Local word counter output enable
	F14	LC0	Local word counter control 0
	F15	LC1	Local word counter control 1

The following FASTBUS master operations are permitted by the sequencer instruction set:

EQU Address(HEX)	FASTBUS Operation
0062 0300	BUS_ARBITRATE
0060 0004	BUS_RELEASE
0060 0304	ADDRESS_DATA_GEOGRAPHICAL
0060 0308	ADDRESS_CSR_GEOGRAPHICAL
0060 030C	ADDRESS_DATA_LOGICAL
0060 0310	ADDRESS_CSR_LOGICAL
0060 0314	ADDRESS_DATA_BROADCAST
0060 0318	ADDRESS_CSR_BROADCAST
0060 031C	ADDRESS_RELEASE
0062 0320	DATA_PROCESSOR_RANDOM_READ
0060 0324	DATA_PROCESSOR_RANDOM_WRITE
0062 0328	DATA_PROCESSOR_SEC_ADDRESS_READ
0060 032C	DATA_PROCESSOR_SEC_ADDRESS_WRITE
0062 0008	DATA_PROCESSOR_BLOCK_TRANSFER_READ
0060 000C	DATA_PROCESSOR_BLOCK_TRANSFER_WRITE
0060 0330	DATA_PROCESSOR_BLOCK_TRANSFER_TERMINATE
0060 0334	DATA_FIFO_BLOCK_TRANSFER_READ
0060 0338	DATA_FIFO_PIPELINED_READ_100
0060 033C	DATA_FIFO_PIPELINED_READ_200
0060 0340	DATA_FIFO_PIPELINED_READ_400
0060 0020	NULL
TBD	LIST
TBD	LIST_REPEAT
0062 0010	LOCAL_COUNTER_LOAD
0062 0014	LOCAL_COUNTER_READ
0062 0018	FIFO_WRITE_DATA
0062 0024	EOE

Sequencer instructions related to slave mode are;

0062 001C	SLAVE_DATA_INPUT
0060 0344	SLAVE_DATA_OUTPUT

Refer to Appendix A for detailed instruction definitions.

The FSCC supports standard and assured access arbitration. It does not support prioritized arbitration. Bits 0-5 of the processor data bus contain the arbitration vector. Bit 7 determines whether the assured access protocol is active. Bit 6 and 8-31 are ignored. These bit assignments correspond to CSR 8. A FASTBUS arbitration can be performed by a processor instruction of the form -

```
MOVE.L    CSR8 , BUS_ARBITRATE
```

If there is more than one master on the segment, the arbitration request should be followed by a check of the parallel port **FRDY**, **FRAK** and **FSLV*** bits to confirm that the arbitration was successful (or that the module was addressed as a slave while waiting for the bus). **FRDY** indicates that the FSCC has won the arbitration cycle and is the pending bus master. Before proceeding with an address cycle, **FRAK** should be checked to ensure that the previous master has released the bus. If **FSLV*** is low following an arbitration request, it indicates that the FSCC has been addressed as a slave by the current bus master. This should simultaneously cause a FASTBUS slave interrupt. The bus can also be acquired simply by asserting the GK line through the parallel port. Note that this is not a standard FASTBUS operation and is provided only for single master systems without ancilliary logic.

The Processor can issue a FASTBUS RB (Reset Bus) signal directly through the parallel port. In this case the parallel port GK signal should be asserted simultaneously. RB does not cause a processor interrupt when it is driven by the FSCC itself.

The FSCC supports a limited slave mode through processor emulation. Geographical address recognition logic is contained in hardware. All other slave functions are controlled by the processor. FASTBUS WT is asserted on each DS transition. The data cycle response time is dependent on software and is typically 10-15 µsec per FASTBUS word.

Accessing the FSCC as a slave causes a Processor interrupt.The processor must poll the DS, MS, RD and AK lines via parallel port inputs. The SS response is then placed on the bus via parallel port outputs and a slave mode input or output operation is executed by the sequencer.

These instructions cause a single word of data to be transferred between the processor and FASTBUS with an associated DK transition. WT is released prior to the DK transition and is reasserted on the next DS transition. Any FASTBUS CSR or DATA location can be defined by processor software.

The FSCC is limited in its ability to execute master and slave operations simultaneously, since the processor is involved in both cases. There is normally no need for the FSCC to address itself, with the possible exception of crate mapping. If the FSCC does address itself, the slave logic will attach (return AK) but no data cycles will be possible. A slave mode interrupt is generated and the processor software must then recognize the simultaneous master/slave condition by examining the parallel port **FSLV*** bit. The software should bypass any FASTBUS data cycles and retrieve the information directly from internal memory.

Application-specific FASTBUS operations may be included through use of the LIST instruction. These routines can use, but are not required to use, the LIST FIFO as a source of data bus operands. The Sequencer List FIFO provides a means of executing complex FASTBUS instruction streams without Processor intervention. The List FIFO contains only the data bus operands, instructions must be preprogrammed in the sequencer. For example, a complete crate of similar slave modules could be read out by a sequencer loop using a List of primary addresses. Small burst data transfers (e.g., FASTBUS messages) can also be supplied by the List FIFO. This FIFO eliminates the overhead associated with Processor-FPORT Controller communication. The LIST routine cannot contain FASTBUS operations which require Processor interaction (such as read cycles) unless the data is simply placed in the Processor FIFO for future access.

The LIST instruction causes the operation to be executed once. The LIST_REPEAT instruction causes the operation to repeat indefinitely. During LIST_REPEAT, the sequencer checks the Processor-FPORT Controller select line at the completion of each pass and exits the loop if

another instruction is pending. The only exception is the NULL instruction where the sequencer returns Processor DSACK and continues. NULL can be used to periodically check that the FPORT Controller (or FASTBUS) is not locked up.

The LIST instructions rely on programming of the FPORT Controller microcode which is EPROM based. Use should be limited to applications requiring very fast operation. In general, the standard primitive instructions provide reasonably fast (Processor speed) access to FASTBUS without changes to microcode.

2.3.1.1 Data Transfer Description and Transfer Rates

The FSCC supports Pipelined Transfer rates of 100, 200 and 400 nsec per word. At the 100 nsec per word rate, End-of-Block is not normally returned in time to avoid another DS transition. Therefore, Pipelined transfers at 100 nsec will read one word beyond the end of block and this word will be included in the output data stream. Other pipelined transfer rates, in any multiple of 50 nsec with a minimum of 100 nsec, can be programmed through minor microcode changes.

The FSCC also supports Block Transfers at a rate of ≥150 nsec per word. The actual transfer rate will be 100 nsec plus the slave DS-DK response time, rounded to the next highest 50 nsec increment.

Block and Pipelined Transfers directed to the Data FIFO are functional for Read mode only. Data is placed in the FIFO and is not accessible to the Processor. However, data can be simultaneously routed to the Processor FIFO by setting the COPYEN bit in the parallel port. A Block Transfer Read/Write instruction for Processor memory is available, but operates at the same speed as single word Processor read and write operations. It implements one step of the block transfer for each processor MOVE instruction executed. The block transfer must be terminated with a DATA_PROCESSOR_BLOCK_TRANSFER_TERMINATE instruction.

Although the Data FIFO is limited to blocks of 2K words or less, larger blocks can still be transferred provided that the destination does not require a leading word count or is capable of interpreting a word count of 2048 as a partial transfer. In the second case, for block transfers which are some exact multiple of 2048 words, a final block of length zero must be transferred. If leading word count insertion is not required and the output port runs at the same rate as the FASTBUS port, the FIFO depth is not a factor.

2.3.1.2 Internal Control and Status Registers

All Control and Status Registers related to FASTBUS operation are implemented in 68020 software. The number of registers is limited only by available processor memory.

2.3.1.3 Error Responses

The FPORT Controller latches FASTBUS error conditions but does not attempt any recovery or retry. A processor interrupt is generated and the processor has the option of attempting recovery or, more likely, skipping the entire readout sequence. Because of internal pipelining, errors may not be reported immediately with the bus cycle generating the error. Also, the amount of status information available to the processor is limited.

The standard FASTBUS instructions will abort on errors and return to the processor with an interrupt. The user can optionally modify the microcode to ignore errors and continue with the FASTBUS operation, although the results may be undefined. If user supplied microcode is capable of recovering from errors without processor intervention, the processor error interrupt can be disabled. An error flag can be set in the header word of the current output event to indicate that the FASTBUS event readout failed and the data should be discarded. The FPORT Controller does not have the ability to test status flags in slave modules to locate problems, so error recovery through microcode is generally limited to reset and continue type operations.

2.3.2 Ethernet

The INTEL 82586 LAN coprocessor performs message framing management in transmission and reception functions. It acts as a bus master, accesses memory by DMA, carries out message error checking, collision recovery functions, etc. The INTEL 82C501 Ethernet Serial Interface (ESI) implements Manchester encoding/decoding and clock recovery. ESI functionality may be checked by the processor using Loopback mode. The National DP8392 Ethernet Transceiver performs collision detection and interfaces to the coaxial cable.

Ethernet Interface

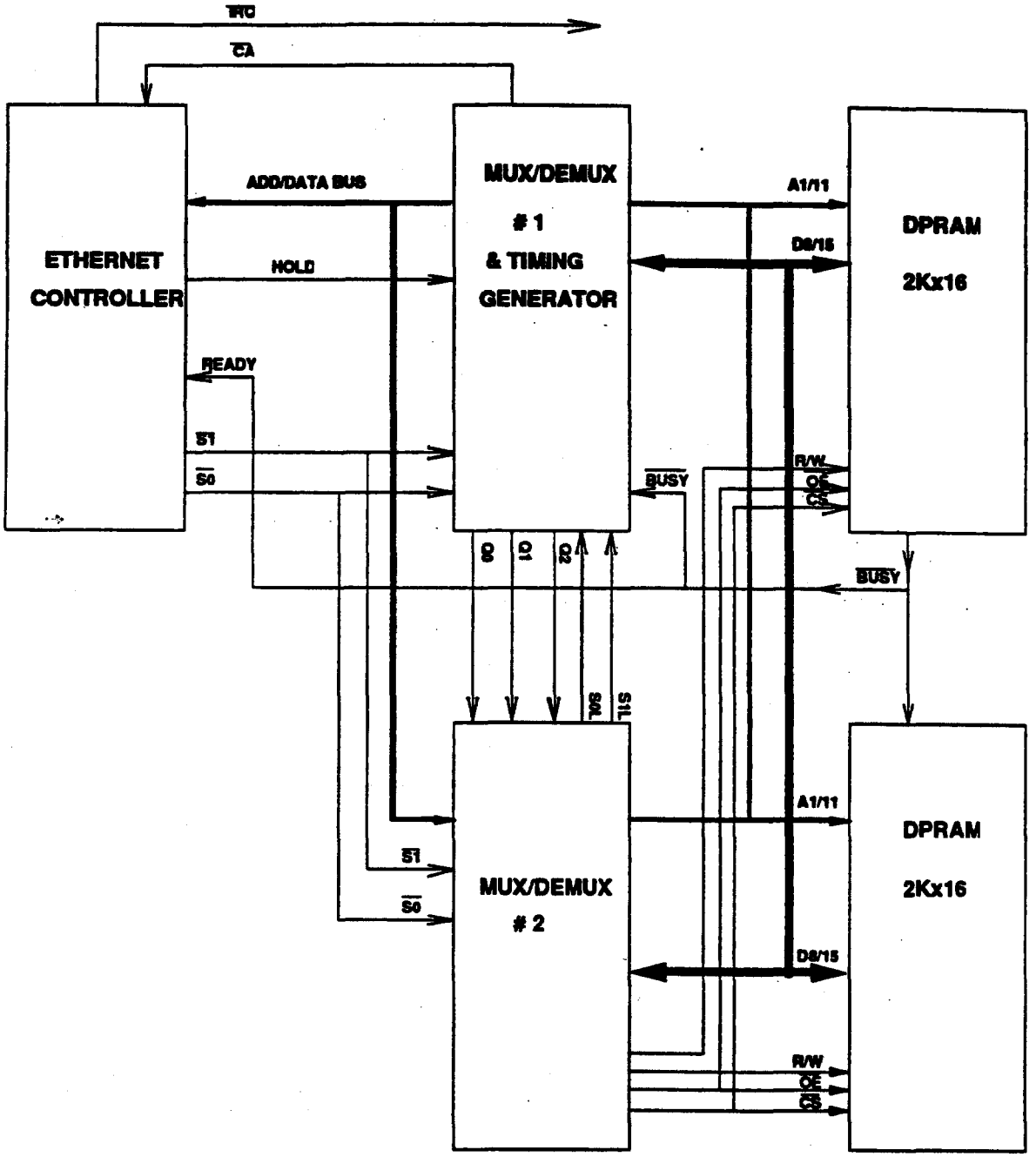
The INTEL 82586 coprocessor interfaces indirectly to the 68020 CPU system through a 4K by 16 bit dual ported memory. Figure 4 is a Block Diagram of the Ethernet Interface.

The dual port RAM speeds up both systems as the processor does not release the bus each time a DMA occurs, also controller bus latency is reduced to zero.

Direct interface is provided with CA and IRQ lines. The 68020 CPU drives CA to get the attention of the controller, indicating that new commands were included in the command list to be processed.

The 82586 Ethernet controller uses IRQ to interrupt the processor when a command is over or upon message reception.

The shared DPRAM structure is composed of four parts: The Initialization Root, the System Control Block (SCB), the Command List and the Receive Frame Area (RFA). The Initialization Root is fixed in memory. The Ethernet controller addresses that variable as **$FFFFF6**, but in fact, due to the partial decoding, its physical location is **$1FF6**. The origin of the DPRAM is **$40000** from the CPU side and **$0000** from the Ethernet controller. Transmission and reception messages are split into small buffers to better use the available memory. The buffers, when necessary, are chained in frames. Transmission and reception buffer descriptors are accessible through the SCB table pointers.



Ethernet Interface Block Diagram

Figure 4

Ethernet Controller Interface

The interface between the Ethernet controller and the DPRAM consists of two ALTERA EPB1400s. The Ethernet controller, address and data buses are multiplexed, so the EPB1400s demultiplex/remultiplex those buses and generate the necessary timing for the memory signals: CS*, R/W*, OE*.

Logic in the EPB1400s is used for as a state machine (Figure 5) to control timing. This logic monitors the state of the BUSY line from the DPORT RAM. To indicate a memory access contention in read and write cycles, BUSY* is connected to the READY line of the controller and inserts wait states.

In both, read and write cycles, the state machine's counter is kept at states 3 and 4 when BUSY* is activated.

Timing Diagrams

The Ethernet controller was designed to share an external bus, it asserts a HOLD line to keep the bus when performing burst memory cycles. The controller has a private bus to the DPRAM, so hold acknowledge HLDA is returned instantaneously, and the state machine goes directly to IDLE to wait for the beginning of a cycle.

S1 and S0, from the Ethernet controller indicate the type of cycle: S1=0, S0=1 is a read cycle and S1=1, S0=0 is a write cycle. Those lines are driven only during T1 and T2, and are latched in S1L and S2L.

Figures 6a and 6b show timing diagrams for read and write cycles. Table one is a truth table.

2.3.3 Description and Operation of OUTPUT PORT

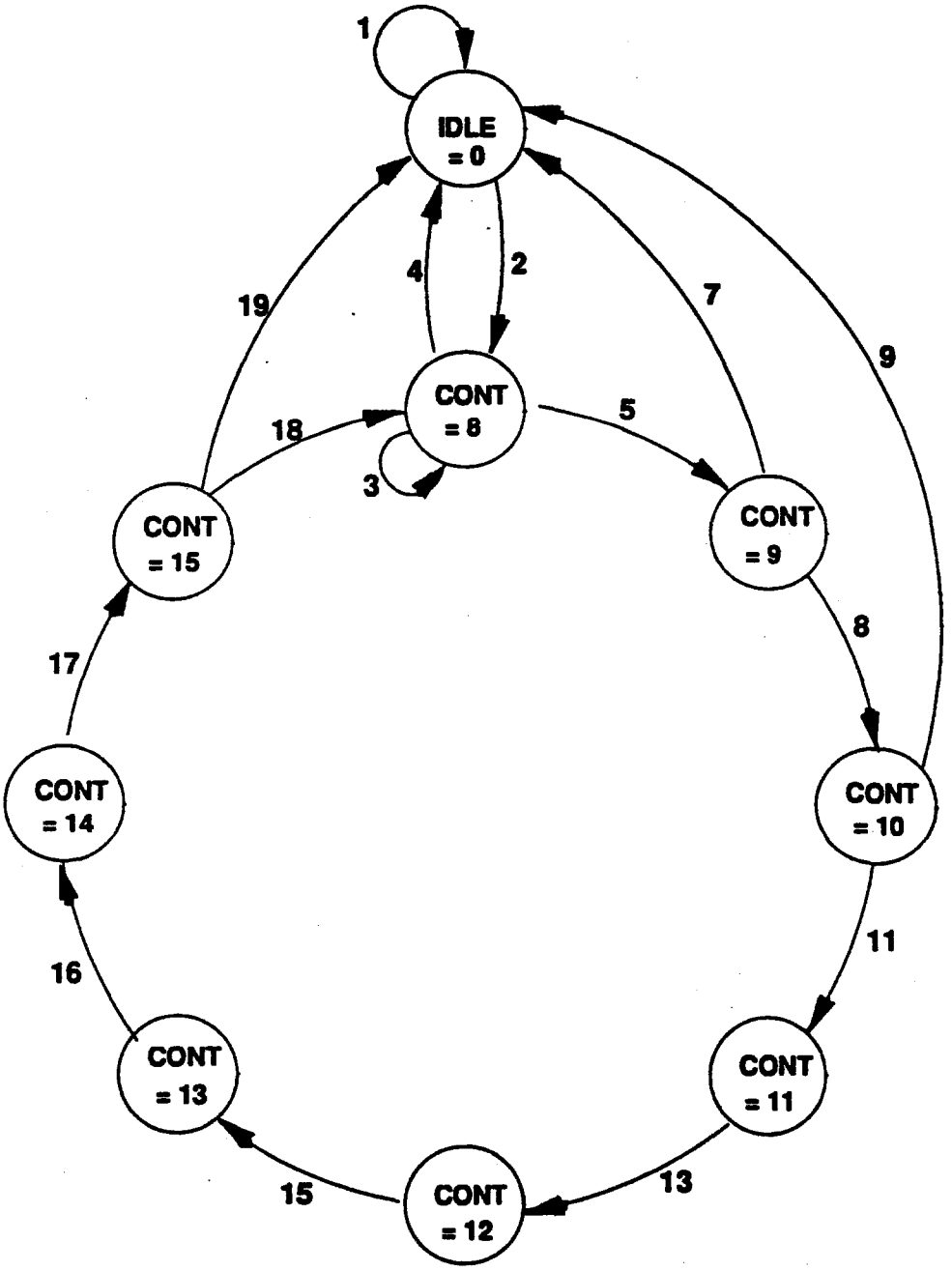
The basic function of the Output Port Controller (OPORT) is to provide a continuous stream of data with strobes for transmission via RS-485, ECLine or fiber-optic auxiliary cards. A Block diagram of the Output Port is shown in Figure 7. For this application, it need only multiplex the FIFO outputs and generate appropriate timing. Connection to more complicated external bus formats will require reprogramming of the OPORT Controller or addition of a more sophisticated sequencer on the auxiliary card. The OPORT can also be disabled to allow an external controller mounted on the FASTBUS auxiliary card to take over data transmission. This allows for various protocols to be implemented by fitting different auxiliary cards to the controller.

All levels at the output port are single ended TTL. Level adapters to different protocols will be mounted on the FASTBUS auxiliary card. The data path is 32 bits wide. Two control lines are provided to regulate data flow: the WAIT input pauses the OPORT and STROBE is a synchronous data strobe output.

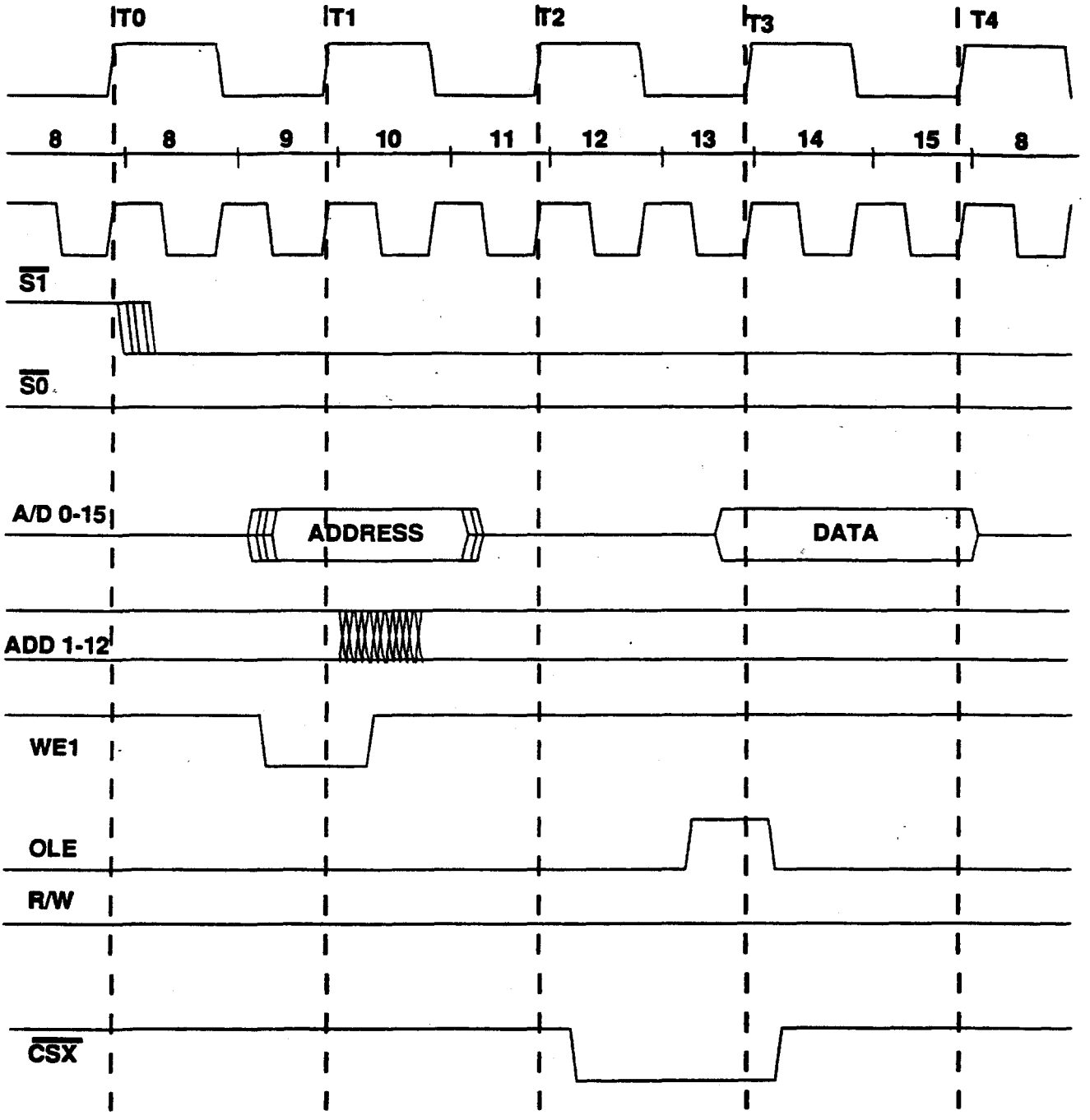
Ethernet Controller Truth Table

	HOLD	S1	S0	BUSY*	COUNT	COUNT
	0	X	X	X	0	0
	1	X	X	X	0	8
	1	1	1	X	8	8
	0	X	X	X	8	0
READ CYCLE						
	1	0	1	X	8	9
	0	X	X	X	9	0
	1	1	1	X	9	0
	1	0	1	X	9	10
	0	X	X	X	10	0
	1	1	1	X	10	0
	1	0	1	X	10	11
	X	X	X	X	11	12
	X	X	X	X	12	13
	X	X	X	H	13	14
	X	X	X	L	13	13
	X	X	X	X	14	15
	1	X	X	X	15	8
	0	X	X	X	15	0
WRITE CYCLE						
	1	1	0	X	8	9
	0	X	X	X	9	0
	1	1	1	X	9	0
	1	1	0	X	9	10
	0	X	X	X	10	0
	1	1	1	X	10	0
	1	1	0	X	10	11
	X	X	X	X	11	12
	X	X	X	X	12	13
	X	X	X	H	13	14
	X	X	X	L	13	13
	X	X	X	X	14	15
	1	X	X	X	15	8
	0	X	X	X	15	0

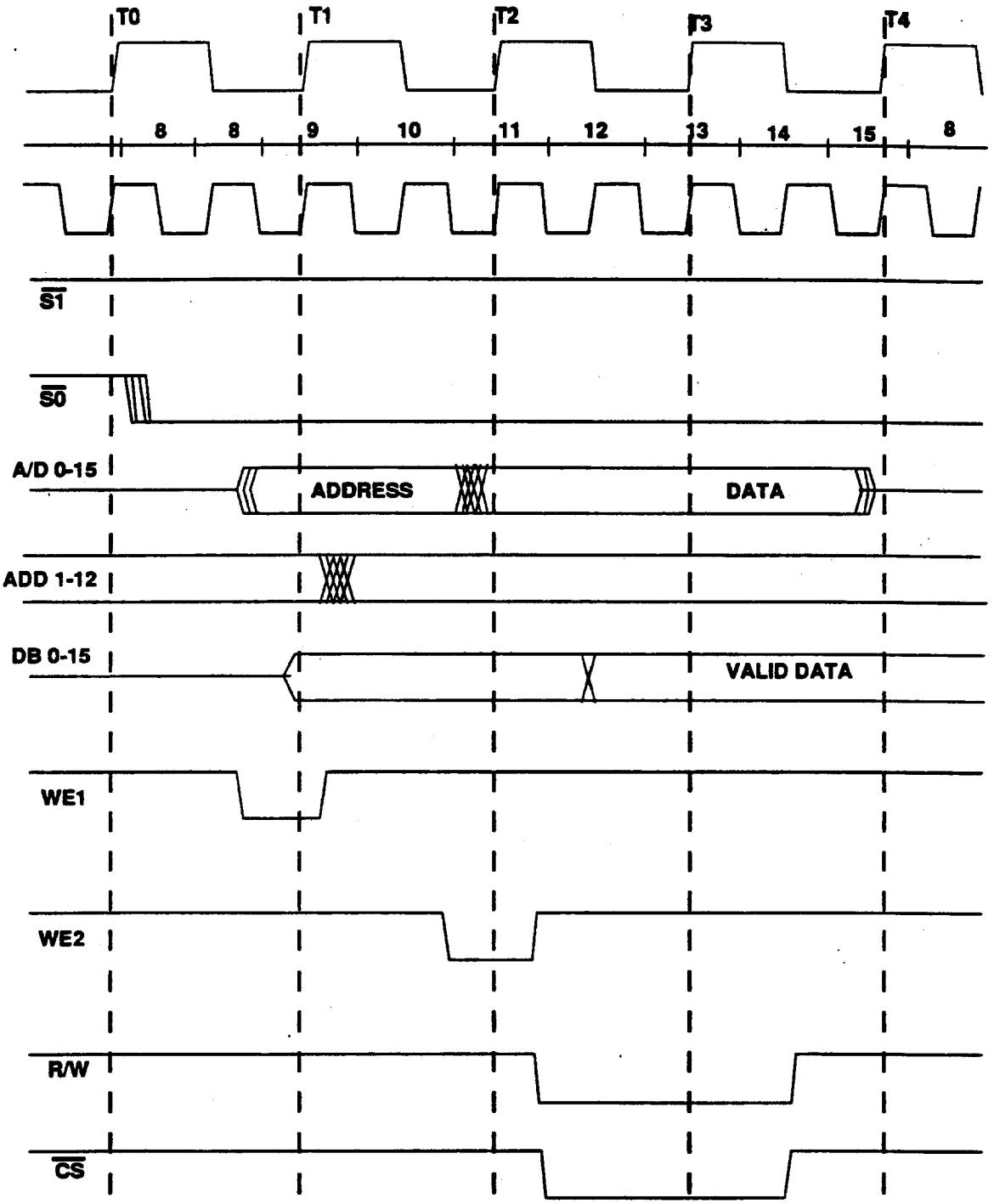
Table 1



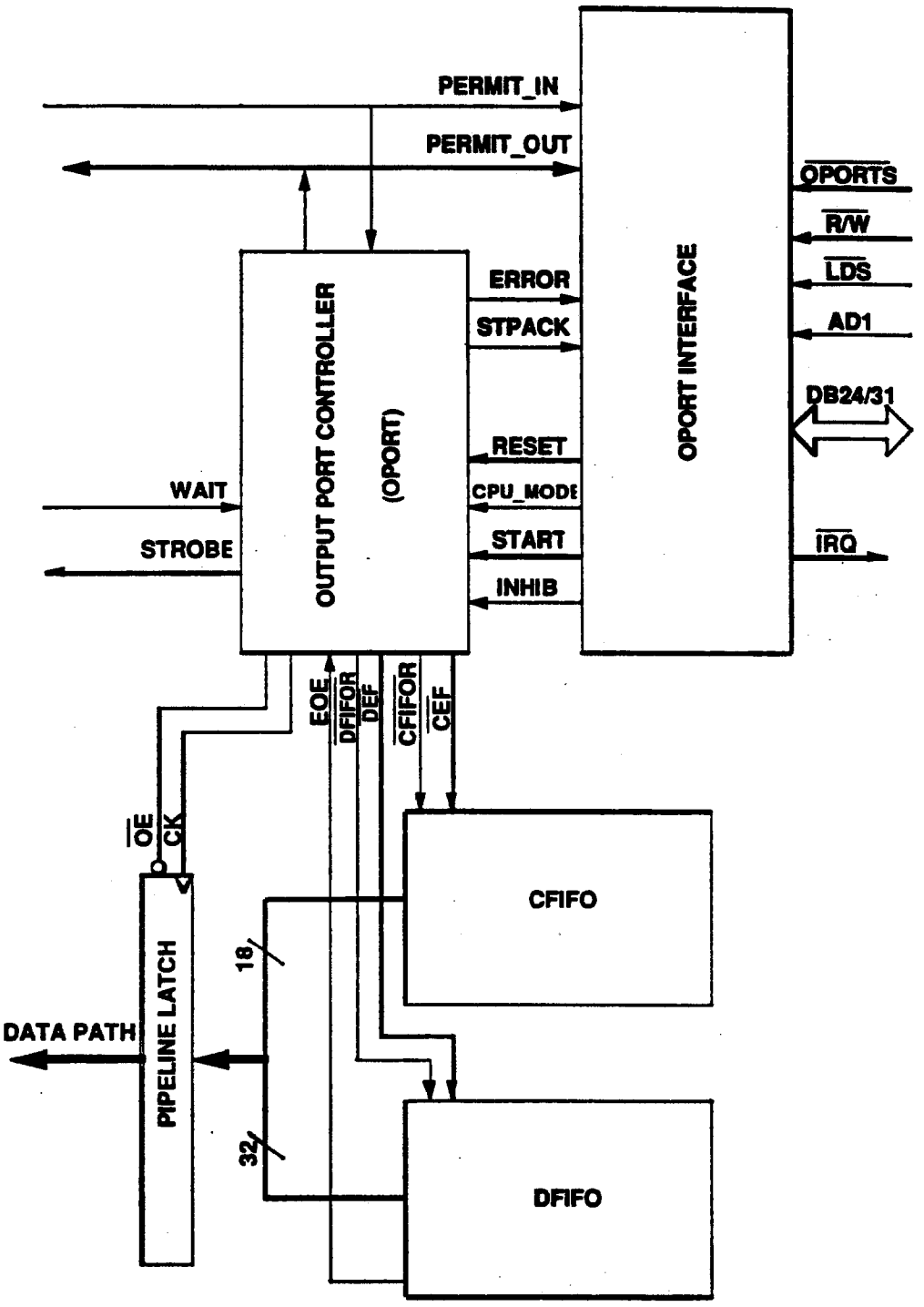
Ethernet Controller State Machine Diagram
Figure 5



Ethernet Controller Read Cycle
Figure 6a



Ethernet Controller Write Cycle
Figure 6b



OPORT Block Diagram

Figure 7

In some applications the auxiliary bus will be common to several FSCC modules. The PERMIT_IN and PERMIT_OUT signals provide a "token passing" mechanism for enabling and disabling data output. The external WAIT input, located on the auxiliary card, can also be used to regulate the data output rate.

The OPORT Controller performs arbitration and control for the Data and Word Count FIFO output ports. It has two modes of operation; *Event Mode*, where a word of Control information is transmitted from the Control FIFO followed by synchronous data from the Data FIFO at a rate of 10 MHz, and *Processor Controlled Mode*, where data is transmitted from the Control FIFO only. The operating mode is selected by the processor.

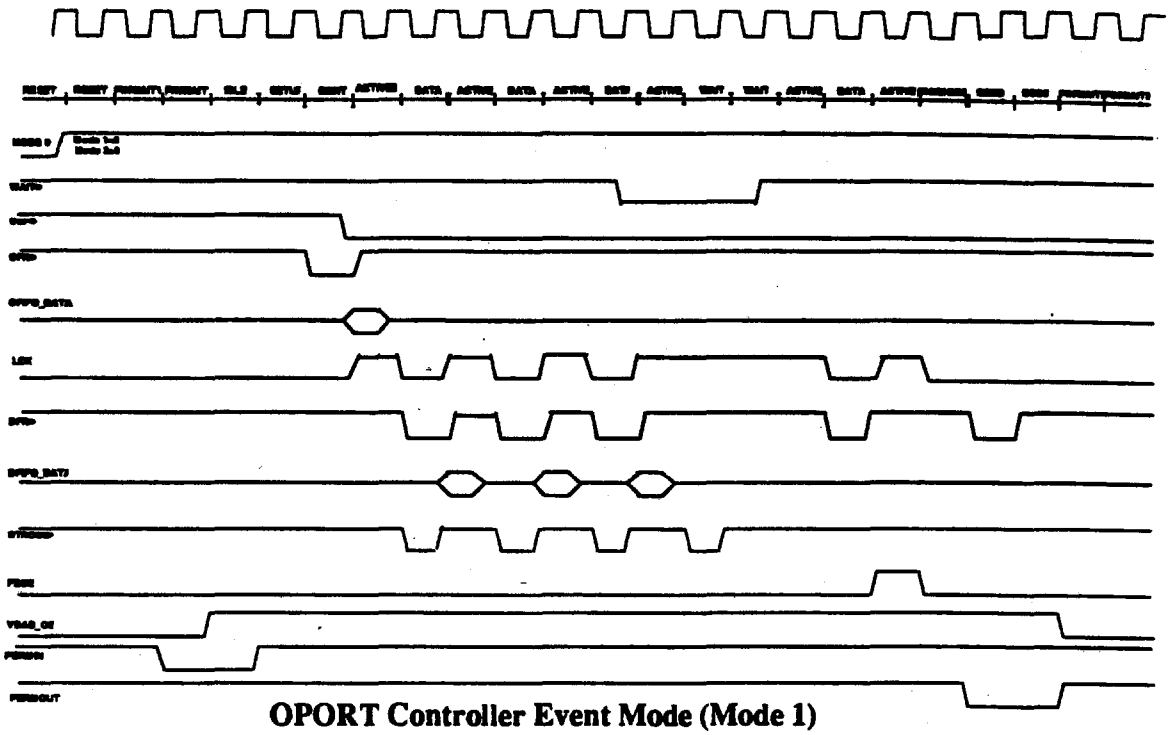
Event Mode : This is the normal data taking mode and data is transmitted to the output with synchronous, no handshake, protocol at 10MHz or 40 Mbyte/sec. The data contains a non-inclusive leading word count,(the lower order 12 bits). That header word from the control FIFO (CFIFO) is transmitted first, followed by the specified number of words from the data FIFO. Bits 12-16 contain a processor programmable header, usually the Trigger ID, and bit 17 is an error flag. Bits 18-31 of the control word are undefined and transmitted as zeros. A pipeline latch between the data FIFO (DFIFO) outputs and the output port stabilizes the data field. A pipeline register holds the data on the Output Port for a period of approximately 50 nsec preceding and 50 nsec following the trailing edge of the output data strobe (STROBE*). Wait states can be inserted by the auxiliary port in increments of 100 nsec. Refer to figure 8a for an example of Output Port timing in normal frame mode.

Processor Controlled Mode: In this mode, data is written into the Control FIFO one word at a time by the processor using the H&C transparent mode (refer to 2.3.3.3.3). Then, the MODE signal is driven high and the OPORT Controller extracts data from the Control FIFO for transmission. The processor can read the Step Acknowledge (STPACK) status bit to confirm transmission or can be interrupted. Data width is limited to 17 bits. Transmission of data in processor controlled mode is also subject to external WAIT. Refer to figure 8b for an example of timing in the processor controlled mode.

Refer to figure 9 for a state transition diagram of the OPORT Controller. At power up or RESET, the state machine is in POWER ON RESET. Receipt of a FORCE_EVENT command takes the controller directly to IDLE, while an EVENT, STEP or EMPO command puts the controller into PINWAIT. A PERMIM will then take the controller from PINWAIT to Idle. As soon as the control FIFO has a word in it, the controller will move from IDLE to SETUP state. A STEP command will take it to STEP state, any other mode will allow the controller to continue on to CONT state. However, if the WAIT signal is high before any data has been transmitted, the controller assumes that the memory buffers are not connected and enters the ERROR state asserting the ERROR signal. A RESET is required to return the controller to IDLE.

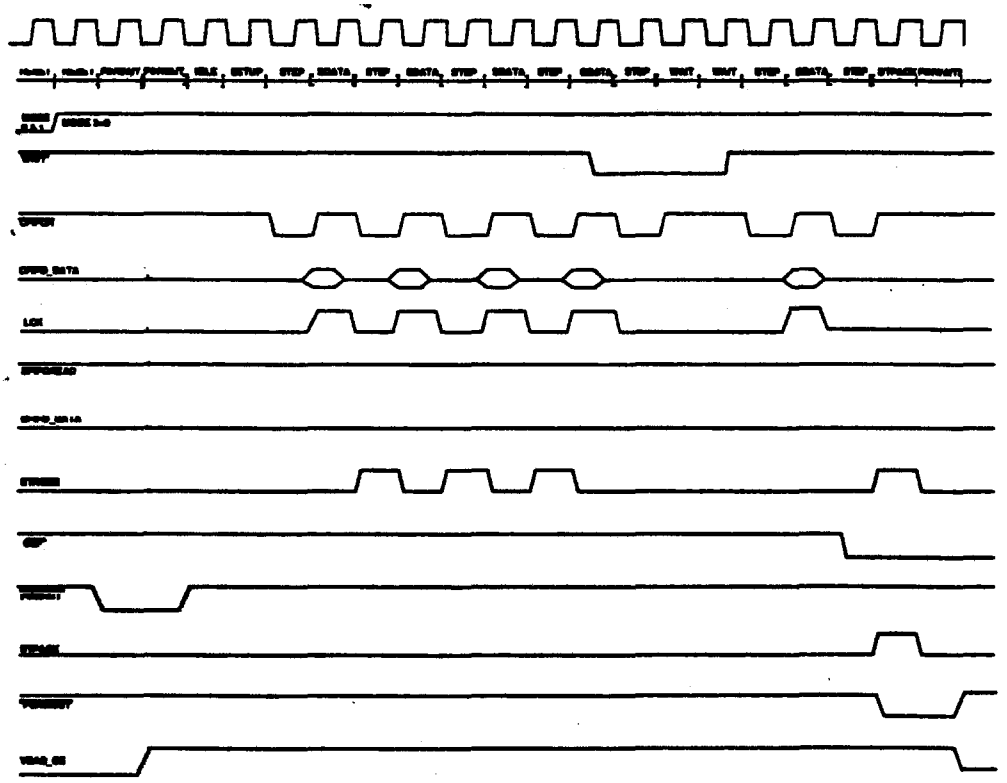
From the CONTINUE state (Event Mode) it proceeds to the ACTIVE2 state and transmits one word from the Control FIFO. It then checks for data in the Data FIFO. As long as the WAIT is low, the controller continues to the DATA state, then loops between ACTIVE and DATA states, transmitting one word of data each cycle. The block transfer is terminated by the EOE signal from the Data FIFO.

From the STEP state (Processor Controlled mode), the controller transmits data from the Control FIFO until FIFO empty, and then asserts the STPACK signal. The processor must initiate the next cycle.



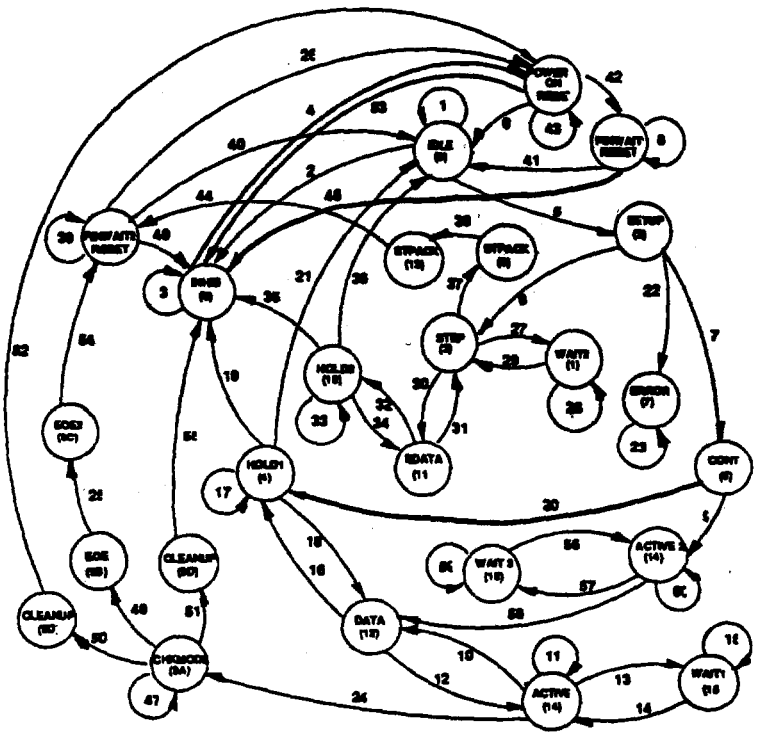
OPORT Controller Event Mode (Mode 1)

Figure 8a



OPORT Controller. CPU Controlled Mode

Figure 8b



OPORT Controller State Machine Diagram

Transition	Branch	Data	Condition	Transition	Branch	Data	Condition
0	2	[out 00]	If Mode=2	30	3	[out 11]	
1	3	[out 00]	Idle loop	31	1	[out 3x]	If Mode=3
2	1	[out 08]	If Mode =4	32	2	[out 10]	
3	1	[out 08]	Mode=4	33	4	[out 10]	
4	2	[out 00]		34	3	[out 11]	If Mode=3
5	2	[out 02]	If Mode=1 or 2 or 3 or 5	35	1	[out 08]	If Mode=4
			/CEF	36	2	[out 00]	If Mode=1 or 2 or 5
6	2	[out 03]	If Mode 3	37	2	[out 05]	If /CEF *
7	3	[out 06]		38		[out 13]	
8	3	[out 00]	Idle loop	39	4	[out 00]	Idle loop
9		[out 14]		40	1	[out 00]	If PERMIN
10	3	[out 12]	If /DEF*	41	1	[out 00]	If PERMIN + Mode=2
11	4	[out 14]	Idle loop	42	1	[out 00]	If Mode=1 or 3 or 5
12	2	[out 14]		43	4	[RESET]	
13	1	[out 15]	If WAIT*	44		[out 00]	
14	1	[out 14]	If /WAIT*				
15	2	[out 15]	Idle loop	46	2	[out 08]	If Mode=4
16	1	[out 04]	If Mode =0	47	4	[out 9A]	
17	4	[out 04]	Idle loop	48	1	[out 9B]	If Mode=6 or 1 or 2
18	3	[out 12]	If Mode=1 or 2 or 5	49	3	[out 08]	If Mode=4
19	1	[out 03]	If Mode=4	50	2	[out 14D]	If Mode=0
20	1	[out 04]	If Mode=0	51	3	[out 9D]	If Mode=4
21	2	[out 00]	If Mode=3	52		[RESET]	
22	1	[out 07]	If WAIT	53	3	[out 08]	If Mode=4
23		[out 07]	Idle loop	54		[RESET]	
24	2	[out 9A]	If EOE	55		[out 08]	
25		[out 9C]		56	1	[out 15]	If WAIT*
26	2	[out 00]	If Mode=0	57	1	[out 14]	If /WAIT*
27	1	[out 01]	If WAIT	58	2	[out 12]	If /DEF*
28	2	[out 01]	Idle loop	59	2	[out 15]	If WAIT*
29	1	[out 03]	If /WAIT	60		[out 14]	

Figure 9

OPORT Controller State Machine Outputs

State	Outputs
RESET	LCLR*
0	LCLR*,LOE*,PARLL(11)(VDAS_OE)
1	CFR*,LOE*,PARLL(11)(VDAS_OE)
2	LCLR*,LOE*,PARLL(11)(VDAS_OE)
3	CFR*,LOE*,PARLL(11)(VDAS_OE)
4	LCK,DFR*,LOE*,PARLL(11)(VDAS_OE)
5	STPACK,LOE*,PARLL(11)(VDAS_OE),PERMOUT*
6.	CFR*,LOE*,PARLL(11)(VDAS_OE)
7	ERROR,LCLR*
8	no output
9A	LCLR*,LOE*,PARLL(11)(VDAS_OE)
9B	DFR*,PERMOUT*,LOE*,PARLL(11)(VDAS_OE)
9C	LCLR*,LOE*,PERMOUT*,PARLL(11)(VDAS_OE)
9D	DFR*,LCLR*
10	LCK,LOE*,PARLL(11)(VDAS_OE)
11	LCK,LOE*,PARLL(11)(VDAS_OE)
12	STROBE,DFR*,LOE*,PARLL(11)(VDAS_OE)
13	STPACK,LOE*,PERMOUT*,PARLL(11)(VDAS_OE)
14	LCK,LOE*,PARLL(11)(VDAS_OE)
15	LCK,LOE*,PARLL(11)(VDAS_OE)

OPORT Input/output signals

The processor can control and monitor the OPORT through an EPB1400 parallel port interface. The following signals are used to configure and monitor the OPORT.

RESET: This asynchronous line forces the OPORT into an inactive state if an external controller is to be used.
MODE 2-0: Selects one of seven possible OPORT operating modes.

Command	Hex-Code	Function
Hold	$00	Send a HOLD request to the OPORT. Normal execution can be resumed any time.
Event_Mode	$01	Sets the OPORT in Event Mode execution.
Force_Event_Mode	$02	Sets OPORT in Event Mode with no PERMIN Required.
CPU_Mode	$03	Sets the OPORT in Processor Cont. Mode execution
Inhib_Mode	$04	Puts the OPORT off-line. The output lines are floated.
EMPO	$05	Event_With_Manual_Permit_Out_Mode: Functionally same as Event_Mode except that upon completion of the event data transfer, the controller waits with it's outputs enabled and does not generate a Permit Out until a PO command is received.
the		
PO	$06	Permit_Out_Mode: Used in conjunction with the EMPO command, it tells the controller to generate Permit_Out, disable it's outputs and return to PINWAIT state until a new token arrives.
Opo_Reset	$08	Send a reset signal to the OPORT.

The OPORT controller drives output lines to indicate status conditions to the processor:

STPACK: Indicates the CPU controlled cycle is over..
ERROR: Indicates that WAIT line has been detected active before the beginning of a cycle. In that case the OPORT assumes the receiver is unconnected and halts itself indicating the error condition with ERROR line.
SM(0-3): Return the OPORT state machine status.

External interface:

PRMOUT: Enable next device onto the token passing logical ring.
PERMIN: Indicates the token has been received.
STROBE: Signals active data on the pipeline latches, the rising edge is used to strobe that data into the personality card.
WAIT: This input stops the output data stream either in Event or Processor Controlled Modes.

The OPORT is controlled from the CPU by programming the 8 bit control register in the OPORT interface. Refer to output port controller interface for programming codes and status conditions.

2.3.3.1 OUTPUT PORT Controller Interface

The output port controller interface (OPO_INTF) allows the CPU to control and monitor the output port microsequencer (OPORT). The CPU sets up the OPORT in one of the 5 defined modes and receives OPORT status information. The OPO_INTF also drives an interrupt to the CPU system under some OPORT conditions.

Configuration:

The OPO_INTF is a byte wide port, placed at address **$4A0000** in the memory map. Its internal architecture consists of one control and two status registers. The control register sets up the OPORT modes allowing it to individually mask the three defined interrupts. The OPORT defined modes are: Inhibit, Event_Mode, Force_Event_Mode, CPU_Mode, Hold, EMPO, PO, OPO_Reset. They are controlled by individual bit settings.

Address	Write Function	Read Function
$4A0000	None	Status register 1 **b3-b0:** State Machine(Table XX) **b4:** Not Implemented **b5:** Permit_in line flag **b6:** STPACK flag **b7:** ERROR flag
$4A0001	Control register **b0-b2:** Mode **b3:** OPO_Res **b4:** Undefined **b5:** CPU_Md_Mask **b6:** Not Implemented **b7:** Error_Mask	Status register 2 **b0-b3:** 4 LSB's of the control register **b4:** Permit_out flag **b5 :** STPACK mask status **b6:** Not Implemented **b7:** ERROR mask

The software reset has the same effect over the OPORT that the hardware reset does. The OPORT goes to the power on reset state, and the control FIFOS are cleared. The INTF output lines.are cleared during a reset operation until a command word is written to the INTF command register.

Three interrupt conditions are recognized by the OPO_INTF:
STPACK: indicates the end of a CPU_mode cycle.
INCOMP: signals that a FASTBUS error has occured during the readout of an event.
ERROR: indicates that there is no receiver connected to the FSCC.

The interrupt conditions are also reflected in the status registers, along with the OPORT state machine status. A read from this register clears the interrupt flags.

2.3.3.2 OUTPUT PORT State Machine

OPORT States:

HOLD.....0
WAIT2 1
SETUP..... 2
STEP 3
HOLD1.....4
STPACK1 5
CONT..... 6
ERROR 7
INHIB.....8
EOE 9
HOLD.....10
SDATA..... 11
DATA.....12
STPACK2 ...13
ACTIVE..... 14
WAIT 15

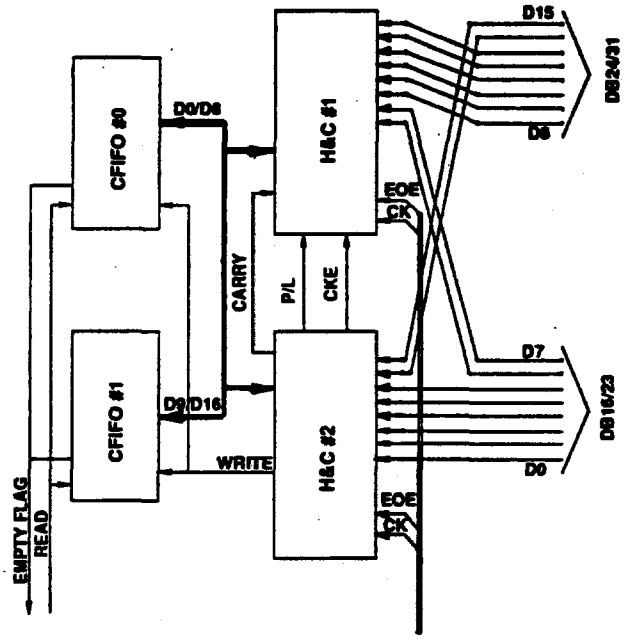
Timing

The OPORT reads the FIFOs and generates the pipeline latch clock, the output data strobe, and the output latch tri-state enable. The timing diagrams are shown in figures 8a and 8b.

2.3.3.3 Total Header and Event Counter Control System

Header and Counter (H&C)

The H&C system keeps a crate wide total count of words read from FASTBUS and transmits this count to a FIFO upon receipt of an EOE (End of Event) signal. The header is a 5 bit word which is directly loaded by the processor into the H&C system. The 12 bit word counter is incremented by a clock signal (CK) coming from the FASTBUS controller. A block diagram is shown in Figure 10.



H&C Block Diagram
Figure 10

Functional Blocks

The H&C system uses two EPB1400 devices. One of them (HEADB) configures the header and the counter's 6 high order bits; the other (HEADA) has the counter's 6 low order bits and the H&C state machine controller. The CPU sees the H&C system as a 16/8 bit peripheral. Four registers can be accessed from the processor bus.

a) Command Register (Write only):
- **RESET**: H&C content are cleared and the state machine is forced to HOLD state.
- **LOAD**: The counter is loaded with the preset register's contents.
- **HOLD**: The count value is kept constant but not cleared.
- **INCREMENT**: Counter increments are enabled and clocked by CK signal.

b) Two counter preset registers.
c) A header register
d) A STATUS register (Read only) which echoes the command register bits and the status of the control state machine.

System Interface

The 12 bit counter has been split in two 6 bit counters. The least significant counter (LSC) generates a CARRY signal to the MSC each time it reaches its maximum value. From the CPU bus, the counters and other H&C registers are accessible in bytes or words.

The internal registers may be selected through select and control lines: H&CSEL*, R/W*, UDS*, LDS* and A1.

A1	UDS	LDS	R/W	Reg. Selected
0	0	0	X	12 bit counter
0	0	1	X	Lower 8 bits of the Counter
0	1	0	X	Upper 4 bits of the counter
1	0	1	X	Header
1	1	0	0	Command
1	1	0	1	Status

The 12 bit counter is placed at **$4E0000** or in consecutive bytes from that address. The lower 12 bits of a 16 bit transfer are loaded into the Preload register during a write cycle.

Address	Write Function	Read Function
$4E0000	Preload register **b0-b11:** Preload count **b12-b15:** Unassigned	Counter Register **b0-b11:** Counter contents **b12-b15:** Unassigned
$4E0002	Header register **b0-b4:** H0-4 Header preload **b5-b7:** Unassigned	Header register **b0-b4:** H0-4 Header contents **b5-b7:** Unassigned
$4E0003	Control register **b0-b1:** S0,S1 Command select field **b2:** RF Software reset bit **b3-b7:** Unassigned	Status register **b0-b1:** S0,S1 Command select field **b2-b5:** Q0-Q3 H&C state status **b6-b7:** Read as zeros

The command modes can be set using the following values:

RESET:	$04
PRELOAD:	$01
HOLD:	$00
TRANSPARENT:	$03
INCRM:	$02

The interface between the H&C controller and the FASTBUS sequencer is through FCLK (counter clock) and CEOE (End of Event) lines. Both are asynchronous to the 20 MHz clock of the state machine. FCLK increments H&C counter when it is in Increment mode CEOE indicates the count is over and the H&C value should be transfered to the CFIFO.

STATE MACHINE

The following states (Figure 11) have been defined in the H&C state machine controller:

OFF_LINE: Is the power up state. At this time the control register content is unknown.
HOLD: This is the idle state. The counter value is kept constant.
PRELOAD: The counter's state is initialized with the value found in the counter input
INCREMENT: Enable the FASTBUS clock to increment the counter.
WRITE: Performs a Write memory cycle to the CFIFOs.
CLEAR: Clears the counter after an EOE cycle.
PASS: Enable data written into the counter's input register to be loaded directly into the FIFO memory.
WDAT: Manages PASS mode detecting DS* (data strobe) line active.
DSHIG: Similar to WDAT's function but detects DS* line low-to-high transition.

Interface between HEADA and HEADB

Only one of the two EPB1400s (i.e. HEADA) has the state machine which controls the H&C functioning. It generates signals for controling the other device.

PL: This signal indicates to HEADB that it has to perform a load operation from the input registers.
CKE: Enables HEADB counter increment on CK clock signal.

Internal Signals:

RES and CLR signals are generated for internal use:
RES is a logical OR of the external RESET line, the command register RF bit and the CLEAR state. The RES line is generated in both devices and feeds the counter's macrocells. CLS line is only generated in HEAD2 and clears the MASK bit when the RF bit is set.

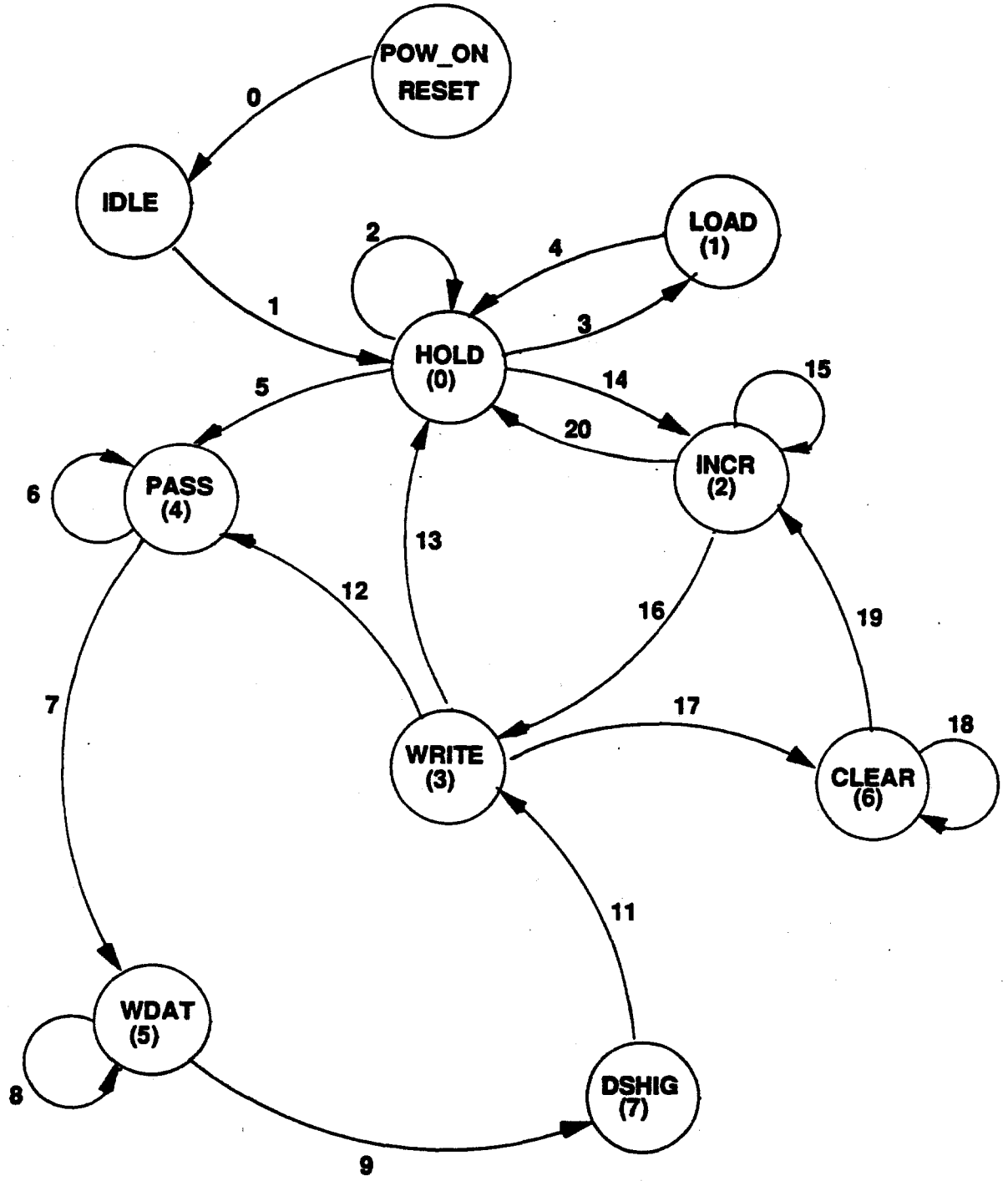
The WRITE signal also functions as a tristate enable for the counter and header outputs during FIFO memory write cycles.

2.3.3.4 OUTPUT PORT Auxiliary Connector Interface

All levels at the output port are single ended TTL. Level adapters to different protocols will be mounted on the FASTBUS auxiliary card. The data path is 32 bits wide. Three control lines are provided to regulate data flow:

WAIT pauses the OPORT
STROBE is a synchronous data strobe

The OPORT generates the pipeline latch clock, the output data strobe, and the output latch tri-state enable. Timing diagrams are shown in figures 8a and 8b.



Header/Counter State Machine diagram
Figure 11

2.3.4 Communication Protocols

The following control line assignments apply to the RS-485 VDAS interface;

AC15-12	not used
AC11	Output Enable
AC10	WAIT
AC09	not used
AC08	STROBE*
AC07-00	not used (Processor controlled I/O)

The following control line assignments apply to the RS-485 RBUF interface;

AC15-13	not used
AC12	Word Select
AC11	Output Enable
AC10	WAIT
AC09	SSTROBE*
AC08	DSTROBE*
AC07-00	not used (Processor controlled I/O)

The following control line assignments apply to the ECLine interface;

AC15-12	not used
AC11	Output Enable*
AC10	WAIT
AC09	End-of-event
AC08	STROBE*
AC07-00	not used (Processor controlled I/O)

The following control line assignments apply to the Fiber-optic interface;

(TBD)

3 System Software Description

3.1 System Software

The operating system is a commercial product of 'Software Components Group, Inc.' which is composed of several related products.

pSOS -68K; A real-time multi-tasking operating system kernel.
pROBE-68K; System debug analyzer.

These combined products form the operating system of the FSCC. The pSOS-pROBE-pRISM operating system is configured at Fermilab for the FSCC hardware implementation. The operating system and diagnostic software are contained in on-board EPROM. A second bank of EPROM is included to contain user programs which may run independently of use services (eg. serial port driver) provided by the operating system. Please refer to "Diagnostics for the FASTBUS Smart Crate Controller", see appendix D.

3.2 Initialization

A reset is generated at power on or from the front panel reset, the FSCC responds by performing initializing all ports & controllers to a passive configuration and sending the following prompt to the serial port:

```
PROBE V3.14 (68020)
COPYRIGHT 1986, SOFTWARE COMPONENTS GROUP INC.
ALL RIGHTS RESERVED

pROBE>
```

The following items 1-4, will need to be set by the user on the first power on cycle. As the values are contained in NVRAM this operation need only be done once unless a change is required.

1. Set the real time clock.
2. Set Ethernet Address
3. Set FASTBUS ID
4. Terminal and Host ports are configurable by writing to the NVRAM at address 30000E as follows:

Byte	bit		
Hex	6	5	
00	0	0	Host and terminal functions use TERM port.
20	0	1	Host function at HOST port and terminal function at TERM port
40	1	0	Host function uses the TERM port and terminal function uses the HOST port.
60	1	1	Host and terminal functions use the HOST port.

These bits should be selective-set as other bits in the byte may be used for other purposes. These bits will normally be set to 20 Hex when testing is completed.

3.3 PROBE Initialization

Because PROBE is designed to be independent of the hardware implementation it doesn't directly access any hardware devices (other than the microprocessor itself). The only device it accesses indirectly is the serial port. The boot code in the EPROM provides the routines to pROBE that it uses to access the serial port. This boot code also initializes whatever other hardware on the board is not put into a known or usable state by RESET. Once pROBE is called, it loops on polling for input from the serial port. If there is a problem with the serial port hardware, pROBE may not be able to communicate at all.

The boot code was originally debugged using XRAY, emulating the serial port.

3.4 PROBE Memory Map

System ROM for the FSCC nominally occupies 64K bytes of EPROM bank 1. It consists of four sections at offsets that are fixed, i.e., hardwired into the boot code source file and intended to remain invariant for all future versions. These four sections and their offsets (from the start of ROM) are:

Section 1	Offset	Function
	0	boot code
Section 2	1000	pSOS code
	2700	dummy pSOS configuration table
	2762	dummy pSOS I/O jump table
Section 3	2800	pROBE code
Section 4	CA00	pRISM code (not used on the FSCC)

The first 4K bytes are allocated for boot code, configuration tables, the pROBE serial port driver, etc. The boot code must be specific to each hardware platform. Following this are: the code for pSOS, together with a default configuration table and I/O jump table (about 6 KB); the code for pROBE (about 40 KB); and the code for pRISM (about 13 KB). Within the boot code for the FSCC, the layout is as follows:

Offset	
0	reset vector (initial SP and PC)
8	boot data (flags, identification text in ASCII)
3C	default boot configuration table
AC	boot code proper (initial PC points here)
2A4	dummy pSOS ROOT process
2A8	pROBE serial port driver
40A	prototype system configuration table
---	prototype pRISM configuration table
---	prototype pRISM initialization table
49A	prototype pROBE configuration table

At boot time (when the board is RESET), the prototype configuration tables are used to construct corresponding tables in RAM, after which pROBE is started up. The application program that is downloaded contains a pSOS configuration table, which is specific to that application; the location of this table is patched into the system configuration table as part of the download. Then, when pSOS is started (by the pROBE 'gs' command), it finds the application and runs it. The dummy pSOS tables and ROOT process allow pSOS to be started without an application downloaded; it just traps to pROBE.

4 System and Module Diagnostics

4.1 Initial Hardware Inspection

The incoming inspection will be as described in "Diagnostics for the FASTBUS Smart Crate Controller".

4.2 Diagnostic Tests

- Memory Test
  - RAM
  - ROM
- Reset Test
  - Front Panel Switch
  - Front Panel Connector
  - Power On
  - FASTBUS
- Serial Port
- Interrupt Controller
- Ethernet Interface
  - Loopback
  - Dual-Port RAM
  - Channel Atten.
- Output Port
  - Output Port Control Registers
  - Header Counter Registers
  - Control Fifo
  - Data Out to Aux. Interrupt.
  - Auxiliary Parallel Port
  - Data Transfer
    - Normal Frame Mode
    - Processor Controlled Mode
  - Permit in/Permit out
- Front Panel Output Port
- Front Panel Trigger Port
- FASTBUS functions.
  - Parallel ports
    - Port 1 Registers
    - Port 2 Registers
    - Check Status & Control bits
  - FPORT Controller Instructions
  - All FASTBUS operations
  - Short and Long Timers
  - FASTBUS Error Handling
  - List Fifo
  - Data Transfers
- Local Word Counter
- Data Fifo
- Processor Fifo
- Real Time Clock
  - Set and Read Clock
  - Set and Read Ethernet Address
- General Hardware exerciser runs all of the Diagnostics

4.3 Diagnostic Operating Instructions

The following sequence is executed to start the diagnostic program. This loads the diagnostic from EPROM into the RAM space of the FSCC. Once in the diagnostic menu most commands are self explanatory. For procedures and detailed descriptions of the tests refer to "Diagnostics for the FASTBUS Smart Crate Controller".

```
pROBE>go 10000

Exception. Level=7  F/O=00A4 (TRAP #9)  Running: **** NO pSOS ****
-------------------------------------------------------------------
SR=2700-tfSm.111...xnzvc USP=00000000 MSP=00200800 ISP=00200800
VBR=00200000 SFC=0 DFC=0 CACR=00 CAAR=00000000
DR=00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000
AR=00000000 00000000 00200E80 002007F8  00002D54 00200DE0 00007F56
PC=00010036-00010036  FFFF            DC.W   $FFFF

pROBE>gs

pSOS Initialized. Running: 'SPRS' -#002262E4
-------------------------------------------------------------------
SR=1000-tfsM.000...xnzvc USP=0023F262 MSP=0023F0D2 ISP=00225C44
VBR=00200000 SFC=0 DFC=0 CACR=00 CAAR=00000000
DR=FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF  FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
AR=FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF  00210AB2 0021E308 00000000
PC=002129CC-002129CC  4EBACF46        JSR    $0020F914(PC)

pROBE>go

      FSCC
Diagnostic Menu    key: -  not available
      ====================          .. not complete
       1) Memory test
       2) Reset test
       3) Serial Port
       4)- Interrupt Controller
       5) Ethernet Interface
       6) Output Port /Data FIFO
       7) Trigger Port
       8) FASTBUS
       9) Local Word Counter
      10) Processor FIFO
      66) Setup Default test Slave  PAD (hex: 12)
      77) Loop on a menu item switch (OFF)
      88) General Hardware Exerciser Test
      99) Exit to pROBE
Enter Command:
```

Appendix A - FPORT Controller Instruction Set

BUS_ARBITRATE	$0062 0300

Description: Arbitrate for FASTBUS using the low byte of the data operand. Bits 0-5 supply the arbitration vector. Bit 7 enables assured access mode. Bit 6 (prioritized access mode) is ignored. Note that the data operand is a long word and is normally identical to the value of CSR 8.

Example Syntax: MOVE.L CSR_8, BUS_ARBITRATE

Operation:

```
C1: FPORT select;
C2: FPORT instruction fetch;
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    {goto INTS;}
    elseif  IGK(FRDY)  return processor acknowledge A; FDSACK*⇓
    else                   {request bus; FREQ⇑
                           goto C4;}
C5: return processor acknowledge B; FDSACK*⇓
C6: Delay Cycle; /*FPORT deselect*/
C7: Delay Cycle;
C8: Delay Cycle;
```

Note: In a multi-master system, the processor should examine the parallel port FSLV*, FRDY and FRAK inputs to confirm that the FSCC has either acquired the bus or been addressed as a slave while attempting to acquire the bus. 0

BUS_RELEASE	$060 0004

Description: Release FASTBUS.

Example Syntax: MOVE.L XX, BUS_RELEASE

Operation:

C1: FPORT select;
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: {DS=0; FCDS⇑
    DK=0; FCDK⇑
    AS=0; FCAS⇑
    release bus;} FREL⇑
C5: Delay Cycle; /*FPORT deselect*/
C6: Delay Cycle;
C7: Delay Cycle;

ADDRESS_DATA_GEOGRAPHICAL $0060 0304

Description: Perform a FASTBUS geographical primary address cycle to DATA Space.

Example Syntax: MOVE.L FB_ADDR, ADDRESS_DATA_GEOGRAPHICAL

Operation:

```
C1: FPORT select;
    latch FB_ADDR; DCPBA⇑
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C4;}
    elseif  AK(FRAK)  {reset short timer; TIMER⇓
                       MS=0;
                       exit;}
    else              {AS=1; FSAS⇑
                       RD=0;
                       MS=0;
                       EG=1; FDEG⇑
                       enable short timer;
                       goto C4;} TIMER⇑
```

ADDRESS_CSR_GEOGRAPHICAL $0060 0308

Description: Perform a FASTBUS geographical primary address cycle to CSR Space.

Example Syntax: MOVE.L FB_ADDR,ADDRESS_CSR_GEOGRAPHICAL

Operation:

```
C1: FPORT select;
    latch FB_ADDR; DCPBA⇑
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C4;}
    elseif  AK(FRAK)  {reset short timer; TIMER⇓
                       MS=0; FDMS0⇑
                       exit;}
    else              {AS=1; FSAS⇑
                       RD=0;
                       MS=1; FDMS0⇑
                       EG=1; FDEG⇑
                       enable short timer; TIMER⇑
                       goto C4;}
```

ADDRESS_DATA_LOGICAL $0060 030C

Description: Perform a FASTBUS logical primary address cycle to DATA Space.

Example Syntax: MOVE.L FB_ADDR,ADDRESS_DATA_LOGICAL

Operation:

```
C1: FPORT select;
    latch FB_ADDR; DCPBA⇑
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C4;}
    elseif  AK(FRAK)  {reset short timer; TIMER⇓
                       MS=0;
                       exit;}
    else              {AS=1; FSAS⇑
                       RD=0;
                       enable short timer; TIMER⇑
                       goto C4;}
```

ADDRESS_CSR_LOGICAL $0060 0310

Description: Perform a FASTBUS logical primary address cycle to CSR Space.

Example Syntax: MOVE.L FB_ADDR,ADDRESS_CSR_LOGICAL

Operation:

```
C1: FPORT select;
    latch FB_ADDR; DCPBA⇑
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C4;}
    elseif  AK(FRAK)  {reset short timer; TIMER⇓
                       MS=0; FDMS0⇓
                       exit;}
    else              {AS=1; FSAS⇑
                       RD=0;
                       MS=1; FDMS0⇑
                       enable short timer; TIMER⇑
                       goto C4;}
```

ADDRESS_DATA_BROADCAST $0060 0314

Description: Perform a FASTBUS broadcast primary address cycle to DATA Space.

Example Syntax: MOVE.L FB_ADDR,ADDRESS_DATA_BROADCAST

Operation:

```
C1: FPORT select;
    latch FB_ADDR; DCPBA⇑
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)     goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                      goto C4;}
    elseif  AK(FRAK)  {reset short timer; TIMER⇓
                      MS=0; FDMS1⇓
                      exit;}
    else              {AS=1; FSAS⇑
                      RD=0;
                      MS=2; FDMS1⇑
                      enable short timer;} TIMER⇑
```

ADDRESS_CSR_BROADCAST $0060 0318

Description: Perform a FASTBUS geographical primary address cycle to CSR Space.

Example Syntax: MOVE.L FB_ADDR,ADDRESS_CSR_BROADCAST

Operation:

```
C1: FPORT select;
    latch FB_ADDR; DCPBA⇑
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C4;}
    elseif  AK(FRAK)  {reset short timer; TIMER⇓
                       MS=0; FDMS0⇓, FDMS1⇓
                       exit;}
    else              {AS=1; FSAS⇑
                       RD=0;
                       MS=3; FDMS0⇑, FDMS1⇑
                       enable short timer; TIMER⇑
                       goto C4;}
```

ADDRESS_RELEASE $0060 031C

Description: Release address lock.

Example Syntax: MOVE.L XX,ADDRESS_RELEASE

Operation:

```
C1: FPORT select;
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                      goto C4;}
    else              {DS=0; FCDS⇑
                      DK=0; FCDK⇑
                      MS=0; FDMS0⇓, FDMS1⇓,FDMS2⇓
                      AS=0; FSAS⇑
                      enable short timer;} TIMER⇑

C5: if      IRQ(SEQINT) goto INTF;
    elseif  AK(FRAK)    goto C5;
    else                exit;
```

DATA_PROCESSOR_RANDOM_READ $0062 0320

Description: Perform a FASTBUS single word read data cycle.

Example Syntax: MOVE.L DATA_PROCESSOR_RANDOM_READ,DATA

Operation:

```
C1: FPORT select;
C2: FPORT instruction fetch;
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTS;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C4;}
    elseif  DK(FRDK)  return processor acknowledge A; FDSACK*⇓
    else              {
                       enable short timer; TIMER⇑
                       RD=1; FDRD⇑
                       MS=0;
                       DS=1; FSDS⇑
                       goto C4;}
C5: reset short timer; TIMER⇓
    return processor acknowledge B; FDSACK*⇓
C6: if      IRQ(SEQINT)    INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C6;}
    elseif  !DK(FRDK*)     reset short timer; TIMER⇓
    else              {enable short timer; TIMER⇑
                       RD=0; FDRD⇓
                       MS=0;
                       DS=0; FCDS⇑
                       goto C6;}
C7: delay cycle; /* processor deselect */
C8: delay cycle;
C9: delay cycle;
```

DATA_PROCESSOR_RANDOM_WRITE $0060 0324

Description: Perform a FASTBUS single word write data cycle.

Example Syntax: MOVE.L DATA,DATA_PROCESSOR_RANDOM_WRITE

Operation:

```
C1: FPORT select;
    latch DATA; DCPBA⇑
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C4;}
    elseif  DK(FRDK)   reset short timer; TIMER⇓
    else               {enable short timer; TIMER⇑
                       RD=0;
                       MS=0;
                       DS=1; FSDS⇑
                       goto C4;}
C5: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C5;}
    elseif  !DK(FRDK*)exit;
    else               {enable short timer; TIMER⇑
                       RD=0;
                       MS=0;
                       DS=0; FCDS⇑
                       goto C5;}
```

DATA_PROCESSOR_SEC_ADDRESS_READ $0062 0328

Description: Perform a FASTBUS secondary address read cycle.

Example Syntax: MOVE.L DATA_PROCESSOR_SEC_ADDRESS_READ,SADDR

Operation:

```
C1: FPORT select;
C2: FPORT instruction fetch;
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTS;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C4;}
    elseif  DK(FRDK)  return processor acknowledge A; FDSACK*⇓
    else              {enable short timer; TIMER⇑
                       RD=1; FDRD⇑
                       MS=2; FDMS1⇑
                       DS=1; FSDS⇑
                       goto C4;}
C5: reset short timer; TIMER⇓
    return processor acknowledge B; FDSACK*⇓
C6: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C6;}
    elseif  !DK(FRDK*)     {reset short timer; TIMER⇓
    else              {enable short timer; TIMER⇑
                       RD=0; FDRD⇓
                       MS=2; FDMS1⇑
                       DS=0; FCDS⇑
                       goto C6;}
C7: delay cycle;       /* processor deselect */
C8: delay cycle;
C9: delay cycle;
```

DATA_PROCESSOR_SEC_ADDRESS_WRITE $0060 032C

Description: Perform a FASTBUS secondary address write cycle.

Example Syntax:

```
MOVE.L
    SADDR,DATA_PROCESSOR_SEC_ADDRESS_WRITE
```

Operation:

```
C1: FPORT select;
    latch SADDR; DCPBA⇑
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)     goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                      goto C4;}
    elseif  DK(FRDK)  reset short timer; TIMER⇓
    else              {enable short timer; TIMER⇑
                      RD=0;
                      MS=2; FDMS1⇑
                      DS=1; FSDS⇑
                      goto C4;}
C5: if      IRQ(SEQINT)     goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                      goto C5;}
    elseif  !DK(FRDK*)      exit;
    else              {enable short timer; TIMER⇑
                      RD=0;
                      MS=0; FDMS1⇓
                      DS=0; FCDS⇑
                      goto C5;}
```

DATA_PROCESSOR_BLOCK_TRANSFER_READ $0062 0008

Description: Perform one step of a FASTBUS block transfer read cycle.

Example Syntax:
```
MOVE.L
DATA_PROCESSOR_BLOCK_TRANSFER_READ,DATA
```

Operation:
```
C1: FPORT select;
C2: FPORT instruction fetch;
C3: instruction dispatch;
C4: if      IRQ(SEQINT)     goto INTS;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                      goto C4;}
    elseif  ! DS(FRDS*)     {enable short timer; TIMER⇑
                      RD=1; FDRD⇑
                      MS=1; FDMS1⇑
                      DS=1; FSDS⇑
                      goto C6;}
    else              {enable short timer; TIMER⇑
                      RD=1; FDRD⇑
                      MS=1; FDMS0⇑
                      DS=0;} FCDS⇑
C5: if      IRQ(SEQINT)     goto INTS;
    elseif  ! DK(FRDK*)     {return processor acknowledge A; FDSACK*⇓
                      reset short timer; TIMER⇓
                      goto C7;}
    else              goto C5;
C6: if      IRQ(SEQINT)     goto INTS;
    elseif  DK(FRDK)  {return processor acknowledge A; FDSACK*⇓
                      reset short timer;} TIMER⇓

    else              goto C6;
C7: return processor acknowledge B; FDSACK*⇓
C8: delay cycle;        /* processor deselect */
C9: delay cycle;
C10: delay cycle;
C11: delay cycle;
```

DATA_PROCESSOR_BLOCK_TRANSFER_WRITE $0060 000C

Description: Perform one step of a FASTBUS block transfer write cycle.

Example Syntax:
```
MOVE.L
DATA,DATA_PROCESSOR_BLOCK_TRANSFER_WRITE
```

Operation:
```
C1: FPORT select;
    latch DATA; DCPBA⇑
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)     goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                      RD=0;
                      MS=1; FDMS0⇑
                      goto C4;}
    elseif  ! DS(FRDS*)     {enable short timer; TIMER⇑
                      RD=0;
                      MS=1; FDMS0⇑
                      DS=1; FSDS⇑
                      goto C6;}
    else              {enable short timer; TIMER⇑
                      RD=0;
                      MS=1; FDMS0⇑
                      DS=0;} FCDS⇑
C5: if      IRQ(SEQINT)     goto INTF;
    elseif  ! DK(FRDK*)     {reset short timer; TIMER⇓
                      RD=0;
                      MS=0; FDMS0⇓
                      exit;}
    else              goto C5;
C6: if      (IRQ)     goto INTF;
    elseif  (DK)      {reset short timer; TIMER⇓
                      RD=0;
                      MS=0; FDMS0⇓
                      exit;}
    else              goto C6;
```

DATA_PROCESSOR_BLOCK_TRANSFER_TERMINATE $0060 0330

Description: Perform termination step of a FASTBUS block transfer.

Example Syntax:

MOVE.Lxx,DATA_PROCESSOR_BLOCK_TRANSFER_TERMINATE

Operation:

C1: FPORT select;
return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if IRQ(SEQINT) goto INTF;
elseif WT(FRWT) goto C4;
else {MS=0; FDMS0⇓, FDMS1⇓, FDMS2⇓
RD=0; FDRD⇓
DS=0;} FCDS⇑
C5: delay cycle;
C6: delay cycle;
C7: delay cycle;

DATA_FIFO_BLOCK_TRANSFER_READ $0060 0334

Description: Perform a FASTBUS block transfer read to the Data FIFO.

Example Syntax: MOVE.L xx,DATA_FIFO_BLOCK_TRANSFER_READ

Operation:

```
C1: FPORT select;
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  {reset short timer; TIMER⇓
                       goto C4;}
    else              {enable FIFO; DFIFOEN⇑,PFIFOEN⇑
                       RD=1; FDRD⇑
                       MS=1; FDMS0⇑
                       DS=1; FSDS⇑
                       local counter mode = decrement; LC0⇑, LC1⇑
                       clock global word counter;} FCLK⇑
C5: {local counter mode = hold; LC0⇓, LC1⇓
    enable short timer;} TIMER⇑
C6: continue;                          /* delay cycle */
C7: if      ((FRDK*FRSS1)#SEQINT #FEOB)
    goto C11;
    elseif  WT(FRWT)  {reset short timer;  LC0⇑, LC1⇑
                       goto C7;}
    elseif  DK(FRDK)  {DS=0; FCDS⇑
                       local counter mode = decrement; LC0⇑, LC1⇑
                       reset short timer; TIMER⇓
                       clock global word counter;} FCLK⇑
    else              {enable short timer; TIMER⇑
                       goto C7;}
C8: {local counter mode = hold; LC0⇓, LC1⇓     /* delay cycle */
    enable short timer;} TIMER⇑
C9: delay cycle;
C10: if     ((!FRDK*FRSS1)#SEQINT#FEOB)
    goto C11;
    elseif  WT(FRWT)  {reset short timer;
                       goto C9;}
```

```
        elseif   !DK(!FRDK)      {DS=1; FSDS⇑
                                  local counter mode=decrement; LC0⇑, LC1⇑
                                  reset short timer; TIMER⇓
                                  clock global word counter; FCLK⇑
                                  goto C5;}
        else                      goto C9;
C11: if         SS1(FRSS1) clock global word counter; FCLK⇑
/* block transfer termination */
C12: if         IRQ(SEQINT)      goto INTF;
     elseif     DS(FRDS)    continue;
     else                     goto C22;
/* termination routine for odd word count transfer */
/* check that DK is high */
/* set DS low and wait for DK low */
/* data written to FIFO on DK down is dummy word with EOE flag*/
C13:if          IRQ(SEQINT)      goto INTF;
     elseif     DK(FRDK)    {reset short timer; TIMER⇓
                                  MS=0;} FDMS0⇓
     else                         {enable short timer; TIMER⇑
                                  goto C13;}
C14:continue;                              /* delay cycle */
C15:continue;                              /* delay cycle */
C16:continue;                              /* delay cycle */
C17:if          IRQ(SEQINT)      goto INTF;
     elseif     WT(FRWT)  {reset short timer; TIMER⇓
                                  goto C17;}
     else                         {enable short timer; TIMER⇑
                                  clock global word counter; FCLK⇑
                                  DS=0;} FCDS⇑
C18:if          IRQ(SEQINT)      goto INTF;
     elseif     ! DK(! FRDK)     {RD=0; FDRD⇓
                                  reset short timer;} TIMER⇓
     else                         {enable short timer; TIMER⇑
                                  goto C18;}
C19:continue;                              /* delay cycle */
C20:continue;                              /* delay cycle */
C21:exit;
/* termination routine for even word count transfer */
C22:if          IRQ(SEQINT)      goto INTF;   /* check that DK is low */
     elseif     ! DK(! FRDK)     {RD=0; FDRD⇓
                                  reset short timer;} TIMER⇓
     else                         {enable short timer; TIMER⇑
                                  goto C20;}
C23:continue;                              /* delay cycle */
C24:continue;                              /* delay cycle */
```

```
C25:continue;                    /* delay cycle */
C25:exit;
```

DATA_FIFO_PIPELINED_READ_100 $0060 0338

Description: Perform a FASTBUS pipelined read to the Data FIFO at 100 nsec/word.

Example Syntax: MOVE.L xx,DATA_FIFO_PIPELINED_READ_100

Operation:

```
C1: FPORT select;
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  goto C4;
    elseif  EOB(FEOB) goto PIPELINE_TERMINATE;
    else            {enable FIFO; PFFEN⇑, DFFEN⇑
                    RD=1; FDRD⇑
                    MS=3; FDMS0⇑,FDMS1⇑
                    DS=1; FSDS⇑
                    local counter mode = decrement; LC0⇑,LC1⇑
                    clock global word counter;} FCLK⇑
C5: local counter mode = hold; LC0⇓,LC1⇓
C6: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  goto C6;
    elseif  EOB(FEOB) goto PIPELINE_TERMINATE;
    else            {DS=0; FCDS⇑
                    local counter mode = decrement; LC0⇑,LC1⇑
                    clock global word counter;} FCLK⇑
C7: {local counter mode = hold; LC0⇓,LC1⇓
    goto C4;}
```

DATA_FIFO_PIPELINED_READ_200 $0060 033C

Description: Perform a FASTBUS pipelined read to the Data FIFO at 200 nsec/word.

Example Syntax: MOVE.L xx,DATA_FIFO_PIPELINED_READ_200

Operation:

```
C1: FPORT select;
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)     goto INTF;
    elseif  WT(FRWT)  goto C4;
    elseif  EOB(FEOB)  goto PIPELINE_TERMINATE;
    else            {enable FIFO; PFFEN⇑, DFFEN⇑
                    RD=1; FDRD⇑
                    MS=3; FDMS0⇑,FDMS1⇑
                    DS=1; FSDS⇑
                    local counter mode = decrement; LC0⇑,LC1⇑
                    clock global word counter;} FCLK⇑
C5: local counter mode = hold; LC0⇓,LC1⇓
C6: continue;                   /* delay cycle */
C7: continue;                   /* delay cycle */
C8: if      IRQ(SEQINT)     goto INTF;
    elseif  WT(FRWT)  goto PIPELINE_TERMINATE;
    elseif  EOB(FEOB)  goto C12;
    else            {DS=0; FCDS⇑
                    local counter mode = decrement; LC0⇑,LC1⇑
                    clock global word counter;} FCLK⇑
C9: local counter mode = hold; LC0⇓,LC1⇓
C10:continue;                   /* delay cycle */
C11:goto C4;                    /* delay cycle */
```

DATA_FIFO_PIPELINED_READ_400 $0060 0340

Description: Perform a FASTBUS pipelined read to the Data FIFO at 400 nsec/word.

Example Syntax: MOVE.L xx,DATA_FIFO_PIPELINED_READ_400

Operation:

```
C1: FPORT select;
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)     goto INTF;
    elseif  WT(FRWT)  goto C4;
    elseif  EOB(FEOB) goto PIPELINE_TERMINATE;
    else            {enable FIFO; PFFEN⇑, DFFEN⇑
                    RD=1; FDRD⇑
                    MS=3; FDMS0⇑,FDMS1⇑
                    DS=1; FSDS⇑
                    local counter mode = decrement; LC0⇑,LC1⇑
                    clock global word counter;} FCLK⇑
C5: local counter mode = hold; LC0⇓,LC1⇓
C6: continue;                   /* delay cycle */
C7: continue;                   /* delay cycle */
C8: continue;                   /* delay cycle */
C9: continue;                   /* delay cycle */
C10:continue;                   /* delay cycle */
C11:continue;                   /* delay cycle */
C12:if      IRQ(SEQINT)     goto INTF;
    elseif  WT(FRWT)  goto C12;
    elseif  EOB(FEOB) goto PIPELINE_TERMINATE;
    else            {DS=0; FCDS⇑
                    local counter mode = decrement; LC0⇑,LC1⇑
                    clock global word counter;} FCLK⇑
C13:local counter mode = hold; LC0⇓,LC1⇓
C14:continue;                   /* delay cycle */
C15:continue;                   /* delay cycle */
C16:continue;                   /* delay cycle */
C17:continue;                   /* delay cycle */
C18:continue;                   /* delay cycle */
C19:goto C4;                    /* delay cycle */
```

NULL	$0060 0020

Description: Access the FPORT Controller without performing any operation. Confirms that the FPORT Controller is active. If the FPORT Controller is stalled, NULL will cause a Processor BUS ERROR interrupt.

Example Syntax: MOVE.L xx,NULL

Operation:

C1: FPORT select;
return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: exit;

Note: If FASTBUS WT is asserted, the FPORT Controller will wait indefinitely for a slave response. A WT timeout will only be generated (after the long timeout period) if the processor attempts another FASTBUS instruction while the first operation is pending. In cases where the processor does not access the FPORT Controller for extended periods of time (e.g., a standalone microcode readout loop) an occasional NULL instruction will detect a FASTBUS lockup condition. NULL will also guarantee that all pending FASTBUS operations have been completed by clearing the instruction pipeline.

LOCAL_COUNTER_LOAD $0062 0010

Description: Initialize Local Word Counter for block transfer.

Example Syntax: MOVE.L COUNT,LOCAL_COUNTER_LOAD

Operation:

C1: FPORT select;
C2: FPORT instruction fetch;
C3: instruction dispatch;
C4: {return processor acknowledge A; FDSACK*⇓
local counter mode = load;} LC1⇑
C5: {return processor acknowledge B; FDSACK*⇓
local counter mode = hold;} LC1⇓
C6: delay cycle; /* processor deselect */
C7: delay cycle;
C8: delay cycle

Note: The counter is loaded from the low order 12 bits of COUNT.

LOCAL_COUNTER_READ $0062 0014

Description: Read current value of Local Word Counter.

Example Syntax: MOVE.L LOCAL_COUNTER_READ,COUNT

Operation:

C1: FPORT select;
C2: FPORT instruction fetch;
C3: instruction dispatch;
C4: {return processor acknowledge A; FDSACK*⇓
local counter mode = read;} SLCOE⇑
C5: return processor acknowledge B; FDSACK*⇓
C6: delay cycle; /* processor deselect */
C7: delay cycle;
C8: delay cycle;

Note: The counter is returned in the low order 12 bits of COUNT.

FIFO_WRITE	$0062 0018

Description: Write a single word from the processor to the Data FIFO.

Example Syntax: MOVE.L DATA,FIFO_WRITE

Operation:

C1: FPORT select;
C2: FPORT instruction fetch;
C3: instruction dispatch;
C4: {return processor acknowledge A; FDSACK*⇓
clock FIFO input;} SDW⇑
C5: {return processor acknowledge B; FDSACK*⇓
clock global word counter;} FCLK⇑
C6: delay cycle; /* processor deselect */
C7: delay cycle;
C8: delay cycle;

EOE	$0060 0024

Description: Write dummy word to the data FIFO with the End-Of-Event bit set.

Example Syntax: MOVE.L XX,EOE

Operation:

C1: FPORT select;
C2: FPORT instruction fetch;
C3: instruction dispatch;
C4: {set EOE flag to output controller; CEOE⇑
set EOE flag to data FIFO;} DEOE⇑
C5: delay cycle;
C6: delay cycle;
C7: delay cycle;

SLAVE_DATA_INPUT	$0062 001C

Description: Transfer one data word from FASTBUS to the processor in slave mode.

Example Syntax: MOVE.L SLAVE_DATA_INPUT,DATA

Operation:

```
C1: FPORT select;
C2: FPORT instruction fetch;
C3: instruction dispatch;
C4: if      IRQ(SEQINT)  goto INTS;
    elseif  WT(FRWT)     goto C4;
    elseif  DS(FRDS)     {return processor acknowledge A; FDSACK*⇓
                         clear WT;} FCWT⇑
    else                 {return processor acknowledge A; FDSACK*⇓
                         clear WT; FCWT⇑
                         goto C6;}
C5: {return processor acknowledge B; FDSACK*⇓
    DK=1; FSDK⇑
    goto C7;}
C6: {return processor acknowledge B; FDSACK*⇓
    DK=0;} FCDK⇑
C7: delay cycle;
C8: delay cycle;
C9: delay cycle;
```

SLAVE_DATA_OUTPUT $0060 0344

Description: Transfer one data word from the processor to FASTBUS in slave mode.

Example Syntax: MOVE.L DATA,SLAVE_DATA_OUTPUT

Operation:

```
C1: FPORT select;
    latch DATA; DCPBA⇑
    return processor acknowledge A; FDSACK*⇓
C2: FPORT instruction fetch;
    return processor acknowledge B; FDSACK*⇓
C3: instruction dispatch;
C4: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)       got0 C4;
    elseif  DS(FRDS)       {clear WT; FCWT⇑
                           DK=1; FSDK⇑
                           exit;}
    else                   {clear WT; FCWT⇑
                           DK=0; FCDK⇑
                           exit;}
```

PIPELINE_TERMINATE Internal Subroutine

Description: Internal routine to terminate pipelined transfers.

Example Syntax:

Operation:

```
C1: if      SS1(FRSS1) clock global work coutner; FCLK⇑
C2: if      IRQ(SEQINT)    goto INTF;
    elseif  DS(FRDS)   continue; TIMER⇑
    else                   goto C8;
/* termination routine for odd word count transfer */
/* check that DK is high */
/* set DS low and wait for DK low */
C3: if      IRQ(SEQINT)    goto INTF;
    else    ! DK(! FRDK)   goto C3; TIMER⇑
C4: continue;                  /* delay cycle */
C5: continue;                  /* delay cycle */
C6: if      IRQ(SEQINT)    goto INTF;
    elseif  WT(FRWT)  goto C6;
    else            {clock global word counter; FCLK⇑
                    DS=0;} FCDS⇑
C7: if      IRQ(SEQINT)    goto INTF;
    elseif  ! DK(! FRDK)   {RD=0; FDRD⇓
                           exit;
    else            goto C7; TIMER⇑
/* termination routine for even word count transfer */
/* check that DK is low */
C8:if       IRQ(SEQINT)    goto INTF;
    elseif  ! DK(! FRDK)   reset short timer; TIMER⇑
    else            goto C8;
C9:continue;
C10:continue;
C11:exit;
```

INTS	Internal Subroutine

Description: Internal routine to abort instruction on error interrupt.
INTS assumes that processor acknowledge has not yet been returned.

Operation:

C1: return processor acknowledge A; FDSACK*⇓

C2: return processor acknowledge B; FDSACK*⇓
goto INTF;

INTF	Internal Subroutine

Description: Internal routine to abort instruction on error interrupt.
INTF assumes that processor acknowledge has already been returned.
Exit the current FASTBUS operation by returning all signals to inactive state.

Operation:

```
C1: {disable tranceivers;
    AS=0; FCAS⇑
    DS=0; FCDS⇑
    DK=0;} FCDK⇑
C2: delay cycle;        /* processor deselect */
C3: exit;
```

Appendix B - FSCC Parts List

Description	Quan	Manf.	Manf. #		
3.6864 MHz Crystal	1	CTS	MP037-3.6864 MHz		
.01uF Non-Polarized Capacitor	3				
.057" FUSE Sockets	20	AUGAT	8134-HC-6P2		
.1uF 25V Non-Polarized Capacitor	5				
.1uF Ceramic,Dip Cap	118				
Coaxial LEMO Connector, K-Loc	3	Kings	1077-3		
10 Pin Sip PAK, 100 Ohms	3	Bourns	4310R-101-101		
10 Pin Sip Socket	2	Samtec	SL-110-G-19		
10 Pin Sip, 100 Ohms, 5 Individual Res.	2	Bourns	4310R-102-101		
10/32 Shoulder Screws	2	R.A.F. Electt.	7039-SS MOD E=1/2		
100pF Non-Polarized Capacitor	1				
10KH ECL PAL	2	TI	PAL1016P8JC		
16 Pin Dip Socket, 300 mils wide	2	Samtec	ICO-316-SGG		
1K Ohm, 10-Pin Sip PAK	2	Bourns	4310R-101-102		
1N914 Diode	1	Motorola	1N914		
20 Pin Dip Socket, 300 mils wide	1	Samtec	ICO-320-SGG		
20.000 MHz Crystal	1	CTS	MP200-20MHz		
20pF Non-Polarized Capacitor	2				
22V10 Reprogramable AND-OR Array	14	Lattice	GAL22V10-15LP		
24 Pin Dip Socket, 300 mils wide	16	Samtec	ICO-324-SGG		
28 Pin Dip Socket, 300 mils wide	7	Preci-Dip	110-99-328-41-001		
28 Pin Dip Socket, 600 mils wide	30	Samtec	ICO-628-SGG		
2K X 16-Bit, Dual-Port RAM (Master)	1	IDT	IDT7133S70G		
2K X 16-Bit, Dual-Port RAM (Slave)	1	IDT	IDT7143S70G		
32-Bit Microprocessor (20 MHz)	1	Motorola	MC68020RC20		
32K X 8-Bit CMOS 70ns Static RAM	8	Sony	CXK58258P-45		
4-Pin LEMO Connector	2	LEMO	RA0.304NYL		
4/40 X 1/4" Bind Head Screw	4				
40 MHz Oscillator	1	CTS	MX055GA-2C-40 Mhz		
40 Pin Dip Socket, 600 mils wide	6	Samtec	ICO-640-SGG		
48 Pin Dip Socket, 600 mils wide	1	Samtec	ICA-648-SGG		
5-Bit Magnitude Comparator	1	Motorola	MC10H166		
5pF Non-Polarized Capacitor	2				
5V-IN,9V-OUT,DC-DC Converter	1	Reliability	2VP5U9		
6.8uF Polarized Capacitor	3				
64K X 8 Bit CMOS EPROM	8	AMD	AM27C512-120		
68 Pin Grid Array	2	Samtec	MVAS-68-ZSGG-11		
75uH, Pulse Transformer	1	Datatronics	PT10312		
9-Bit D-Type Edge-Triggered Flip	1	Signetics	N74F823N		
Dialight Green LED with Integral RES	1	Dialight	558-0202-003		
Dialight Yellow LED with Integral RES	1	Dialight	558-0302-003		
Dual D-Type Master-Slave Flip-Flop	2	Motorola	MC10H131		

Duart	1	Signetics	SCN68681C1N40		
ECL/TTL Octal Transceiver/Translator	9	Brooktree	BT501KC		
Ethernet Serial Interface	1	Intel	P82C501-10MHz		
Ethernet Transceiver Chip	1	National	DP8392A		
FASTBUS Front Panel Mounting Bracket	2	FNAL Drawing	0882-MB-199070		
FASTBUS Module Auxilliary Connector	1	AMP	534974-9		
FASTBUS Module Segment Connector	1	AMP	1-102585-3		
FSCC Front Panel	1				
FSCC P.C. Board	1				
HEX 2-Input or Drivers	1	TI	SN74AS832BN		
Local Area Network Coprocessor	1	Intel	P82586-10MHz		
Male 20-Pin 100M X 100M Dip Header	2	3M	3428-5302		
Multi-Function Peripheral	1	Motorola	MC68901		
NE555 Precision Timer	1	RCA	LM555CN		
Octal Bidirectional Transceiver, 3-state	4	Signetics	N74F545N		
Octal Buffer and Driver, 3-State	3	Signetics	N74F1244		
Octal Bus Transceiver, 3-State	4	Signetics	N74F646N		
Octal D-Type Edge-Triggered Flip-Flops	3	Signetics	N74F825N		
Octal D-Type Flip-Flop w/3-State Outputs	2	Signetics	N74F574N		
Isolated BNC Solder Jack Assembly	1	AMP	227726-1		
Parallel 2048 X 9-Bit FIFO	4	Sharp	LH5498-35		
Parallel 256 X 9-Bit FIFO	10	Sharp	LH5496-35		
Parallel Interface/Timer	2	Motorola	MC68230P10		
Subminiature Fuse	10	Pico	251.005		
Quad 2-Input or Gate	1	Motorola	MC10H103		
Quadruple Differential Line Driver	1	Fairchild	UA96174		
Quadruple Differential Line Receiver	2	Fairchild	UA96175		
Resistor 1.5K 1/8W 5%	4				
Resistor 100K 1/8W 5%	1				
Resistor 10K 1/8W 5%	1				
Resistor 150 1/8W 5%	1				
Resistor 1K 1/8W 5%	16				
Resistor 1K 1/8W 1%	1				
Resistor 1M 1/8W 5%	4				
Resistor 20 1/4W 5%	10				
Resistor 220 1/8W 5%	5				
Resistor 240 1/4W 5%	2				
Resistor 39 1/8W 5%	6				
RS232 Driver/Receiver	1	Maxim	MAX233C		
SPDT Push Button Switch	1	C&K	8121-S-D-A6-G-E		
Stand-Alone Microsequencer	4	Altera	EPS488DC-25		
User-Configurable uP Peripheral	6	Altera	EPB1400DC		
Watchdog Timekeeper, R/T Clock	1	Dallas Semi.	DS1286		
114 Pin Grid Array	1	Samtec	CPAS-114-ZSGG-13A		

Appendix C - PLD Equations

ADD_REC.ABL-8/25/89

```
MODULE ADDRESS_RECOGNITION
TITLE 'ADDRESS_RECOGNITION    VERSION 25-AUG-89 0930 hrs'

ADD_REC  DEVICE  'EC16P8N';


AS              PIN     1;
AD7             PIN     2;
AD6             PIN     3;
AD5             PIN     9;
MS2             PIN     10;
MS1             PIN     11;
AK              PIN     13;
EG              PIN     14;
AI              PIN     15;
IREQ            PIN     16;
IREL            PIN     21;
EAI             PIN     22;
IGK             PIN     23;

D               PIN     7;
CLK             PIN     18;
R               PIN     20;
!AS_DELAY       PIN     4;
!GA_MATCH       PIN     8;
IAR             PIN     17;


EQUATIONS

"       Address Recognition

D               =       !AK & !MS1 & !MS2 & !AD7 & !AD6 & !AD5 & GA_MATCH;

R               =       !AS;

CLK             =       EG & AS_DELAY;

AS_DELAY        =       AS;

"       Arbitration Latch

IAR             =       (IAR & !IREL & !IGK) # (IREQ & (!EAI # !AI));

END ADDRESS_RECOGNITION
```

BERR.ABL-2/15/90

```
module bus_err
title 'bus error generator
      Gustavo Cancelo,  Fermilab,  4-18-89'
"       Changed polarity of LTO 1-SEP-89 Rick Kwarciany
"       Added EXTRES to Reset equations  15-Feb-90  RK

      berr    device    'p22v10';

" bus error controller inputs/outputs
CLK,AS                  pin  1,2;
ADDERR,!LTO             pin  3,5;
BERR                    pin  23;
Q2,Q1,Q0                pin  17,18,19;

" reset generator i/o
!EXTRES                      pin 8;
WDTO                    pin 9;
PUSHB,PWRUP             pin 10,11;
RESETI                  pin 13;
TRIG                    pin 14;
RESET                   pin 16;
RESETN                  pin 15;

L,H,C,X = 0,1,.C.,.X.;

Q2,Q1,Q0 ISTYPE 'reg_d,feed_reg';

"buser  =   [Q2,Q1,Q0];

"IDLE=0;  BER1=1;   BER3=3;  BER4=4;   BER5=5;

equations

Q2 := !ADDERR & !LTO & !AS & !Q2 & !Q1 & !Q0
   #  Q2 & !Q1 & !Q0
   #  Q2 & !Q1 & Q0 & !AS;

Q1 :=  !Q2 & !Q1 & Q0
   #   !Q2 & Q1 & Q0 & LTO;

Q0 :=  !Q2 & !Q1 & !Q0 & !AS & LTO
   #   !Q2 & !Q1 & Q0
   #   !Q2 &  Q1 & Q0 & LTO
   #   Q2 & !Q1 & !Q0
   #   Q2 & !Q1 & Q0 & !AS;

!BERR := !Q2 & !Q1 & !Q0 & !AS & LTO
      #  !Q2 & !Q1 & Q0
      #  !Q2 & !Q1 & !Q0 & !LTO & !AS & !ADDERR
      #  Q2 & !Q1 & !Q0;

!TRIG = PUSHB # !PWRUP # !WDTO;

RESET = RESETI + EXTRES;
!RESETN = RESETI + EXTRES;
```

```
test_vectors

 ([CLK,AS,ADDERR,LTO,Q2,Q1,Q0] -> [Q2,Q1,Q0,BERR])
  [ C, H,    L,     L, L, L, L ] -> [L, L, L,  H  ];
  [ C, H,    X,     X, L, L, L ] -> [L, L, L,  H  ];
  [ C, L,    X,     H, L, L, L ] -> [L, L, H,  L  ];
  [ C, X,    X,     X, L, L, H ] -> [L, H, H,  L  ];
  [ C, X,    X,     H, L, H, H ] -> [L, H, H,  H  ];
  [ C, X,    X,     L, L, H, H ] -> [L, L, L,  H  ];
  [ C, L,    L,     L, L, L, L ] -> [H, L, L,  L  ];
  [ C, X,    X,     X, H, L, L ] -> [H, L, H,  L  ];
  [ C, L,    X,     X, H, L, H ] -> [H, L, H,  H  ];
  [ C, H,    X,     X, H, L, H ] -> [L, L, L,  H  ];

test_vectors

 ([PUSHB,PWRUP,WDTO] -> [TRIG])
  [  L,     H,    H  ] -> [ H  ];
  [  H,     H,    H  ] -> [ L  ];
  [  L,     L,    H  ] -> [ L  ];
  [  L,     H,    L  ] -> [ L  ];

end bus_err_
```

CLOCKS.ABL-11/11/89

```
module clock
title 'FSCC PC2 VERSION 11-NOV-89 0800'
clocks device 'P22V10';

OSC,OSC2        pin   1,2;
CLK1            pin   22;
CLK2A           pin   23;
CLK2B           pin   21;
CLK2C           pin   16;
CLK3            pin   20;
CLK4            pin   17;
CLK5           pin   19;
CLK2N           pin   18;

CLK = [CLK5,CLK4,CLK3,CLK2A];

equations

CLK1 = OSC2;

CLK := CLK - 1;

CLK2B := !CLK2A;

CLK2C := !CLK2A;

CLK2N := CLK2A;

END
```

DECO1.ABL-3/24/90

```
module decoder_1
title 'Fastbus Readout Controller (FSCC), decoder map
Gustavo Cancelo,  Fermilab, 4/3/89'
"Changed ADDERR to allow access to 500000-57FFFF(LPBK,and ETHRES) 2-11-89 RK
"Changed ADDERR to not allow access to 420000-43FFFF(Former LOOPS) 2-11-89 RK
"Changed ADDERR to allow access to 6C0000-6DFFFF now FBUSRS.  24-3-90  RK

          deco1    device    'p22V10';

AS,RW,FC1,FC0                              pin     2,4,5,6;
A22,A21,A20,A19,A18,A17                    pin     7,8,9,10,11,13;
NVDS,LTEN                                  pin     17,15;
ADDERR                                     pin     18;
IACK,TMRS                                  pin     19,14;
ROM2S,ROM1S                                pin     20,21;
RAM2S,RAM1S                                pin     22,23;

L,H,X,Z = 0,1,.X.,.Z.;
ADDRESS = [X,A22,A21,A20,A19,A18,A17,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X];
ROM1LOW =  ^h000000;
ROM1HIGH = ^h03FFFF;
ROM2LOW =  ^h040000;
ROM2HIGH = ^h07FFFF;

RAM1LOW =  ^h200000;
RAM1HIGH = ^h21FFFF;
RAM2LOW =  ^h220000;
RAM2HIGH = ^h23FFFF;

TMRSLOW = ^h480000;
"TMRSHIGH = ^h49FFFF;
NVDLOW = ^h300000;
NVDHIGH = ^h31FFFF;

RES1LOW = ^h080000;
RES1HIGH = ^h0FFFFF;
RES2LOW = ^h100000;
RES2HIGH = ^h17FFFF;
RES3LOW = ^h180000;
RES3HIGH = ^h1FFFFF;

RES4LOW = ^h240000;
RES4HIGH = ^h2FFFFF;

RES5LOW = ^h320000;
RES5HIGH = ^h3FFFFF;

RES6LOW = ^h420000;
RES6HIGH = ^h43FFFF;

RES7LOW = ^h580000;
RES7HIGH = ^h5FFFFF;

RES8LOW = ^h6E0000;
RES8HIGH = ^h6FFFFF;
RES9LOW = ^h700000;
```

```
RES9HIGH = ^hFFFFFF;

ADCODE = [A19,A18,A17];
INTERRUPT = 7;
FCODE = [FC1,FC0];
CPU_SPACE = 3;

equations
!RAM1S=(FCODE != CPU_SPACE)&(ADDRESS >= RAM1LOW)&(ADDRESS <= RAM1HIGH)&!AS;

!RAM2S=(FCODE != CPU_SPACE)&(ADDRESS >= RAM2LOW)&(ADDRESS <= RAM2HIGH)&!AS;

!ROM1S=(FCODE != CPU_SPACE)&(ADDRESS >= ROM1LOW)&(ADDRESS <=
ROM1HIGH)&!AS&RW;

!ROM2S=(FCODE != CPU_SPACE)&(ADDRESS >= ROM2LOW)&(ADDRESS <=
ROM2HIGH)&!AS&RW;

!TMRS =(FCODE != CPU_SPACE)&(ADDRESS == TMRSLOW) & !AS;

!IACK= (FCODE == CPU_SPACE) & (ADCODE == INTERRUPT) & !AS & A22;

"          (ADDRESS >= ROM2LOW)&(ADDRESS <= ROM2HIGH) & !AS
"        # (ADDRESS >= RAM2LOW)&(ADDRESS <= RAM2HIGH) & !AS
!ADDERR = (ADDRESS >= RES1LOW)&(ADDRESS <= RES3HIGH) & !AS
         # (ADDRESS >= RES4LOW)&(ADDRESS <= RES4HIGH) & !AS
         # (ADDRESS >= RES5LOW)&(ADDRESS <= RES5HIGH) & !AS
        # (ADDRESS >= RES6LOW)&(ADDRESS <= RES6HIGH) & !AS
         # (FCODE!= CPU_SPACE)&(ADDRESS >= RES7LOW)&(ADDRESS<= RES7HIGH) & !AS
         # (FCODE!= CPU_SPACE)&(ADDRESS >= RES8LOW)&(ADDRESS<=RES9HIGH) & !AS;

!NVDS = (FCODE != CPU_SPACE)&(ADDRESS >= NVDLOW)&(ADDRESS <= NVDHIGH) & !AS;

LTEN = !AS;

test_vectors
([RW,FC1,FC0,ADDRESS,AS]->[RAM1S,RAM2S,ROM1S,ROM2S,TMRS,IACK,ADDERR,NVDS,LTEN])
 [X, H,  L ,^h000000,H ]->[ H,    H,     H,    H,    H,    H,    H,    H,    L ];
 [L, H,  L ,^h000000,L ]->[ H,    H,     H,    H,    H,    H,    H,    H,    H ];
 [H, H,  L ,^h000000,L ]->[ H,    H,     L,    H,    H,    H,    H,    H,    H ];
 [H, H,  L ,^h01FFFF,L ]->[ H,    H,     L,    H,    H,    H,    H,    H,    H ];
 [H, L,  H ,^h040000,L ]->[ H,    H,     H,    L,    H,    H,    H,    H,    H ];
 [X, L,  H ,^h200400,L ]->[ L,    H,     H,    H,    H,    H,    H,    H,    H ];
 [X, L,  H ,^h220400,L ]->[ H,    L,     H,    H,    H,    H,    H,    H,    H ];
 [X, L,  H ,^h22FFFF,L ]->[ H,    L,     H,    H,    H,    H,    H,    H,    H ];
 [X, H,  L ,^h480000,L ]->[ H,    H,     H,    H,    L,    H,    H,    H,    H ];
 [X, H,  L ,^h080000,L ]->[ H,    H,     H,    H,    H,    H,    L,    H,    H ];
 [X, H,  L ,^h1FFFFF,L ]->[ H,    H,     H,    H,    H,    H,    L,    H,    H ];
 [X, L,  H ,^h300000,L ]->[ H,    H,     H,    H,    H,    H,    H,    L,    H ];
 [X, L,  H ,^h500000,L ]->[ H,    H,     H,    H,    H,    H,    H,    H,    H ];
 [X, L,  H ,^h700000,L ]->[ H,    H,     H,    H,    H,    H,    L,    H,    H ];
 [X, H,  H ,^hFF0000,L ]->[ H,    H,     H,    H,    H,    L,    H,    H,    H ];

END
```

DECO2.ABL-11/1/89

```
module decoder_2 flag '-tl'
title 'Fastbus Readout Controller (FSCC), decoder map
Gustavo Cancelo,  Fermilab, 4/3/89'
"Changed polarity of CAS (positive true)  1-NOV-89  RK
"Added software reset of Ethernet coprocessor  1-NOV-89  RK
"Changed LOOPS to LPBK        1-NOV-89  RK

        deco2    device    'p22V10';

CLK                                          pin     1;
AS,FC1,FC0                                   pin     2,5,6;
A22,A21,A20,A19,A18,A17                      pin     7,8,9,10,11,13;
UARTS                                        pin     19;
CAS,LPBK                                     pin     20,14;
ETHS                                         pin     22;
ETHRES                                    pin     16;
RESET                                     pin     15;
ETHRES1                                          pin
        17;

L,H,X,Z,C = 0,1,.X.,.Z.,.C.;
ADDRESS = [X,A22,A21,A20,A19,A18,A17,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X];

ETHLOW =  ^h400000;
ETHHIGH = ^h41FFFF;

LPBKC = ^h500000;
LPBKS = ^h520000;

ETHRESS = ^h540000;
ETHRESC = ^h560000;

CASLOW = ^h440000;
UARTSLOW = ^h460000;

ADCODE = [A19,A18,A17];
INTERRUPT = 7;
FCODE = [FC1,FC0];
CPU_SPACE = 3;

LPBK    ISTYPE 'REG_D,FEED_REG';
ETHRES1 ISTYPE 'REG_D,FEED_REG';

equations

!ETHS=  (FCODE != CPU_SPACE) & (ADDRESS >= ETHLOW) & (ADDRESS <=
ETHHIGH)&!AS;

CAS = (FCODE != CPU_SPACE) & (ADDRESS == CASLOW) & !AS;

!UARTS = (FCODE != CPU_SPACE) & (ADDRESS == UARTSLOW) & !AS;

ETHRES1 := (ADDRESS == ETHRESS) & !ETHRES1
       #  (ADDRESS != ETHRESC) & ETHRES1;

ETHRES = ETHRES1 # RESET;
```

```
LPBK := (ADDRESS == LPBKC) & !LPBK
     #  (ADDRESS != LPBKS) & LPBK;


END
```

DECO3.ABL-3/24/90

```
module decoder_3
title 'Fastbus Readout Controller (FSCC), decoder map
Gustavo Cancelo,  Fermilab, 4/3/89
FB2S added by RK 4-AUG-89
FBUSRS added at address 6C0000 by RK  24-May-90'

          deco3    device    'p22V10';

CLK                                          pin     1;
AS,DS,RW,FC1,FC0                             pin     2,3,4,5,6;
A22,A21,A20,A19,A18,A17                      pin     7,8,9,10,11,13;
HCSEL,PARLLS,OPORTS,HCSELI                   pin     14,15,16,18;
FIFO1S,FIFO2S                                pin     20,19;
FSSEL,FCSEL,FB1S,FB2S                        pin     21,22,23,17;

L,H,X,Z,C = 0,1,.X.,.Z.,.C.;
ADDRESS = [X,A22,A21,A20,A19,A18,A17,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X];

FB1LOW =  ^h600000;
FB1HIGH = ^h61FFFF;
FB2LOW =  ^h620000;
FB2HIGH = ^h63FFFF;
FCLOW =   ^h640000;
FCHIGH =  ^h65FFFF;
FSLOW =   ^h660000;
FSHIGH =  ^h67FFFF;
FBUSRS =  ^h6C0000;

OPORTLW = ^h4A0000;
"OPORTHG = ^h4BFFFF;

PARLLOW = ^h4C0000;
HCLOW = ^h4E0000;

FIFO1LOW = ^h680000;
FIFO1HG = ^h69FFFF;
FIFO2LOW = ^h6A0000;
FIFO2HG = ^h6BFFFF;

ADCODE = [A19,A18,A17];
INTERRUPT = 7;
FCODE = [FC1,FC0];
CPU_SPACE = 3;

HCSELI istype 'reg_d,feed_reg';
HCSEL istype 'reg_d';

equations

!FB1S= (FCODE != CPU_SPACE) & ((ADDRESS == FB1LOW) # (ADDRESS == FBUSRS)) &
!AS;

!FB2S= (FCODE != CPU_SPACE) & ((ADDRESS == FB2LOW) # (ADDRESS == FBUSRS)) &
!AS;
```

```
!FCSEL= (FCODE != CPU_SPACE) & (ADDRESS >= FCLOW) & (ADDRESS <= FCHIGH) & !AS
& !DS;

!FSSEL= (FCODE != CPU_SPACE) & (ADDRESS >= FSLOW) & (ADDRESS <= FSHIGH) & !AS
& !DS;

!OPORTS= (FCODE != CPU_SPACE) & (ADDRESS == OPORTLW)&!AS;

!PARLLS = (FCODE != CPU_SPACE)&(ADDRESS == PARLLOW) & !AS;

HCSELI := (FCODE != CPU_SPACE)&(ADDRESS == HCLOW) & !AS;

!HCSEL := HCSELI & !AS;

!FIFO1S = (FCODE != CPU_SPACE)&(ADDRESS >= FIFO1LOW)&(ADDRESS <= FIFO1HG) &
!AS & RW;

!FIFO2S = (FCODE != CPU_SPACE)&(ADDRESS >= FIFO2LOW)&(ADDRESS <= FIFO2HG) &
!AS & !RW;

test_vectors

([CLK,RW,DS,FC1,FC0,ADDRESS,AS] ->
[FB1S,FCSEL,FSSEL,FIFO1S,FIFO2S,OPORTS,PARLLS,HCSELI,HCSEL])
 [ C, X, X,  H,  L, ^h600000, H ] -> [ H,     H,    H,    H,     H,     H,
H,    L,      H  ];
 [ C, X, X,  H,  L, ^h600000, L ] -> [ L,     H,    H,    H,     H,     H,
H,    L,      H  ];
 [ C, X, X,  L,  H, ^h640000, L ] -> [ H,     L,    H,    H,     H,     H,
H,    L,      H  ];
 [ C, X, X,  L,  H, ^h660000, L ] -> [ H,     H,    L,    H,     H,     H,
H,    L,      H  ];
 [ C, X, X,  L,  H, ^h4A0000, L ] -> [ H,     H,    H,    H,     H,     L,
H,    L,      H  ];
 [ C, X, X,  L,  H, ^h4C0000, L ] -> [ H,     H,    H,    H,     H,     H,
L,    L,      H  ];
 [ C, H, X,  L,  H, ^h680000, L ] -> [ H,     H,    H,    L,     H,     H,
H,    L,      H  ];
 [ C, L, X,  L,  H, ^h680000, L ] -> [ H,     H,    H,    H,     H,     H,
H,    L,      H  ];
 [ C, H, X,  L,  H, ^h6A0000, L ] -> [ H,     H,    H,    H,     H,     H,
H,    L,      H  ];
 [ C, L, X,  L,  H, ^h6A0000, L ] -> [ H,     H,    H,    H,     L,     H,
H,    L,      H  ];
 [ C, X, X,  L,  H, ^h4E0000, L ] -> [ H,     H,    H,    H,     H,     H,
H,    H,      H  ];
 [ C, X, X,  L,  H, ^h4E0000, L ] -> [ H,     H,    H,    H,     H,     H,
H,    H,      L  ];

END
```

DSGEN.ABL-12/11/89

```
module dsgen
title 'Fastbus Readout Controller (FSCC), data strobe generator
Gustavo Cancelo,  Fermilab, 6/14/89'
"Changed FASTBUS addresses 7-AUG-89, RK
"Changed H&C Select from 0 wait states to 1 wait state. 9-Oct-89 RK.
"Added LPBKS, and ETHRES  1-NOV-89  RK
"Added ROM2, and RAM2, to DSACK1 equation   11-DEC-89   RK.

          dsgen    device    'p22V10';

CLK                                          pin     1;
AS,DS,RW,FC1,FC0                             pin     2,3,4,5,6;
A22,A21,A20,A19,A18,A17                      pin     7,8,9,10,11,13;
BUSYL                                        pin     14;
FBDSACK                                      pin     23;
FDSYNC                                       pin     16;
DSACK0,DSACK1                                pin     18,19;
TMRACK                                       pin     15;
WS0,WS1,WS2                                  pin     20,21,22;

L,H,X,Z,C = 0,1,.X.,.Z.,.C.;
WS0,WS1,WS2 istype 'reg_d,feed_reg';
FDSYNC istype 'reg_d';

ADDRESS = [X,A22,A21,A20,A19,A18,A17,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X,X];
ROM1LOW =  ^h000000;          "32 bits - 1WS
ROM1HIGH = ^h03FFFF;
ROM2LOW =  ^h040000;          "32 bits - 1WS
ROM2HIGH = ^h07FFFF;

RAM1LOW =  ^h200000;          "32 bits - 0WS
RAM1HIGH = ^h21FFFF;
RAM2LOW =  ^h220000;          "32 bits - 0WS
RAM2HIGH = ^h23FFFF;

NVDLOW = ^h300000;            "8bits - 2WS
NVDHIGH = ^h310000;

TMRSLOW = ^h480000;           "8 bits - ASYNCH
"TMRSHIGH = ^h49FFFF;

ETHLOW =  ^h400000;           "32 bits - 1WS
ETHHIGH = ^h41FFFF;

LPBKS = ^h500000;          "16 bits - 0WS
LPBKC = ^h520000;

ETHRESS = ^h540000;         "16 bits - 0WS
ETHRESC = ^h560000;

CASLOW = ^h440000;            "32 bits - 1WS
UARTSLOW = ^h460000;          "8 bits - 3WS

FBLOW =   ^h600000;           "32 bits - 1WS on Writes/0WS on Reads
FBHIGH =  ^h63FFFF;
FCLOW =   ^h640000;           "8 bits - 1WS
```

```
FCHIGH =  ^h65FFFF;
FSLOW =   ^h660000;          "8 bits - 1WS
FSHIGH =  ^h67FFFF;

OPORTLW = ^h4A0000;          "8 bits - 0WS
"OPORTHG = ^h4BFFFF;

PARLLOW = ^h4C0000;          "8 bits - 0WS
HCLOW = ^h4E0000;           "16 bits - 1WS

FIFO1LOW = ^h680000;         "32 bits - 0WS
FIFO1HG = ^h69FFFF;
FIFO2LOW = ^h6A0000;         "32 bits - 0WS
FIFO2HG = ^h6BFFFF;

ADCODE = [A19,A18,A17];
INTERRUPT = 7;
FCODE = [FC1,FC0];
CPU_SPACE = 3;

equations
"          WS0.RE = AS;
"          WS1.RE = AS;
"          WS2.RE = AS;
"          WS3.RE = AS;

         WS0 := !AS;

         WS1 := WS0 & !AS;

         WS2 := WS1 & !AS;

FDSYNC := (!FBDSACK # FDSYNC) * !AS;

!DSACK0 = (ADDRESS >= RAM1LOW) & (ADDRESS <= RAM1HIGH) & !AS
        # (ADDRESS >= ROM1LOW) & (ADDRESS <= ROM1HIGH) & WS1 & !AS
        # (ADDRESS >= RAM2LOW) & (ADDRESS <= RAM2HIGH) & !AS
        # (ADDRESS >= ROM2LOW) & (ADDRESS <= ROM2HIGH) & WS1 & !AS
        # (ADDRESS >= ETHLOW) & (ADDRESS <= ETHHIGH) & !AS & BUSYL & WS1
        # (ADDRESS == CASLOW) & WS1 & !AS
        # (ADDRESS >= FIFO1LOW) & (ADDRESS <= FIFO1HG) & !AS
        # (ADDRESS >= FIFO2LOW) & (ADDRESS <= FIFO2HG) & !AS
        # (ADDRESS == OPORTLW) & !AS
        # (ADDRESS == PARLLOW) & !AS
        # FDSYNC
        # (ADDRESS >= NVDLOW) & (ADDRESS <= NVDHIGH) & !AS & WS2
        # !TMRACK;

!DSACK1 = (ADDRESS >= RAM1LOW)&(ADDRESS <= RAM1HIGH) & !AS
        #  (ADDRESS >= RAM2LOW)&(ADDRESS <= RAM2HIGH) & !AS
        #  (ADDRESS >= ROM1LOW)&(ADDRESS <= ROM1HIGH) & WS1 & !AS
        #  (ADDRESS >= ROM2LOW)&(ADDRESS <= ROM2HIGH) & WS1 & !AS
        #  (ADDRESS >= ETHLOW) & (ADDRESS <= ETHHIGH) & !AS & BUSYL & WS1
        #  (ADDRESS == LPBKS) & !AS
        #  (ADDRESS == LPBKC) & !AS
        #  (ADDRESS == ETHRESS) & !AS
        #  (ADDRESS == ETHRESC) & !AS
```

```
        #   (ADDRESS == CASLOW) & WS1 & !AS
        #   (ADDRESS >= FIFO1LOW)&(ADDRESS <= FIFO1HG) & !AS
        #   (ADDRESS >= FIFO2LOW)&(ADDRESS <= FIFO2HG) & !AS
        #   (ADDRESS == HCLOW) & WS1 & !AS
        # FDSYNC;

test_vectors
([CLK, ADDRESS,AS,BUSYL,TMRACK,FBDSACK]->[WS0,WS1,WS2,DSACK1,DSACK0,FDSYNC])
 [ C, ^h000000, H,  H,     H,       H    ]->[ L,  L,  L,  H,      H,      L   ];
 [ C, ^h000000, L,  H,     H,       H    ]->[ H,  L,  L,  H,      H,      L   ];
 [ C, ^h000000, L,  H,     H,       H    ]->[ H,  H,  L,  L,      L,      L   ];
 [ C, ^h000000, L,  H,     H,       H    ]->[ H,  H,  H,  L,      L,      L   ];
 [ C, ^h000000, L,  H,     H,       H    ]->[ H,  H,  H,  L,      L,      L   ];
 [ C, ^h000000, H,  H,     H,       H    ]->[ L,  L,  L,  H,      H,      L   ];
 [ C, ^h200400, L,  H,     H,       H    ]->[ H,  L,  L,  L,      L,      L   ];
 [ C, ^h200400, H,  H,     H,       H    ]->[ L,  L,  L,  H,      H,      L   ];
 [ C, ^h081400, L,  H,     H,       H    ]->[ H,  L,  L,  H,      H,      L   ];
 [ C, ^h081400, H,  H,     H,       H    ]->[ L,  L,  L,  H,      H,      L   ];

[  C, ^h300000, L,  H,     H,       H    ]->[ H,  L,  L,  H,      H,      L   ];
[  C, ^h300000, L,  H,     H,       H    ]->[ H,  H,  L,  H,      H,      L   ];
[  C, ^h300000, L,  H,     H,       H    ]->[ H,  H,  H,  H,      L,      L   ];
[  C, ^h300000, L,  H,     H,       H    ]->[ H,  H,  H,  H,      L,      L   ];
[  C, ^h300000, H,  H,     H,       H    ]->[ L,  L,  L,  H,      H,      L   ];

"[CLK,ADDRESS,AS,BUSYL,TMRACK,FBDSACK] -> [WS0,WS1,WS2,DSACK1,DSACK0,FDSYNC])
[  C, ^h480000, L,  H,     H,       H    ] -> [ H,  L,  L,  H,      H,      L   ];
[  C, ^h480000, L,  H,     L,       H    ] -> [ H,  H,  L,  H,      L,      L   ];
[  C, ^h480000, H,  H,     H,       H    ] -> [ L,  L,  L,  H,      H,      L   ];
[  C, ^h440000, L,  H,     H,       H    ] -> [ H,  L,  L,  H,      H,      L   ];
[  C, ^h440000, L,  H,     H,       H    ] -> [ H,  H,  L,  L,      L,      L   ];
[  C, ^h440000, L,  H,     H,       H    ] -> [ H,  H,  H,  L,      L,      L   ];
[  C, ^h440000, H,  H,     H,       H    ] -> [ L,  L,  L,  H,      H,      L   ];
[  C, ^h600000, L,  H,     H,       H    ] -> [ H,  L,  L,  H,      H,      L   ];
[  C, ^h600000, L,  H,     H,       H    ] -> [ H,  H,  L,  H,      H,      L   ];
[  C, ^h600000, H,  H,     H,       H    ] -> [ L,  L,  L,  H,      H,      L   ];
[  C, ^h600000, L,  H,     H,       L    ] -> [ H,  L,  L,  L,      L,      H   ];
[  C, ^h600000, H,  H,     H,       H    ] -> [ L,  L,  L,  H,      H,      L   ];
[  C, ^h680000, L,  H,     H,       H    ] -> [ H,  L,  L,  L,      L,      L   ];
[  C, ^h680000, H,  H,     H,       H    ] -> [ L,  L,  L,  H,      H,      L   ];
[  C, ^h4A0000, L,  H,     H,       H    ] -> [ H,  L,  L,  H,      L,      L   ];
[  C, ^h4A0000, H,  H,     H,       H    ] -> [ L,  L,  L,  H,      H,      L   ];

"[CLK, ADDRESS,AS,BUSYL,TMRACK,FBDSACK]->[WS0,WS1,WS2,DSACK1,DSACK0,FDSYNC])
[  C, ^h640000, L,  H,     H,       H    ]->[ H,  L,  L,  H,      H,      L   ];
[  C, ^h640000, L,  H,     H,       H    ]->[ H,  H,  L,  H,      H,      L   ];
[  C, ^h640000, L,  H,     H,       H    ]->[ H,  H,  H,  H,      H,      L   ];
[  C, ^h640000, H,  H,     H,       H    ]->[ L,  L,  L,  H,      H,      L   ];
[  C, ^h660000, L,  H,     H,       H    ]->[ H,  L,  L,  H,      H,      L   ];
[  C, ^h660000, L,  H,     H,       H    ]->[ H,  H,  L,  H,      H,      L   ];
[  C, ^h660000, L,  H,     H,       H    ]->[ H,  H,  H,  H,      H,      L   ];
[  C, ^h660000, H,  H,     H,       H    ]->[ L,  L,  L,  H,      H,      L   ];

END
```

EPB.ADF-11/7/89

```
module epb
Gustavo Cancelo
Fermilab
10/9/89
title mux_demux and timing generator (2)

PART:EPB1400

INPUTS: S1@8,S0@36,Q2@13,Q1@12,Q0@25,AD1@9,CLK@7,CLK2@14

OUTPUTS:
ADT8@26,ADT9@27,ADT10@28,ADT11@29,ADT12@32,ADT13@33,ADT14@34,ADT15@35
         AD8@24,AD9@23,AD10@22,AD11@21,AD12@20,CS2@19,RW1@18,RW2@17
         DB8@4,DB9@3,DB10@2,DB11@1,DB12@40,DB13@39,DB14@38,DB15@37
         OME1@5,OME2@6,S1L@16,S0L@15

NETWORK:

S1=INP(S1)
S0=INP(S0)
Q2=INP(Q2) %Q2,Q1,Q0: epb control lines. refer to epbtemp %
Q1=INP(Q1)
Q0=INP(Q0)
AD1=INP(AD1)
CLK=INP(CLK)
CLK2=INP(CLK2)

RW1=CONF(RW1c,)  %R/!W to dual port memory%
RW2=CONF(RW2c,)  %R/!W to dual port memory%
OME1=CONF(OME1c,)     %!OE to dual port memory%
OME2=CONF(OME2c,)     %!OE to dual port memory%
CS2=CONF(CS2c,)       %!CS to dual port memory #2 %
S0L,S0L=RORF(S0Ld,CLK2,,,) %latchs 82586 S0 line%
S1L,S1L=RORF(S1Ld,CLK2,,,) %latchs 82586 S1 line%

%high add/data byte tranceiver w/tri_state enable%
ADT8,ADT9,ADT10,ADT11,ADT12,ADT13,ADT14,ADT15 = BUSX (IBUS,,OE)

%high address byte input register w/tri_state enable%
OR10,OR11,OR12,OR13,OR14,OR15,OR16,OR17 = RBUSI (IBUS,,WE1)

%high data byte input register w/tri_state enable 2 %
OR20,OR21,OR22,OR23,OR24,OR25,OR26,OR27 = RBUSI (IBUS,,WE2)

%high data byte output register 2 %
IBUS = LBUSO (DB8,DB9,DB10,DB11,DB12,DB13,DB14,DB15,,OLE,RE)

%high address byte external output w/tri_state enable%
AD8 =  CONF (AD8c,)
AD9 =  CONF (AD9c,)
AD10 = CONF (AD10c,)
AD11 = CONF (AD11c,)
AD12 = CONF (AD12c,)

%high data byte external output w/tri_state enable%
DB8,DB8   = COIF (DB8c,HAB2)
```

```
DB9,DB9   = COIF (DB9c,HAB2)
DB10,DB10 = COIF (DB10c,HAB2)
DB11,DB11 = COIF (DB11c,HAB2)
DB12,DB12 = COIF (DB12c,HAB2)
DB13,DB13 = COIF (DB13c,HAB2)
DB14,DB14 = COIF (DB14c,HAB2)
DB15,DB15 = COIF (DB15c,HAB2)


EQUATIONS:

S1Ld = !Q2 * S1
     +  Q2 * S1L;

S0Ld = !Q2 * S0
     +  Q2 * S0L;

OE = S0L * !S1L * (Q2 * !Q1 * Q0
                 + Q2 * Q1 * !Q0
                 + Q2 * Q1 * Q0);  %tranceiver enable%

WE1 = !(!S1 * S0 * !Q2 * !Q1 * Q0
    + S1 * !S0 * !Q2 * !Q1 * Q0);   %latchs add. at rising edge%

WE2 = !(S1 * !S0 * !Q2 * Q1 * Q0);              %latchs data at rising edge%


HAB2 = S1L * !S0L;   %tri_state data out. enable%

OLE = !S1L * S0L * Q2 * !Q1 * Q0;   %latchs data when high%

RE = !S1L * S0L * (Q2 * !Q1 * Q0
                 + Q2 * Q1 * !Q0
                 + Q2 * Q1 * Q0); %out. enable in read cycles%


%dual port RAM selection%

CS2c = !(!S1L * S0L * AD1 *  Q2 * !Q1 * !Q0
        +!S1L * S0L * AD1 *  Q2 * !Q1 * Q0
        + S1L * !S0L * AD1 * Q2 * !Q1 * !Q0
        + S1L * !S0L * AD1 * Q2 * !Q1 * Q0);

%dual port RAM read/write cycle%
RW1c = !(S1L * !S0L * !AD1 * Q2 * !Q1 * !Q0
       + S1L * !S0L * !AD1 * Q2 * !Q1 * Q0);

RW2c = !(S1L * !S0L * AD1 * Q2 * !Q1 * !Q0
       + S1L * !S0L * AD1 * Q2 * !Q1 * Q0);

%dual port RAM output enable%
OME1c = !(!S1L * S0L * !AD1 * Q2 * !Q1 * !Q0
        + !S1L * S0L * !AD1 * Q2 * !Q1 * Q0);

OME2c = !(!S1L * S0L * AD1 * Q2 * !Q1 * !Q0
        + !S1L * S0L * AD1 * Q2 * !Q1 * Q0);
```

```
AD8c = OR10;
AD9c = OR11;
AD10c = OR12;
AD11c = OR13;
AD12c = OR14;

DB8c = OR20;
DB9c = OR21;
DB10c = OR22;
DB11c = OR23;
DB12c = OR24;
DB13c = OR25;
DB14c = OR26;
DB15c = OR27;


END$_
```

EPBTEMP.ADF-7/27/90

```
module prueba
Gustavo Cancelo
Fermilab
7/19/89
title mux_demux and timing generator
%Modified, 7-27-90 to correct state machine operation when busy%
%is true.  Added BUSY wait loop to state 13. RK%

PART:EPB1400

INPUTS: CLK@7,BUSY@13,S1L@12,S0L@25,S1@9,S0@8,CLK2@14,HOLD@36

OUTPUTS: Q0@5,Q1@6,Q2@15,Q3@16
         ADT0@26,ADT1@27,ADT2@28,ADT3@29,ADT4@32,ADT5@33,ADT6@34,ADT7@35
         CS1@24,AD1@23,AD2@22,AD3@21,AD4@20,AD5@19,AD6@18,AD7@17
         DB0@4,DB1@3,DB2@2,DB3@1,DB4@40,DB5@39,DB6@38,DB7@37

NETWORK:

CLK=INP(CLK)
CLK2=INP(CLK2)
BUSY=INP(BUSY) %from dual port RAM BUSY contention line%
S1=INP(S1)     %S1,S0: read/write cycle indicators%
S0=INP(S0)
S0L=INP(S0L)
S1L=INP(S1L)
HOLD=INP(HOLD)

Q3,Q3 = TOTF(Q3t,CLK2,,,)
Q2,Q2 = TOTF(Q2t,CLK2,,,)   %Q2,Q1,Q0: internal counter for timing generator%
Q1,Q1 = TOTF(Q1t,CLK,,,)
Q0,Q0 = TOTF(Q0t,CLK,,,)

%82586 add/data bus interface%
ADT0,ADT1,ADT2,ADT3,ADT4,ADT5,ADT6,ADT7 = BUSX (IBUS,,OE)

%low address byte input latch. RBUSI_A(1)%
OR10,OR11,OR12,OR13,OR14,OR15,OR16,OR17 = RBUSI (IBUS,,WE1)

%low data byte input latch. RBUSI_A(2)%
OR20,OR21,OR22,OR23,OR24,OR25,OR26,OR27 = RBUSI (IBUS,,WE2)

%low data byte output latch. LBUSO(2)%
IBUS = LBUSO (DB0,DB1,DB2,DB3,DB4,DB5,DB6,DB7,,OLE,RE1)

% este registro no se usa%
%IBUS = LBUSO (AD0,AD1,AD2,AD3,AD4,AD5,AD6,AD7,,OLE2,RE2)%

%low address byte at external pins w/tri_state habilitation line%
AD1,AD1 = COIF (AD1c,)
AD2,AD2 = COIF (AD2c,)
AD3,AD3 = COIF (AD3c,)
AD4,AD4 = COIF (AD4c,)
AD5,AD5 = COIF (AD5c,)
AD6,AD6 = COIF (AD6c,)
AD7,AD7 = COIF (AD7c,)
```

```
%low data byte at external pins w/tri_state habilitation line%
DB0,DB0 = COIF (DB0c,HAB2)
DB1,DB1 = COIF (DB1c,HAB2)
DB2,DB2 = COIF (DB2c,HAB2)
DB3,DB3 = COIF (DB3c,HAB2)
DB4,DB4 = COIF (DB4c,HAB2)
DB5,DB5 = COIF (DB5c,HAB2)
DB6,DB6 = COIF (DB6c,HAB2)
DB7,DB7 = COIF (DB7c,HAB2)

CS1 = CONF(CS1c,)

EQUATIONS:
%Read mode: wait_state insertion if BUSY active & count == 13 %
%Write mode: wait_state insertion if BUSY active & count == 13 %

          Q0t = !(!HOLD * !Q3 * !Q2 * !Q1 * !Q0
               +   HOLD * !Q3 * !Q2 * !Q1 * !Q0  %S1 Y S0 DON'T CARE%
               +   HOLD *  Q3 * !Q2 * !Q1 * !Q0 * S1 * S0
               +  !HOLD *  Q3 * !Q2 * !Q1 * !Q0
               +  !HOLD *  Q3 * !Q2 *  Q1 * !Q0
               +   HOLD *  Q3 * !Q2 *  Q1 * !Q0 * S1 * S0
              +  !BUSY *  Q3 *  Q2 * !Q1 *  Q0);
        %added busy to state 13%

          Q1t =  Q3 * !Q2 *  !Q1 * Q0 * !S0 * S1 * HOLD
               + Q3 * !Q2 *  !Q1 * Q0 * !S1 * S0 * HOLD
               + Q3 * !Q2 *  Q1 * !Q0 * !HOLD
               + Q3 * !Q2 *  Q1 * !Q0 * HOLD * S1 * S0
               + Q3 * !Q2 *  Q1 *  Q0 %* BUSY%
               + Q3 *  Q2 * !Q1 *  Q0 * BUSY
               + Q3 *  Q2 *  Q1 *  Q0;

          Q2t = BUSY * Q3 * !Q2 * Q1 * Q0
               + Q0 * Q1 * Q2 * Q3;

          Q3t =   HOLD * !Q3 * !Q2 * !Q1 * !Q0 *  S1 *  S0
               + !HOLD *  Q3 * !Q2 * !Q1 * !Q0
               + !HOLD *  Q3 * !Q2 * !Q1 *  Q0
               +  HOLD *  Q3 * !Q2 * !Q1 *  Q0 *  S1 *  S0
               +  HOLD *  Q3 * !Q2 * !Q1 *  Q0 * !S1 * !S0
               + !HOLD *  Q3 * !Q2 *  Q1 * !Q0
               +  HOLD *  Q3 * !Q2 *  Q1 * !Q0 *  S1 *  S0
               + !HOLD *  Q3 *  Q2 *  Q1 *  Q0;

OE = S0L * !S1L * (Q2 * !Q1 * Q0 + Q2 * Q1 * !Q0
                + Q2 * Q1 * Q0);  %transceiver abilitation%

WE1 = !(!S1 * S0 * !Q2 * !Q1 * Q0
    + S1 * !S0 * !Q2 * !Q1 * Q0);   %latchs add. at rising edge%

WE2 = !(S1 * !S0 * !Q2 * Q1 * Q0);            %latchs data at rising edge%


HAB2 = S1L * !S0L;  %tri_state data out. enable%
```

```
OLE = !S1L * S0L * Q2 * !Q1 * Q0;   %latchs data when high%

RE1 = !S1L * S0L * (Q2 * !Q1 * Q0
                  + Q2 * Q1 * !Q0
                  + Q2 * Q1 * Q0); %out. enable in read cycles%

RE2 = !S1 * S1;
OLE2 = !S1 * S1;

CS1c= !(!S1L * S0L * !AD1 * (Q2 * !Q1 * !Q0        % address 1 %
                           + Q2 * !Q1 * Q0)
       + S1L * !S0L * !AD1 * (Q2 * !Q1 * !Q0
                           + Q2 * !Q1 * Q0));


AD1c = OR11;
AD2c = OR12;
AD3c = OR13;
AD4c = OR14;
AD5c = OR15;
AD6c = OR16;
AD7c = OR17;

DB0c = OR20;
DB1c = OR21;
DB2c = OR22;
DB3c = OR23;
DB4c = OR24;
DB5c = OR25;
DB6c = OR26;
DB7c = OR27;

END$
```

ERR_DETE.ABL-11/1/89

```
MODULE ERROR_DETECT
TITLE 'ERROR_DETECT        VERSION Wednesday 1-NOV-89 0900 hrs'
ERR_DETECT DEVICE 'P22V10';

CLK     PIN     1;
FRDK    PIN     2;
FRAK    PIN     3;
FRSS0   PIN     4;
FRSS1   PIN     5;
FRSS2   PIN     6;
!FSLV   PIN     7;
FEOBA   PIN     8;
!LTC0   PIN     9;
!LTC1   PIN     10;
LCEN    PIN     11;
FCLERR  PIN     13;
CCLERR  PIN     16;

SSERR   PIN     14;
LTC0Z   PIN     15;
FEOB    PIN     17;
FLSS2   PIN     18;
FLSS1   PIN     19;
FLSS0   PIN     20;
ADDR    PIN     21;
FRAKZ   PIN     22;
FRDKZ   PIN     23;

LATCHED_SS      =       [FLSS2..FLSS0];
SS              =       [FRSS2..FRSS0];

ZERO            =       [0,0,0];
TWO             =       [0,1,0];

EQUATIONS

FRAKZ           :=      FRAK;
FRDKZ           :=      FRDK;
LTC0Z           :=      LTC0;

"LATCH THE CURRENT VALUE OF SS IF AK UP OR DK UP/DOWN WITH NON-ZERO SS,
"OTHERWISE KEEP LAST VALUE OF LATCHED SS

"LATCHED_SS HAS LAST NON-ZERO SS RESPONSE

"LATCHED_SS IS ZERO AFTER CLEAR

LATCHED_SS      :=      ((((FRAK & !FRAKZ) # (FRDK != FRDKZ))
                                & (SS != ZERO) & SS)
                        # ((((!(FRAK & !FRAKZ) & (FRDK == FRDKZ))
                                # (SS == ZERO)) & LATCHED_SS))
                        & !FCLERR & !CCLERR & !FSLV;

"ADDR INDICATES WHETHER VALUE OF LATCHED_SS OCCURRED ON ADDRESS
"OR DATA CYCLE
```

```
ADDR            :=      ((FRAK & !FRAKZ) & (SS != ZERO)
                      # ((FRDK == FRDKZ) # (SS == ZERO)) & ADDR)
                      & !FCLERR & !CCLERR & !FSLV;

"SSERR IS SET FOR LATCHED_SS NOT EQUAL ZERO ON ADDRESS CYCLE
"OR LATCHED_SS NOT EQUAL ZERO/TWO ON DATA CYCLE

SSERR           :=      ((LATCHED_SS != ZERO) & ADDR
                      #  (LATCHED_SS != ZERO) & (LATCHED_SS != TWO) & !ADDR);

"FEOB IS SET IF COUNTER REACHES ZERO AND COUNTER IS ENABLED OR
"IF SS=2 ON DATA CYCLE

"IT IS RESET BY SEQUENCER FEOB ACKNOWLEDGE


FEOB            :=      ((LTC0 & !LTC0Z) & LTC1 & LCEN
                      # (FRDK != FRDKZ) & (SS == TWO)
                      # FEOB & !FEOBA)
                      & !FCLERR & !CCLERR & !FSLV;


END ERROR_DETECT
```

FB_ARB.ABL-8/25/89

```
MODULE FASTBUS_ARBITRATION
TITLE 'FASTBUS_ARBITRATION    VERSION 23-AUG-89 1000 hrs'
FB_ARB DEVICE 'EC16P8N';


OAL5            PIN     18;
OAL4            PIN     20;
OAL3            PIN     4;
OAL2            PIN     8;
OAL1            PIN     17;
OAL0            PIN     21;

IAL5            PIN     3;
IAL4            PIN     9;
IAL3            PIN     10;
IAL2            PIN     11;
IAL1            PIN     13;
IAL0            PIN     14;

FAL5            PIN     1;
FAL4            PIN     2;

ISMINE          PIN     5;
GATEALS                 PIN     15;


EQUATIONS

!OAL5   =       !IAL5 # !GATEALS;

!OAL4   =       !IAL4 # (!IAL5 & FAL5) # !GATEALS;

!OAL3   =       !IAL3 # (!IAL4 & FAL4) # (!IAL5 & FAL5) # !GATEALS;

!OAL2   =       !IAL2 # (!IAL3 & OAL3) # (!IAL4 & FAL4) # (!IAL5 & FAL5)
                # !GATEALS;

!OAL1   =       !IAL1 # (!IAL2 & OAL2) # (!IAL3 & OAL3) # (!IAL4 & FAL4)
                # (!IAL5 & FAL5) # !GATEALS;

!OAL0   =       !IAL0 # (!IAL1 & OAL1) # (!IAL2 & OAL2) # (!IAL3 & OAL3)
                # (!IAL4 & FAL4) # (!IAL5 & FAL5) # !GATEALS;

!ISMINE                 =       (!IAL0 & OAL0) # (!IAL1 & OAL1) # (!IAL2 &
OAL2)
                # (!IAL3 & OAL3) # (!IAL4 & FAL4) # (!IAL5 * FAL5)
                # !GATEALS;

END FASTBUS_ARBITRATION
```

FB_REQ.ABL-7/25/90

```
MODULE FB_REQUEST
TITLE 'FB_REQUEST        VERSION 25-JUL-90 1500 hrs'
FB_REQ DEVICE 'P22V10';
"INPUTS
CLK     PIN     1;
!FRESET         PIN     2;
SSERR   PIN     3;
!BUSYR  PIN     4;
!STO    PIN     5;
!DFF    PIN     6;
FRSR    PIN     7;
!FSLV   PIN     8;
FCWT    PIN     9;
FRDS    PIN     10;
CCLERR  PIN     11;
FCLERR  PIN     13;
"OUTPUTS
FRDSZ   PIN     14;
FDWT    PIN     15;
!SBUSYR         PIN     17;     "synchronous BUSYR
ECLK6M  PIN     18;
EC0     PIN     19;     "CLK13M
EC1     PIN     20;
!SEQINT         PIN     21;
!FBERR  PIN     22;
!FBREQ  PIN     23;
ECLK    =       [EC1..EC0];
ZERO    =       [0,0];
TWO     =       [1,0];
EQUATIONS
"Ethernet clock generator
"       divide 40MHz by 3 to get 13.333MHz
"       then divide 13.3 by 2 to get 6.667MHz
WHEN (ECLK == ZERO) THEN ECLK := TWO;
        ELSE ECLK := ECLK - 1;
WHEN (ECLK == TWO) THEN ECLK6M := !ECLK6M;
        ELSE ECLK6M := ECLK6M;
"Synchrnize BUSYR* for Ethernet interface to ensure proper setup, and hold
"       time for the EPB1400's.
WHEN (ECLK == TWO) THEN SBUSYR := BUSYR;
        ELSE SBUSYR := SBUSYR;
"Interrupt equations:
"FBREQ = We're a slave, or We've received SR, or RB
"FBERR = Any SS error except SS=2, or Short Timout, or Data FIFO full.
"SEQINT = FBERR or RB.
FBERR           =       (SSERR # STO # DFF # FBERR)
                        & !FCLERR
                        & !CCLERR;
FBREQ           =       FSLV # FRSR # FRESET;
SEQINT          =       FBERR # FRESET;
"WT generation:
"FASTBUS WT is set high whenever the fscc is a slave, and there is a DS
"transition.  WT is cleared by FCWT from the FASTBUS sequencer.
FRDSZ           :=      FRDS;
FDWT            =       (FSLV * (FRDS != FRDSZ)) # (FDWT * !FCWT);
END FB_REQUEST
```

FBSEQV2.ASM-9/24/90

```
    Richard Kwarciany
    Fermilab
    Saturday, 24-Mar-90  1500hrs
    FSCC Version: 2
    Revision: 2
    EPS448
    U106,U110,U108

    PART:   EPS448,EPS448,EPS448

    INPUTS:         I0@12
            I1@11
            I2@10
            I3@9
            I4@5
            I5@4
            I6@3
            I7@2
    OUTPUTS:LC1@1           %       1       %
            LC0@28          %       2       %
            LCOE@27                 %       3       %
            /SRT@26                 %       4       %
            FW@25           %       5       %
            /POE@24                 %       6       %
            DDIR@23                 %       7       %
            CPAB@22                 %       8       %
            SAB@20          %       9       %
            CPBA@19                 %       10      %
            SBA@18          %       11      %
            /DG@17          %       12      %
            /FDOE@16        %       13      %
            /FCOE@15        %       14      %
            FPCREQ@14       %       15      %
            DFIFOEN@13      %       16      %
    OUTPUTS:REL@1           %       17      %
            REQ@28          %       18      %
            FEOBA@27        %       19      %
            EG@25           %       20      %
            RD@24           %       21      %
            MS2@23          %       22      %
            MS1@22          %       23      %
            MS0@20          %       24      %
            CWT@19          %       25      %
            CDK@18          %       26      %
            SDK@17          %       27      %
            CDS@16          %       28      %
            SDS@15          %       29      %
            CAS@14          %       30      %
            SAS@13          %       31      %
    OUTPUTS:PFIFOEN@1       %       32      %
            /SR@28          %       33      %
            SCLK@27                 %       34      %
            FCLK@26                 %       35      %
            DEOE@25                 %       36      %
            CEOE@24                 %       37      %
            TMR@23          %       38      %
```

```
        CCLEAR@22       %       39      %
        STAT3@20        %       40      %
        STAT2@19        %       41      %
        STAT1@18        %       42      %
        STAT0@17        %       43      %
        /DSACK@16       %       44      %
        /MUX2@15        %       45      %
        /MUX1@14        %       46      %
        /MUX0@13        %       47      %

%DEFAULT = all bits false except: DDIR=1, FCOE*=0, MUX1*=0%
%12345678901234567890123456789012345678901234567%
DEFAULT:[0001011000011010000000000000000010000000001101B]

%       Macros for Chip 1       %

MACROS:         CHOLD           ="000"          %LC1,LC0,LCOE%
        CLOAD           ="100"
        COUNT           ="110"
        CREAD           ="001"
        CDEF            ="000"
        SRT             ="0"            %SRT*%
        NSRT            ="1"
        DW              ="1"            %SDW%
        NDW             ="0"
        DATADEF                 ="1000001"
        %POE*,DDIR,CPAB,SAB,CPBA,SBA,DG*%
        DATAOUT                 ="0000000"
        DATAOUTLATCH    ="0000100"
        DATAOUTHOLD     ="1000010"
        DATAINPROC      ="0100000"
        DATAINLATCH     ="0110000"
        DATAINHOLD      ="0101000"
        DATAINFIFO      ="1101000"
        COUNTIN                 ="0000001"
        FDOE            ="0"            %FDOE*%
        NFDOE           ="1"
        FCOE            ="0"            %FCOE*%
        NFCOE           ="1"
        IRQ             ="0"            %FPCREQ*%
        NIRQ            ="1"
        DFFEN           ="1"            %DFIFOEN%
        DFFDIS          ="0"
        CTRLDEF                 ="0001011000011010"
        %DEFAULTS FOR U42X40%

        Macros for Chip 2       %

        REQ             ="010000000000000"      %REL,REQ%
        REL             ="101000000101010"      %(EOEA, Clear DS,DK,AS,GK)%
        HOLDB           ="00"
        NREQ            ="00"
        EOBA            ="1"            %EOBA%
        NEOBA           ="0"
        EG              ="1"            %EG%
        NEG             ="0"
        RD              ="1"            %RD%
```

```
NRD             ="0"
MS0             ="000"          %MS2,MS1,MS0%
MS1             ="001"
MS2             ="010"
MS3             ="011"
MS4             ="100"
MS5             ="101"
MS6             ="110"
MS7             ="111"
CWT             ="1"            %CWT%
XWT             ="0"
SDK             ="01"           %CDK,SDK%
CDK             ="10"
XDK             ="00"
SDS             ="01"           %CDS,SDS%
CDS             ="10"
XDS             ="00"
SAS             ="01"           %CAS,SAS%
CAS             ="10"
XAS             ="00"
FBDEF           ="000000000000000"      %DEFAULTS FOR U47X40%

Macros for Chip 3

PFFEN           ="1"            %PFIFOEN%
PFFDIS          ="0"
SR              ="0"            %SR*%
NSR             ="1"
SCLK            ="1"            %SCLK%
NSCLK           ="0"
FCLK            ="1"            %FCLK%
NFCLK           ="0"
DEOE            ="1"            %DEOE%
NDEOE           ="0"
CEOE            ="1"            %CEOE%
NCEOE           ="0"
TMR             ="1"            %TMR%
NTMR            ="0"
CCLR            ="1"            %CCLEAR%
NCCLR           ="0"
STAT0           ="0000"                 %STATE3,STATE2,STATE1,STATE0%
STAT1           ="0001"
STAT2           ="0010"
STAT3           ="0011"
STAT4           ="0100"
STAT5           ="0101"
STAT6           ="0110"
STAT7           ="0111"
STAT8           ="1000"
STAT9           ="1001"
STAT10          ="1010"
STAT11          ="1011"
STAT12          ="1100"
STAT13          ="1101"
STAT14          ="1110"
STAT15          ="1111"
DSAK            ="0"            %DSACK*%
```

```
        NDSAK           ="1"
        MUX2            ="011"          %MUX2,MUX1,MUX0%
        MUX1            ="101"
        MUX0            ="110"
%       RESETO = All bits false except: FREL FCWT FCDK FCDS FCAS MUX1%
%                         12345678901234567890123456789012345678901234567%
        RESETO          ="00010110000111101000000011010100100000000001101"

%-------------------------------------------------------------------------%
%Here the 448's set all outputs to their default state, then             %
%wait for either the 68020 to call, or somebody on FASTBUS               %
%to want something.  When FBSEL is true, that means the 68020            %
%is calling, and we need to get the vector from MUX2 and go to           %
%the routine at that address.  If, on the other hand, FSLV is            %
%true, we go to the slave handler.                                       %
%-------------------------------------------------------------------------%

PROGRAM:

RESET:  [RESETO] JUMP SELECT;

%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%
%       Routine entry points.                                             %
%       Entry points are the first instruction of each routine.          %
%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%

%-------------------------------------------------------------------------%
%       Routine ARBITRATE                                                 %
%       --SLOW-- --WRITE--                                                %
%-------------------------------------------------------------------------%
ARB:    IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF I2       %FRDY%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT1 DSAK MUX1]
                JUMP ARBX;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD NFDOE FCOE NIRQ DFFDIS
                REQ
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT1 NDSAK
MUX0]
                JUMP ARB;

%-------------------------------------------------------------------------%
%       Routine BUS_RELEASE                                               %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%                                                                         %
%       This routine sets FREL true, clears DS, DK, and AS, then goes     %
%       home.  Note that since no data is needed, the control lines       %
%       latching the 68020 data       are returned to their default states,  %
```

```
%       discarding the data.                                                %
%-------------------------------------------------------------------------%

BUSREL:                 [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT2 NDSAK
MUX1]
                JUMP BUSRELX;

%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%
%       Primary Address Cycles                                              %
%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%

%-------------------------------------------------------------------------%
%       Routine ADDRESS_DATA_GEOGRAPHICAL                                   %
%                                                                           %
%       Geographical primary address cycle to data space.                   %
%       (we assert EG)                                                      %
%       --FAST-- --WRITE--                                                  %
%-------------------------------------------------------------------------%
%                                                                           %
%       IF FBREQ Release bus, TMR=0, JUMP WAYHOME.                          %
%       ELSEIF WT=1, loop here.                                             %
%       ELSEIF AK=1 THEN turn off FASTBUS AD line drivers, negate EG,       %
%               MS, and go home.                                            %
%       ELSE set AS, TMR (turn on short timer), and loop here.              %
%                                                                           %
%-------------------------------------------------------------------------%
ADDDG:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA EG NRD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDDG;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA EG NRD MS0 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDDG;

%-------------------------------------------------------------------------%
%       Routine ADDRESS_CSR_GEOGRAPHICAL                                    %
%                                                                           %
%       Geographical primary address cycle to CSR space.                    %
%       (We assert EG)                                                      %
```

```
%       --FAST-- --WRITE--                                                    %
%-------------------------------------------------------------------------%
%                                                                           %
%       IF FBREQ THEN Release bus, JUMP WAYHOME.                            %
%       ELSEIF WT=1, loop here.                                             %
%       ELSEIF AK=1 THEN turn off FASTBUS AD line drivers, negate EG,       %
%              MS, and go home.                                             %
%       ELSE set AS, TMR (turn on short timer), and loop here.              %
%                                                                           %
%-------------------------------------------------------------------------%
ADDCG:  IF /I6          %FBREQ%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA EG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDCG;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA EG NRD MS1 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDCG;

%-------------------------------------------------------------------------%
%       Routine ADDRESS_DATA_LOGICAL                                        %
%                                                                           %
%       This routine is identical to ADDRESS_DATA_GEOGRAPHICAL              %
%       except that EG is not driven.                                       %
%       --FAST-- --WRITE--                                                  %
%-------------------------------------------------------------------------%
%                                                                           %
%       IF FBREQ THEN Release bus, TMR=0, JUMP WAYHOME.                     %
%       ELSEIF WT=1, loop here.                                             %
%       ELSEIF AK=1 THEN turn off FASTBUS AD line drivers, negate MS,       %
%              and go home.                                                 %
%       ELSE set AS, TMR (turn on short timer), and loop here.              %
%                                                                           %
%-------------------------------------------------------------------------%
ADDDL:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS0 XWT XDK XDS XAS
```

```
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDDL;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS0 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDDL;

%-------------------------------------------------------------------------%
%       Routine ADDRESS_CSR_LOGICAL                                       %
%-------------------------------------------------------------------------%
%       This routine is identical to ADDRESS_CSR_GEOGRAPHICAL except      %
%       that EG is not driven.                                            %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%                                                                         %
%       IF FBREQ THEN Release bus, TMR=0, JUMP WAYHOME.                   %
%       ELSEIF WT=1, loop here.                                           %
%       ELSEIF AK=1 THEN turn off FASTBUS AD line drivers, negate MS,     %
%       and go home.                                                      %
%       ELSE set AS, TMR (turn on short timer), and loop here.            %
%                                                                         %
%-------------------------------------------------------------------------%
ADDCL:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX0]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDCL;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDCL;

%-------------------------------------------------------------------------%
%       Routine ADDRESS_DATA_BROADCAST                                    %
%                                                                         %
%       This routine does a broadcast primary address cycle to            %
%       DATA space.                                                       %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
```

```
%       IF FBREQ THEN Release bus, TMR=0, jump WAYHOME.                  %
%       ELSEIF WT=1, then loop here.                                     %
%       ELSEIF AK=1, Then turn off FASTBUS AD line drivers, negate MS,   %
%              and go home.                                              %
%       ELSE set AS, TMR (turn on short timer), and loop here.           %
%                                                                        %
%------------------------------------------------------------------------%
ADDDB:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDDB;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDDB;

%------------------------------------------------------------------------%
%       Routine ADDRESS_CSR_BROADCAST                                    %
%                                                                        %
%       This routine does a broadcast primary address cycle to           %
%       CSR space.                                                       %
%       --FAST-- --WRITE--                                               %
%------------------------------------------------------------------------%
%       IF FBREQ THEN Release bus, TMR=0, JUMP WAYHOME.                  %
%       ELSEIF WT=1, loop here.                                          %
%       ELSEIF AK=1 THEN turn off FASTBUS AD line drivers, negate MS,    %
%              and go home.                                              %
%       ELSE set AS, TMR (turn on short timer), and loop here.           %
%                                                                        %
%------------------------------------------------------------------------%
ADDCB:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDCB;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
```

```
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDCB;

%-------------------------------------------------------------------------%
%       Routine ADDRESS_RELEASE                                           %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%               EG=0 RD=0 MS=0 WT=0 Clear DK,DS,AS                        %
ADDREL:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT4 NDSAK
MUX1]
                JUMP ADDREL;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT CDK CDS CAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT4 NDSAK
MUX0]
                JUMP ADDRELX;

%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%
%       Data Cycle Routines                                               %
%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_RANDOM_READ                                %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
DATPRR:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPRR;
        ELSEIF I4               %FRDK%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 DSAK MUX0]
```

```
                JUMP DATPRRX;
        ELSE    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS0 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DATPRR;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_RANDOM_WRITE                               %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
DATPRW:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPRW;
        ELSEIF I4               %FRDK%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPRWX;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS0 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DATPRW;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_SEC_ADDRESS_READ                           %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
DATPSR:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK
MUX0]
                JUMP DATPSR;
        ELSEIF I4               %FRDK%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 DSAK MUX0]
                JUMP DATPSRX;
        ELSE    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS2 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT6 NDSAK MUX0]
```

```
                JUMP DATPSR;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_SEC_ADDRESS_WRITE                          %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
DATPSW:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK
MUX0]
                JUMP DATPSW;
        ELSEIF I4               %FRDK%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK
MUX0]
                JUMP DATPSWX;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT6 NDSAK MUX0]
                JUMP DATPSW;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_BLOCK_TRANSFER_READ                        %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
DATPBR:                 [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPBRX;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_BLOCK_TRANSFER_WRITE                       %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
DATPBW:                 [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPBWX;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_BLOCK_TRANSFER_TERMINATE                   %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%       This routine does a graceful termination of a block transfer      %
%       where DS and DK are brought to their idle states without          %
%       effecting the slave.                                              %
%               MS=0 RD=0                                                 %
```

```
%               then
%               DS=0
DATPBT:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX1]
                JUMP DATPBT;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT XDK CDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Routine LOCAL_COUNTER_LOAD                                        %
%       --SLOW-- --WRITE--                                                %
%-------------------------------------------------------------------------%
LC_LOAD:        [CLOAD NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT9 DSAK MUX1]
                JUMP LC_LOADX;

%-------------------------------------------------------------------------%
%       Routine LOCAL_COUNTER_READ                                        %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
LC_READ:        [CREAD NSRT NDW COUNTIN NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT9 DSAK MUX1]
                JUMP LC_READX;

%-------------------------------------------------------------------------%
%       Routine FIFO_WRITE                                                %
%       --SLOW-- --WRITE--                                                %
%-------------------------------------------------------------------------%
FIFO_WR:        [CDEF NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT10 DSAK MUX1]
                JUMP FIFO_WRX;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_BLOCK_TRANSFER_READ                             %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
DATFBR:         IF /I6                  %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
```

```
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP DATFBR;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK SDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP DATFBRX;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_PIPELINED_READ_1  (100ns transfers)             %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
DATFP1:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP1;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP1X;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_PIPELINED_READ_2  (200ns transfers)             %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
DATFP2:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP2;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP2X;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_PIPELINED_READ_4  (400ns transfers)             %
%       --FAST-- --READ or WRITE--                                        %
%-------------------------------------------------------------------------%
DATFP4:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
```

```
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP4;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP4X;

%-------------------------------------------------------------------------%
%       SLAVE_DATA_INPUT                                                  %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
SLVDIN:                 [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 CWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 NDSAK
MUX0]
                JUMP SLVDINX;

%-------------------------------------------------------------------------%
%       SLAVE_DATA_OUTPUT                                                 %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%       (WT has been cleared back at LETSGO).                             %
%       IF FBREQ* THEN release bus, jump WAYHOME.                         %
%       ELSEIF WT=1 THEN wait here.                                       %
%       ELSEIF DS=1 THEN set DK jump HOME.                                %
%       ELSE clear DK jump HOME.                                          %

SLVDOUT:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP WAYHOME;
        ELSEIF I7       %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 NDSAK
MUX0]
                JUMP SLVDOUT;
        ELSEIF I1       %FRDS%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 XWT SDK XDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 NDSAK
MUX1]
                JUMP WAYHOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 XWT CDK XDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 NDSAK
MUX1]
                JUMP WAYHOME;
```

```
%-------------------------------------------------------------------------%
%       Null                                                              %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
NULL:           [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT12 NDSAK
MUX1]
                JUMP NULLX;

%-------------------------------------------------------------------------%
%       End Of Event                                                      %
%       --SLOW-- --WRITE --                                               %
%-------------------------------------------------------------------------%
EOE:            [CHOLD NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR NSCLK NFCLK DEOE CEOE NTMR NCCLR STAT14 DSAK MUX1]
                JUMP EOEX;

%-------------------------------------------------------------------------%
%       Bulb Test                                                         %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
BULBTST:        [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA EG RD MS7 XWT SDK SDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT11 NDSAK
MUX1]
                JUMP BULBTST;

%-------------------------------------------------------------------------%
%       Sequencer Select                                                  %
%       Wait here to be selected by the 68020                             %
%               FB1S* + FB2S means that the FASTBUS standard sequencer    %
%                       has been selected.                                %
%               FB1S* * FB2S* means that the USER sequencer has been      %
%                       selected.                                         %
%                                                                         %
%       Whichever sequencer has not been selected, puts it's outputs      %
%               into high Z, and is not accessable unless it is reset.    %
%-------------------------------------------------------------------------%
SELECT:         IF /I1 * /I5            %FB1S* * FB2S*%
        THEN [Z] CONTINUE;
        ELSEIF (/I1 * I5) + (I1 * /I5)          %FB2S* + FB1S*%
        THEN [RESETO] JUMP HOME;
        ELSE [RESETO] JUMP SELECT;

TURNOFF:[Z] JUMP TURNOFF;
%-------------------------------------------------------------------------%
%       If, in the cycle executed just previous to returning here,        %
%       DSACK was asserted, then the select line (FBxS*)                  %
%       will still be asserted (it takes the 68020 and it's PAL's a       %
%       certain amount of time to negate a select line after it sees      %
%       DSACK).  HOME would then see FBxS* true and think that it is      %
%       being selected again, when, in fact, FBxS* is just lingering      %
%       from the previous cycle.  As you might imagine, if this were      %
%       to happen, the sequencer would jump to what ever address          %
%       just happened to be on the 68020's address bus at that time       %
```

```
%       and anarchy would reign.  So, to keep order, if we              %
%       want to come HOME right after DSACK, then we instead return     %
%       here where three NOP type instructions are executed.  This      %
%       allows the 68020 enough time to negate FBxS* from any           %
%       preveously executed cycle.                                      %
%-----------------------------------------------------------------------%
WAYHOME:        [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX1]
                CONTINUE;

                [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX1]
                CONTINUE;

                [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX1]
                CONTINUE;

%       Since the IOD lines could be latched on the next cycle (if there%
%       is a pending instruction), the Processor transceivers must be    %
%       set to DATAOUT before coming HOME.  Any instructions leading to %
%       HOME must have the processor transceivers set in this way or it %
%       will be         bad.                                            %
%       When HOME, all outputs are default
HOME:   IF /I1          %FB1S* (Fast cycle, return DSACK now.)
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 DSAK MUX2]
                JUMP LETSGOF;
        ELSEIF /I5      %FB2S* (Slow cycle, return DSACK in routine.%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX2]
                JUMP LETSGOS;
        ELSE    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX1]
                JUMP HOME;

%-----------------------------------------------------------------------%
%       These are the fast and slow jumps to the appropriate FASTBUS    %
%       primative routines as determined by the vector received from    %
%       the '020 at MUX2.                                               %
%-----------------------------------------------------------------------%


LETSGOF:        [CDEF NSRT NDW DATAOUTLATCH NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 DSAK MUX2]
```

```
                PUSHI;

%       Clear WT here is for Slave write routine, dosn't effect anything
else.%
                [CDEF NSRT NDW DATAOUTHOLD NFDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 CWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX0]
                RETURN;

LETSGOS:        [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX2]
                PUSHI;

                [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX0]
                RETURN;

%-------------------------------------------------------------------------%
%       Routine ARBITRATE                                                 %
%       --SLOW-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%                                                                         %
%       Starting with something simple, this routine checks to make       %
%       sure that FBREQ is false, then sets FREQ true, waits for          %
%       FRDY (means that we have the bus), then returns DSACK and goes    %
%       home.                                                             %
%-------------------------------------------------------------------------%

ARBX:           [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT1 DSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Bus Release                                                       %
%-------------------------------------------------------------------------%
%       Release Bus, Status=2                                             %
%                                                                         %
%       Status=2, Status clock = 1                                        %
BUSRELX:        [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT2 NDSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Routine ADDRESS_RELEASE                                           %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%               Wait for AK to go low, then go HOME.                      %
ADDRELX:IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
```

```
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I5               %FRAK%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT CDK CDS CAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT4 NDSAK MUX0]
                JUMP ADDRELX;
        ELSE    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT4 NDSAK MUX1]
                JUMP HOME;

%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%
%       Data Cycle Routines                                               %
%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_RANDOM_READ                                %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, JUMP TO INTS.           %
%       ELSEIF WT=1 THEN Enable FASTBUS transceivers and drivers          %
%               (FDOE*=0,DDIR=1,CPBA=0, DSAB=0,DSBA=0,DG*=0), enable      %
%               processor transceivers (POE*=0), Assert RD=1, MS=0 and    %
%               loop here.                                                %
%       ELSEIF DK=1 THEN return DSACK, and continue.                      %
%       ELSE Enable FASTBUS AD line drivers, enable proccessor xceivers   %
%               Assert RD=1, MS=0, DS(u), TMR, and loop here.             %
%                                                                         %
%       Hold DSACK true for one more cycle, and continue.                 %
%                                                                         %
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, NDSACK, JUMP TO         %
%               WAYHOME.                                                  %
%       ELSEIF WT=1 THEN negate DSACK, disable processor and FASTBUS      %
%               transceivers, continue to assert RD=1, MS=0, and DS=1,    %
%               and loop here.                                            %
%       ELSEIF DK=0 THEN negate DSACK, disable processor and FASTBUS      %
%               transceivers, deassert TMR, RD, and MS, and go WAYHOME.   %
%       ELSE negate DSACK, disable processor and FASTBUS transceivers,    %
%               deassert RD, MS, and DS, assert TMR, and loop here.       %
%                                                                         %
%-------------------------------------------------------------------------%
DATPRRX:        [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 DSAK MUX0]
                CONTINUE;

WAIT1:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
```

```
        THEN    [CTRLDEF
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP WAIT1;
        ELSEIF /I4              %/FRDK%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP WAYHOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP WAIT1;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_RANDOM_WRITE                               %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, JUMP TO WAYHOME.        %
%       ELSEIF WT=1 THEN Enable FASTBUS transceivers and drivers          %
%               (FDOE*=0, DDIR=0, CPBA=0, DSAB=0, DSBA=1, DG*=0),         %
%               Assert RD=0, MS=0 and loop here.                          %
%       ELSEIF DK=1 THEN disable FASTBUS transceivers and drivers, and    %
%               continue.                                                 %
%       ELSE Assert DS(u), TMR, keep FASTBUS lines asserted, and loop     %
%               here.                                                     %
%       be bop a lula she's my baby                                       %
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, JUMP TO HOME.           %
%       ELSEIF WT=1 THEN continue to assert DS=1, and loop here.          %
%       ELSEIF DK=0 THEN go HOME.                                         %
%       ELSE deassert DS, assert TMR, and loop here.                      %
%-------------------------------------------------------------------------%
DATPRWX:IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPRWX;
        ELSEIF /I4              %/FRDK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DATPRWX;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_SEC_ADDRESS_READ                           %
```

```
%       --SLOW-- --READ--                                                    %
%-------------------------------------------------------------------------%
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, JUMP TO INTS.              %
%       ELSEIF WT=1 THEN Enable FASTBUS transceivers and drivers             %
%               (FDOE*=0, DDIR=1,CPBA=0, DSAB=0,DSBA=0,DG*=0), enable        %
%               processor transceivers (POE*=0), Assert RD=1, MS=2 and       %
%               loop here.                                                   %
%       ELSEIF DK=1 THEN return DSACK, and continue.                         %
%       ELSE Assert DS(u), TMR, and loop here.                               %
%                                                                            %
%       Hold DSACK true for one more cycle, and continue.                    %
%                                                                            %
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, DEassert DSACK, jump       %
%               to WAYHOME.                                                  %
%       ELSEIF WT=1 THEN negate DSACK, disable processor and FASTBUS         %
%               transceivers, continue to assert RD=1, MS=2, and DS=1,       %
%               and loop here.                                               %
%       ELSEIF DK=0 THEN negate DSACK, disable processor and FASTBUS         %
%               transceivers, deassert RD, and MS, and go WAYHOME.           %
%       ELSE negate DSACK, disable processor and FASTBUS transceivers,       %
%               deassert RD, MS, and DS, assert TMR, and loop here.          %
%                                                                            %
%-------------------------------------------------------------------------%
DATPSRX:        [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 DSAK MUX0]
                CONTINUE;

WAIT2:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG RD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK
MUX0]
                JUMP WAIT2;
        ELSEIF /I4              %/FRDK%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK MUX1]
                JUMP WAYHOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS2 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT6 NDSAK MUX0]
                JUMP WAIT2;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_SEC_ADDRESS_WRITE                             %
%       --FAST-- --WRITE--                                                   %
%-------------------------------------------------------------------------%
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, jump to WAYHOME.           %
%       ELSEIF WT=1 THEN Enable FASTBUS transceivers and drivers             %
%               (FDOE*=0, DDIR=0, CPBA=0, DSAB=0, DSBA=1, DG*=0),            %
```

```
%               Assert RD=0, MS=2 and loop here.                        %
%       ELSEIF DK=1 THEN disable FASTBUS transceivers and drivers, and  %
%               continue.                                               %
%       ELSE Assert DS(u), keep FASTBUS lines asserted, assert TMR,     %
%               and loop here.                                          %
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, jump to HOME.         %
%       ELSEIF WT=1 THEN continue to assert DS=1, and loop here.        %
%       ELSEIF DK=0 THEN go HOME.                                       %
%       ELSE deassert DS, assert TMR, and loop here.                    %
%-----------------------------------------------------------------------%
DATPSWX:IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK
MUX0].
                JUMP DATPSWX;
        ELSEIF /I4              %/FRDK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT6 NDSAK MUX0]
                JUMP DATPSWX;

%-----------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_BLOCK_TRANSFER_READ                      %
%       --SLOW-- --READ--                                               %
%-----------------------------------------------------------------------%
%       This routine is going to be a little bit different than any of  %
%               the others, in that, DSACK will have to be returned     %
%               after each DK(t) to allow the 68020 to read the data.   %
%       Here goes:                                                      %
%                                                                       %
%       Turn on FASTBUS AD line receivers, Assert RD=1, MS=1,switch     %
%               to MUX1 and continue.                                   %
%       IF FBREQ THEN Release bus, JUMP TO INTS.                        %
%       ELSEIF WT=1 THEN Assert RD=1, MS=1, on FASTBUS  Turn on FASTBUS %
%               xceivers and enable processor xceivers                  %
%               (POE*=0, DDIR=1, CPBA=0, DSAB=0, DSBA=0, DG*=0, FDOE*=0)%
%               and loop here.                                          %
%       ELSEIF DS=0 THEN Assert RD=1, MS=1, DS(u), enable FASTBUS and   %
%               processor transceivers (POE*=0, DDIR=1, CPBA=0,         %
%               DSAB=0, DSBA=0, DG*=0, FDOE*=0)and assert TMR, and JUMP %
%               to DKUPR.                                               %
%       ELSE Assert RD=1, MS=1, DS(d), enable FASTBUS and processor     %
%               transceivers (POE*=0, DDIR=1, CPBA=0, DSAB=0,           %
%               DSBA=0, DG*=0, FDOE*=0) assert TMR, and JUMP to DKDWNR. %
%DKDWNR                                                                 %
%       IF FBREQ THEN RD=1, MS=1, Clear DS, jump to INTS.               %
```

```
%       ELSEIF DK=0 THEN Assert DSACK, continue to drive FASTBUS lines, %
%              and processor and FASTBUS transceivers, deassert TMR,    %
%              and JUMP to next1.                                       %
%       ELSE Continue to drive FASTBUS lines, and transceivers, and     %
%              TMR, and loop here.                                      %
%DKUPR                                                                  %
%       IF FBREQ THEN RD=1, MS=1, Clear DS, assert DSACK, jump to       %
%              NEXT2.                                                   %
%       ELSEIF DK=1 THEN Assert DSACK, continue to drive FASTBUS lines, %
%              and processor and FASTBUS transceivers, deassert TMR,    %
%              and JUMP to next1.                                       %
%       ELSE Continue to drive FASTBUS lines, and transceivers,         %
%              assert TMR, and loop here.                               %
%NEXT1                                                                  %
%       Continue to drive FASTBUS lines, DSACK, and transceivers, and   %
%              Continue.                                                %
%       Continue to drive FASTBUS lines, and transceivers negate DSACK  %
%              and JUMP WAYHOME.                                        %
%                                                                       %
%NEXT2  Clear FASTBUS lines, release bus, keep DSACK asserted, jump      %
%              to WAYHOME.                                              %
%-----------------------------------------------------------------------%
DATPBRX:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPBRX;
        ELSEIF /I1              %/FRDS%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKUPR;
        ELSE    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKDWNR;

DKDWNR:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG RD MS1 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF /I4              %/FRDK%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 DSAK MUX1]
                JUMP NEXT1;
        ELSE    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
```

```
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKDWNR;

DKUPR:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG RD MS1 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 DSAK
MUX1]
                JUMP NEXT2;
        ELSEIF I4               %FRDK%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 DSAK MUX1]
                JUMP NEXT1;
        ELSE    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKUPR;

NEXT1:          [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 DSAK MUX1]
                CONTINUE;

                [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP WAYHOME;

NEXT2:          [CTRLDEF
                REL
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 DSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_BLOCK_TRANSFER_WRITE                       %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%       As with DATPBR, DSACK will have to be returned after each DK(t)   %
%               to allow the 68020 to output new data.                    %
%                                                                         %
%       Keep FASTBUS AD line drivers enabled, assert MS=1, set MUX1,      %
%               and continue.                                             %
%       IF FBREQ THEN Release bus, JUMP TO WAYHOME.                       %
%       ELSEIF WT=1 THEN Assert MS=1, keep FASTBUS transceivers enabled,  %
%               and enable FASTBUS drivers (POE*=1, DDIR=0,               %
%               CPBA=0, DSAB=0, DSBA=1, DG*=0, FDOE*=0), and loop here.   %
%       ELSEIF DS=0 THEN Assert MS=1, DS(u), TMR, keep FASTBUS            %
%               transceivers enabled and enable FASTBUS drivers (POE*=1,  %
%               DDIR=0, CPBA=0, DSAB=0, DSBA=1, DG*=0, FDOE*=0), and      %
%               JUMP to DKUPW.                                            %
%       ELSE Assert MS=1, DS(d), TMR, keep FASTBUS transceivers enabled,  %
%               and enable FASTBUS drivers (POE*=1, DDIR=0,               %
%               CPBA=0, DSAB=0, DSBA=1, DG*=0, FDOE*=0), and JUMP to      %
%               DKDWNW.                                                   %
%DKDWNW                                                                   %
%       IF FBREQ THEN Release bus, JUMP TO HOME.                          %
```

```
%       ELSEIF DK=0 THEN deassert MS, TMR, disable FASTBUS transceivers %
%              and drivers (POE*=0, DDIR=0, CPBA=0, DSAB=0, DSBA=0,     %
%              DG*=1, FDOE*=1), and JUMP home.                          %
%       ELSE continue to assert MS, TMR, and keep FASTBUS transceivers  %
%              and drivers enabled, and loop here.                      %
%DKUPW                                                                  %
%       IF FBREQ THEN Release bus, JUMP TO HOME.                        %
%       ELSEIF DK=1 THEN deassert MS, TMR, disable FASTBUS transceivers %
%              and drivers (POE*=0, DDIR=0, CPBA=0, DSAB=0, DSBA=0,     %
%              DG*=1, FDOE*=1), and JUMP home.                          %
%       ELSE continue to assert MS, TMR, and keep FASTBUS transceivers  %
%              and drivers enabled, and loop here.                      %
%-----------------------------------------------------------------------%
DATPBWX:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPBWX;
        ELSEIF /I1              %/FRDS%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKUPW;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKDWNW;

DKDWNW:         IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF /I4              %/FRDK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKDWNW;

DKUPW:  IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
```

```
        ELSEIF I4               %FRDK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKUPW;

%-------------------------------------------------------------------------%
%       Routine LOCAL_COUNTER_LOAD                                        %
%       --SLOW-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%       This routine loads the Local Word Counter with data from          %
%       the processor.  Only the lower 12 bits of the 32 bit word         %
%       are used.                                                         %
%                                                                         %
%       Set POE* true, put Local Word Counter into load mode,             %
%               assert DSACK and continue.                                %
%       Negate POE*, put Local Word Counter into hold, keep DSACK         %
%               asserted, and continue.                                   %
%       Negate DSACK (hold DSACK false for two cycles to prevent          %
%               sequencer from seeing FBSEL from this cycle when          %
%               returning home).                                          %
%                                                                         %
LC_LOADX:       [CHOLD NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT9 DSAK MUX1]
                CONTINUE;

                [CHOLD NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT9 NDSAK MUX1]
                JUMP WAYHOME;


%-------------------------------------------------------------------------%
%       Routine LOCAL_COUNTER_READ                                        %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
%       Routine to allow the processor to read the value of the Local     %
%               Word Counter.                                             %
%                                                                         %
%       Set POE* true, put Local Word Counter into Read mode, assert      %
%               DSACK, and continue.                                      %
%       Keep all outputs the same for one more cycle.                     %
%       Negate DSACK (hold DSACK false for two cycles to prevent          %
%               sequencer from seeing FBSEL from this cycle when          %
%               returning home).                                          %
%                                                                         %
LC_READX:       [CREAD NSRT NDW COUNTIN NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT9 DSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
```

```
%       Routine FIFO_WRITE                                                  %
%       --SLOW-- --WRITE--                                                  %
%-------------------------------------------------------------------------%
%       Routine to write a word from the 68020 into the processor           %
%       and data FIFO's.                                                    %
%                                                                           %
%       Enable processor transceivers (POE*=0), set FIFO Write true         %
%               (DW=1), enable FIFO's, set DSACK true.                      %
%       Set FCLK true.                                                      %
%       Negate DSACK, DW, FCLK, and POE*, JUMP TO WAYHOME.                  %
%                                                                           %
FIFO_WRX:       [CDEF NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFEN NSR SCLK FCLK NDEOE NCEOE NTMR NCCLR STAT10 DSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_BLOCK_TRANSFER_READ                               %
%       --FAST-- --WRITE or READ--                                          %
%-------------------------------------------------------------------------%
%       This routine does a FASTBUS handshake block transfer read from      %
%       the FASTBUS crate segment to the FSCC's Output Port Data FIFO,      %
%       and also the Processor Data FIFO.                                   %
%-------------------------------------------------------------------------%
%       Note: DDIR, and SAB must be held high throughout the block          %
%               transfer operation.                                         %
%                                                                           %
%DATFBR:IF FBREQ                                                           %
%       THEN release bus jump to INTF                                       %
%       ELSEIF WT=1                                                         %
%       THEN Enable FASTBUS xceivers, enable processor and Data FIFO's      %
%               MS=1, RD=1, Place local word counter into hold, and         %
%               loop here.                                                  %
%       ELSE Turn on FASTBUS xcievers, enable processor and Data FIFO's     %
%               MS=1, RD=1, Place local word counter into count, set        %
%               DS=1, NO FCLK!, and jump to DATFBRX.                        %
%                                                                           %
%BXFERLP:IF (/FRDK * FRSS1) + EOB + FBREQ                                  %
%       THEN Enable FASTBUS xceivers, enable processor and Data FIFO's      %
%               MS=1, RD=1, Place local word counter into hold mode,        %
%               and JUMP to EOBIRQ.                                         %
%       ELSEIF WT=1                                                         %
%       THEN Enable FASTBUS xceivers, enable processor and Data FIFO's      %
%               MS=1, RD=1, Place local word counter into hold, and         %
%               loop here.                                                  %
%       ELSEIF DK=0                                                         %
%       THEN Turn on FASTBUS xcievers, enable processor and Data FIFO's     %
%               MS=1, RD=1, Place local word counter into count, set        %
%               DS=1, set FCLK, and continue.                               %
%       ELSE Place local word counter into hold, and loop here.             %
%                                                                           %
%DATFBRX:Place counter into hold mode, negate FCLK, set TMR, and            %
%               continue.  This        cycle ensures that the FSCC's minimum %
%               FASTBUS cycle time (100ns) is not violated.  If some        %
%               slave returns DK too quickly, the FSCC could                %
%               try to do another DS before the FIFO's were ready for       %
%               the next word.  This cycle keeps the sequencer from         %
```

```
%               looking for DK before it should.                        %
%                                                                       %
%       Keep TMR true, and do another NOP for the same reason as above. %
%                                                                       %
%       IF (FRDK* FRSS1) + EOB + FBREQ                                  %
%       THEN counter=hold, NTMR, keep FIFO's enabled and jump to EOBIRQ.%
%       ELSEIF WT=1                                                     %
%       THEN counter=hold, NTMR, keep both FIFO's enabled and loop here.%
%       ELSEIF DK=1                                                     %
%       THEN counter=decr, set DS=0, set FCLK, keep FIFO's enabled,     %
%               Clear TMR, and continue.                                %
%       ELSE counter=hold, TMR, keep FIFO's enabled, and loop here.     %
%                                                                       %
%       Place local word counter in hold, negate FCLK, keep processor   %
%               and oport FIFO's enabled, set TMR, and JUMP to DATFBR.  %
%               This cycle ensures that the FSCC's minimum cycle time   %
%               is not violated in the same way as above.               %
%       Keep TMR set, do another NOP for the same reason as above, and  %
%               jump to BXFERLP.                                        %
%                                                                       %
%EOBIRQ:IF FBREQ THEN Release bus, jump to HOME.                         %
%       ELSEIF DS=1                                                     %
%       THEN counter=hold, TMR, continue.                               %
%       ELSE counter=hold, TMR, JUMP DSLOWH.                            %
%                                                                       %
%       --case DS=1--                                                   %
%       IF FBREQ THEN Release bus, jump to HOME.                        %
%       ELSEIF DK=1 THEN Clear TMR, continue.                           %
%       ELSE keep TMR true, and wait here.                              %
%                                                                       %
%       Wait two cycles to allow for FIFO write.                        %
%                                                                       %
%       IF FBREQ THEN Release bus, jump to HOME.                        %
%       ELSEIF WT=1                                                     %
%       THEN RD=1, MS=0, counter=hold, FIFO's disabled loop here.       %
%       ELSE DS=0, RD=0, MS=0, FIFO's Disabled continue.                %
%                                                                       %
%       IF FBREQ THEN Release bus, jump HOME.                           %
%       ELSEIF DK=0 THEN NTMR, EOBA, IRQ, jump HOME.                    %
%       ELSE TMR, wait here.                                            %
%                                                                       %
%DSLOWH:IF FBREQ THEN Release bus, jump to HOME.                         %
%       ELSEIF DK=0 THEN clear TMR, continue.                           %
%       ELSE TMR, wait here.                                            %
%                                                                       %
%       wait three cycles to ensure FIFO write, then IRQ, EOBA, jump    %
%       home.                                                           %
%                                                                       %
BXFERLP:IF (/I4 * I0) + I3 + /I6      %(/FRDK * FRSS1) + FEOB + FBREQ*%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP EOBIRQ;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
```

```
            JUMP BXFERLP;
        ELSEIF /I4              %/FRDK%
        THEN    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK SDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP BXFERLP;

DATFBRX:        [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

WAIT3:  IF (I4 * I0) + I3 + /I6                 %(FRDK * FRSS1) + FEOB + FBREQ*%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP EOBIRQ;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP WAIT3;
        ELSEIF I4               %FRDK%
        THEN    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK CDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP WAIT3;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP BXFERLP;

EOBIRQ:         IF /I0          %/FRSS1%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
```

```
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

        IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX0]
                JUMP INTF;
        ELSEIF I1               %FRDS%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP DSLOWH;

        Case--DS=1
DSHIGHH:IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX0]
                JUMP INTF;
        ELSEIF I4               %FRDK%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP DSHIGHH;

                [CHOLD NSRT NDW DATAINFIFO NFDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO NFDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

WAIT5:  IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
```

```
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK
MUX0]
                JUMP WAIT5;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

WAIT6:  IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF /I4              %FRDK=0%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE IRQ DFFDIS
                HOLDB EOBA NEG NRD MS0 XWT XDK XDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP WAIT6;

%       Case--DS=0
DSLOWH:         IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF /I4              %FRDK=0%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX1]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP DSLOWH;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX1]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX1]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE IRQ DFFEN
                HOLDB EOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX1]
                JUMP HOME;

%-------------------------------------------------------------------------%
```

```
%       Routine DATA_FIFO_PIPELINED_READ_1  (100ns transfers)          %
%       --FAST-- --WRITE or READ--                                     %
%--------------------------------------------------------------------%
%       FASTBUS pipeline block transfer read from the FASTBUS crate    %
%       segment to the FSCC's Output Port Data FIFO, and also the      %
%       Processor Data FIFO if enabled through the parrallel port.     %
%                                                                      %
%       Set FASTBUS xcievers for input (DDIR=1, SAB=1, FDOE*=0),       %
%              enable data and processor FIFO's (DFIFOEN=1, PFIFOEN=1),%
%              MS=3, RD=1, and continue.                               %
%                                                                      %
%       Note: DDIR, and SAB must be held high throughout the block     %
%              transfer operation.                                     %
%                                                                      %
%DATFP1:IF FBREQ THEN Release bus, jump to INTS.                        %
%       ELSEIF WT=1                                                    %
%       THEN Enable FASTBUS xceivers for input, MS=3, RD=1, Place local %
%              word counter into hold, and loop here.                  %
%       ELSE Enable FASTBUS xceivers for input, MS=3, RD=1, Place local %
%              word counter into count, set DS=1, and JUMP to DATFP1X:  %
%                                                                      %
%PXFR1LP:IF FBREQ THEN Release bus, jump to INTS.                       %
%       ELSEIF WT=1                                                    %
%       THEN Enable FASTBUS xceivers for input, MS=3, RD=1, Place local %
%              word counter into hold, and loop here.                  %
%       ELSEIF EOB                                                     %
%       THEN Enable FASTBUS xceivers for input, MS=3, RD=1, Place local %
%              word counter into hold mode, and JUMP to EOBPIPE.       %
%       ELSE Enable FASTBUS xceivers for input, MS=3, RD=1, Place local %
%              word counter into count, set DS=1, set FCLK, and        %
%              continue.                                               %
%                                                                      %
%DATFP1X:Place counter into hold mode, negate FCLK, and                 %
%              continue.  This       cycle provides the FSCC's        %
%              FASTBUS cycle time (100ns).                             %
%                                                                      %
%       IF FBREQ THEN Release bus, jump to HOME.                       %
%       ELSEIF WT=1                                                    %
%       THEN counter=hold, keep both FIFO's enabled and loop here.     %
%       ELSEIF EOB                                                     %
%       THEN counter=hold, keep FIFO's enabled and jump to EOBPIPE.    %
%       ELSE counter=decr, set DS=0, set FCLK, keep FIFO's enabled,    %
%              and continue.                                           %
%                                                                      %
%       Place local word counter in hold, negate FCLK, keep processor  %
%              and oport FIFO's enabled, and JUMP to PXFR1LP.          %
%              This cycle provides the FSCC's cycle time.              %
%                                                                      %
%EOBPIPE:IF SS <> 2 (Local counter terminated the block transfer)       %
%       THEN FCLK continue.                                            %
%       ELSE (SS=2 terminated the block transfer) continue.            %
%                                                                      %
%       IF FBREQ THEN Release bus, jump to HOME.                       %
%       ELSEIF DS=1                                                    %
%       THEN counter=hold, TMR, continue.                              %
%       ELSE counter=hold, TMR, JUMP DSLOWP.                           %
%                                                                      %
```

```
%       --case  DS=1--                                                  %
%DSHIGHP:IF FBREQ THEN Release bus, jump to HOME.                       %
%       ELSEIF DK=1 THEN Clear TMR, continue.                           %
%       ELSE keep TMR true, and wait here.                              %
%                                                                       %
%       Wait two cycles to allow for FIFO write.                        %
%                                                                       %
%       IF FBREQ THEN Release bus, jump to HOME.                        %
%       ELSEIF WT=1 THEN RD=1, counter=hold, disable FIFO's, loop here. %
%       ELSE RD=0, MS=0, DS=0, disable FIFO', continue.                 %
%                                                                       %
%       IF FBREQ THEN Release bus, jump to HOME.                        %
%       ELSEIF DK=0 THEN clear TMR, EOBA, IRQ, continue.                %
%       ELSE TMR, wait here.                                            %
%                                                                       %
%       --case  DS=0--                                                  %
%DSLOWP:IF FBREQ THEN Release bus, jump to HOME.                        %
%       ELSEIF DK=0 THEN clear TMR, CONTINUE.                           %
%       ELSE TMR, wait here.                                            %
%                                                                       %
%       Wait three cycles to ensure FIFO write, then IRQ, EOBA, jump    %
%       home.                                                           %
%                                                                       %
PXFR1LP:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP PXFR1LP;
        ELSEIF I3               %FEOB%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP EOBPIPE;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

DATFP1X:        [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

WAIT7:  IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
```

```
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP WAIT7;
        ELSEIF I3               %FEOB%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP EOBPIPE;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK CDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP PXFR1LP;

EOBPIPE:IF /I0          %/FRSS1%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

        IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I1               %FRDS%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX0]
                JUMP DSLOWP;

%       Case  DS=1                                                      %
DSHIGHP:IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I4               %FRDK%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
```

```
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX0]
                JUMP DSHIGHP;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

WAIT9:  IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK
MUX0]
                JUMP WAIT9;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG RD MS0 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

WAIT10:         IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF /I4      %FRDK=0%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE IRQ DFFDIS
                HOLDB EOBA NEG NRD MS0 XWT XDK XDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX0]
                JUMP WAIT10;

%       Case  DS=0                                                      %
DSLOWP:         IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF /I4              %FRDK=0%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
```

```
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX1]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO NFDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DSLOWP;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX1]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX1]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE IRQ DFFEN
                HOLDB EOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX1]
                JUMP HOME;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_PIPELINED_READ_2  (200ns transfers)             %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
%       This routine does a FASTBUS pipeline block transfer read from     %
%       the FASTBUS crate segment to the FSCC's Output Port Data FIFO,    %
%       and also the Processor Data FIFO.                                 %
%                                                                         %
%       Note: DDIR, and SAB must be held high throughout the block        %
%               transfer operation.                                       %
%                                                                         %
%DATFP2:IF FBREQ THEN Release bus, JUMP WAYHOME.                          %
%       ELSEIF WT=1                                                       %
%       THEN Enable FASTBUS xceivers for input, Enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into hold, and loop here.%
%       ELSEIF EOB                                                        %
%       THEN Enable FASTBUS xceivers for input, enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into hold mode, and        %
%               JUMP to EOBPIPE.                                          %
%       ELSE Enable FASTBUS xceivers for input, enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into count, set DS=1,      %
%               and JUMP to DATFP2X.                                      %
%                                                                         %
%PXFR2LP:IF FBREQ THEN Release bus, JUMP WAYHOME.                         %
%       ELSEIF WT=1                                                       %
%       THEN Enable FASTBUS xceivers for input, Enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into hold, and loop here.%
%       ELSEIF EOB                                                        %
%       THEN Enable FASTBUS xceivers for input, enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into hold mode, and        %
%               JUMP to EOBPIPE.                                          %
%       ELSE Enable FASTBUS xceivers for input, enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into count, set DS=1,      %
%               set FCLK, and continue.                                   %
%                                                                         %
```

```
%DATFP2X:Place counter into hold mode, negate FCLK, and                 %
%               continue.  These cycles provide the FSCC's              %
%               FASTBUS cycle time (200ns).                             %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       IF FBREQ THEN Release bus, jump to HOME.                        %
%       ELSEIF WT=1                                                     %
%       THEN counter=hold, keep both FIFO's enabled and loop here.      %
%       ELSEIF EOB                                                      %
%       THEN counter=hold, keep FIFO's enabled and jump to EOBPIPE.     %
%       ELSE counter=decr, set DS=0, set FCLK, keep FIFO's enabled,     %
%               and continue.                                           %
%                                                                       %
%       Place local word counter in hold, negate FCLK, keep processor   %
%               and oport FIFO's enabled, and Continue.                 %
%               These cycles provide the FSCC's cycle time.             %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       Jump to PXFR2LP. (add 50ns to cycle time).                      %
%                                                                       %
PXFR2LP:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP PXFR2LP;
        ELSEIF I3               %FEOB%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP EOBPIPE;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

DATFP2X:        [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
```

```
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

WAIT11:         IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP WAIT11;
        ELSEIF I3               %FEOB%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP EOBPIPE;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK CDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP PXFR2LP;

%----------------------------------------------------------------------%
%       Routine DATA_FIFO_PIPELINED_READ_4  (400ns transfers)          %
%       --FAST-- --READ or WRITE--                                     %
%----------------------------------------------------------------------%
%       This routine does a FASTBUS pipeline block transfer read from  %
%       the FASTBUS crate segment to the FSCC's Output Port Data FIFO, %
%       and also the Processor Data FIFO.                              %
%                                                                      %
%       Note: DDIR, and SAB must be held high throughout the block     %
%               transfer operation.                                    %
%                                                                      %
%DATFP4:IF FBREQ THEN Release bus, jump to WAYHOME.                    %
%       ELSEIF WT=1                                                    %
%       THEN Enable FASTBUS xceivers for input, enable data and        %
%               processor FIFO's, MS=3, RD=1, Place local word counter %
%               into hold, and loop here.                              %
%       ELSEIF EOB                                                     %
%       THEN Enable FASTBUS xceivers for input, enable FIFO's, Place   %
```

```
%               local word counter into hold mode, MS=3, RD=1, and JUMP %
%               to EOBPIPE.                                             %
%       ELSE Enable FASTBUS xceivers for input, enable FIFO's, Place    %
%               local word counter into count, MS=3, RD=1, set DS=1,    %
%               and JUMP to DATFP4X.                                    %
%                                                                       %
%PXFR4LP:IF FBREQ THEN Release bus, jump to WAYHOME.                     %
%       ELSEIF WT=1                                                     %
%       THEN Enable FASTBUS xceivers for input, enable data and         %
%               processor FIFO's, MS=3, RD=1, Place local word counter  %
%               into hold, and loop here.                               %
%       ELSEIF EOB                                                      %
%       THEN Enable FASTBUS xceivers for input, enable FIFO's, Place    %
%               local word counter into hold mode, MS=3, RD=1, and JUMP %
%               to EOBPIPE.                                             %
%       ELSE Enable FASTBUS xceivers for input, enable FIFO's, Place    %
%               local word counter into count, MS=3, RD=1, set DS=1,    %
%               set FCLK, and continue.                                 %
%                                                                       %
%       Place counter into hold mode, negate FCLK, and                  %
%               continue.  These cycles provide the FSCC's              %
%               FASTBUS cycle time (400ns).                             %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       IF FBREQ THEN Release bus, jump to HOME.                        %
%       ELSEIF WT=1                                                     %
%       THEN counter=hold, keep both FIFO's enabled and loop here.      %
%       ELSEIF EOB                                                      %
%       THEN counter=hold, keep FIFO's enabled and jump to EOBPIPE.     %
%       ELSE counter=decr, set DS=0, set FCLK, keep FIFO's enabled,     %
%               and continue.                                           %
%                                                                       %
%       Place local word counter in hold, negate FCLK, keep processor   %
%               and oport FIFO's enabled, and Continue.                 %
%               These cycles provide the FSCC's cycle time.             %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
%       Continue (add 50ns to cycle time).                              %
%                                                                       %
```

```
%       Jump to PXFR4LP.(add 50ns to cycle time).
%
PXFR4LP:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP PXFR4LP;
        ELSEIF I3               %FEOB%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP EOBPIPE;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

DATFP4X:        [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;


                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
```

```
                CONTINUE;

WAIT12:         IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP WAIT12;
        ELSEIF I3               %FEOB%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP EOBPIPE;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK CDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP PXFR4LP;
```

```
%-------------------------------------------------------------------------%
%       SLAVE_DATA_INPUT                                                  %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
%       Enable FASTBUS AD line receivers, Clear WT, jump to SLVDINX.      %
%                                                                         %
%       IF FBREQ then release bus, jump to INTS.                          %
%       ELSEIF WT=1, THEN wait here.                                      %
%       ELSEIF DS=1 THEN latch FASTBUS AD lines, set DK, DSACK,           %
%              continue.                                                  %
%       ELSE (DS=0) THEN latch FASTBUS AD lines, clear DK, DSACK,         %
%              continue.                                                  %
%                                                                         %
%       DSACK, Hold latched FASTBUS data, jump WAYHOME.                   %

SLVDINX:IF /I6          %FBREQ%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF I7       %FRWT%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 NDSAK
MUX0]
                JUMP SLVDINX;
        ELSEIF I1       %FRDS%
        THEN    [CDEF NSRT NDW DATAINLATCH FDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 XWT SDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 DSAK
MUX0]
                CONTINUE;
        ELSE    [CDEF NSRT NDW DATAINLATCH FDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 XWT CDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 DSAK
MUX0]
                CONTINUE;

                [CDEF NSRT NDW DATAINHOLD NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 DSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Null                                                              %
%       --FAST-- WRITE or READ--                                          %
%-------------------------------------------------------------------------%
NULLX:          [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT12 NDSAK
MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       End Of Event                                                      %
```

```
%       --SLOW-- --WRITE--                                                    %
%-------------------------------------------------------------------------%
%       --Two dummy words--                                                   %
%       DW (Data FIFO Write), DEOE, CEOE, DFFEN, Set FASTBUS interface        %
%               transceivers for output to FIFO (Data and Processor).         %
%               DSACK, continue.                                              %
%                                                                             %
%       Hold all lines for one more cycle (make sure that the write           %
%               pulse to the FIFO's is long enough).                          %
%                                                                             %
%       Hold DEOE true for one more cycle while DW is false.  This            %
%               provides the "hold" time for the fifo's.                      %
%                                                                             %
%       Set DW, for two cycles to insert another dummy word with DEOE         %
%               false.                                                        %
%                                                                             %
%       Hold DEOE true for one more cycle while DW is false, to               %
%               ensure that the fifo data hold time spec is met.              %
%                                                                             %
EOEX:           [CDEF NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR NSCLK NFCLK DEOE CEOE NTMR NCCLR STAT14 DSAK MUX1]
                CONTINUE;

                [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR NSCLK NFCLK DEOE NCEOE NTMR NCCLR STAT14 NDSAK
MUX1]
                CONTINUE;

                [CDEF NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT14 NDSAK
MUX1]
                CONTINUE;

                [CDEF NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT14 NDSAK
MUX1]
                CONTINUE;

                [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT14 NDSAK
MUX1]
                JUMP HOME;

%-------------------------------------------------------------------------%
%       Interrupt handler                                                     %
%-------------------------------------------------------------------------%
%       Being interrupted means that there was a FASTBUS error, or            %
%       the RB (Reset Bus) line has been driven true.  In either case         %
%       the sequencer's assigned task is simply to abort any operation        %
%       it is currently doing, and return to idle state.                      %
%                                                                             %
%       Returning DSACK to the 68020, is a prcisely timed function            %
```

```
%       which the sequencer must perform.  DSACK is asserted true       %
%       for exactly two sequencer cycles.  Then it is deasserted, and   %
%       the FBxS* lines may not be sampled for three sequencer cycles.  %
%       What this means is that when a routine wants to return DSACK    %
%       and return to idle, it must do three nop type cycles before     %
%       testing either of the two FBxS* lines.                          %
%                                                                       %
%       The interrupt handler is a strange routine since the IRQ line   %
%       is tested at almost every "IF" instruction there is.  If IRQ is %
Richard Kwarciany
Fermilab
Monday, 24-Sep-90  1300hrs
FSCC Version: 2
Revision: 4
EPS448
U106,U110,U108


PART:   EPS448,EPS448,EPS448

INPUTS:         I0@12
        I1@11
        I2@10
        I3@9
        I4@5
        I5@4
        I6@3
        I7@2


OUTPUTS:LC1@1           %       1       %
        LC0@28          %       2       %
        LCOE@27                 %       3       %
        /SRT@26                 %       4       %
        FW@25           %       5       %
        /POE@24                 %       6       %
        DDIR@23                 %       7       %
        CPAB@22                 %       8       %
        SAB@20          %       9       %
        CPBA@19                 %       10      %
        SBA@18          %       11      %
        /DG@17          %       12      %
        /FDOE@16        %       13      %
        /FCOE@15        %       14      %
        FPCREQ@14       %       15      %
        DFIFOEN@13      %       16      %

OUTPUTS:REL@1           %       17      %
        REQ@28          %       18      %
        FEOBA@27        %       19      %
        EG@25           %       20      %
        RD@24           %       21      %
        MS2@23          %       22      %
        MS1@22          %       23      %
        MS0@20          %       24      %
        CWT@19          %       25      %
```

```
        CDK@18          %       26      %
        SDK@17          %       27      %
        CDS@16          %       28      %
        SDS@15          %       29      %
        CAS@14          %       30      %
        SAS@13          %       31      %

OUTPUTS:PFIFOEN@1       %       32      %
        /SR@28          %       33      %
        SCLK@27                 %       34
        FCLK@26                 %       35
        DEOE@25                 %       36
        CEOE@24                 %       37
        TMR@23          %       38      %
        CCLEAR@22       %       39      %
        STAT3@20        %       40      %
        STAT2@19        %       41      %
        STAT1@18        %       42      %
        STAT0@17        %       43      %
        /DSACK@16       %       44      %
        /MUX2@15        %       45      %
        /MUX1@14        %       46      %
        /MUX0@13        %       47      %

%       DEFAULT = all bits false except: DDIR=1, FCOE*=0, MUX1*=0
%        12345678901234567890123456789012345678901234567       %
DEFAULT:[00010110000110100000000000000000010000000001101B]

%       Macros for Chip 1       %

MACROS:         CHOLD           ="000"          %LC1,LC0,LCOE%
        CLOAD           ="100"
        COUNT           ="110"
        CREAD           ="001"
        CDEF            ="000"

        SRT             ="0"            %SRT*%
        NSRT            ="1"

        DW              ="1"            %SDW%
        NDW             ="0"

        DATADEF                 ="1000001"
        %POE*,DDIR,CPAB,SAB,CPBA,SBA,DG*%
        DATAOUT                 ="0000000"
        DATAOUTLATCH    ="0000100"
        DATAOUTHOLD     ="1000010"
        DATAINPROC      ="0100000"
        DATAINLATCH     ="0110000"
        DATAINHOLD      ="0101000"
        DATAINFIFO      ="1101000"
        COUNTIN                 ="0000001"

        FDOE            ="0"            %FDOE*%
        NFDOE           ="1"

        FCOE            ="0"            %FCOE*%
```

```
        NFCOE           ="1"

        IRQ             ="0"            %FPCREQ*%
        NIRQ            ="1"

        DFFEN           ="1"            %DFIFOEN%
        DFFDIS          ="0"

        CTRLDEF                 ="0001011000011010"    %DEFAULTS FOR U42X40%

        Macros for Chip 2       %

        REQ             ="010000000000000"      %REL,REQ%
        REL             ="101000000101010"      %(EOEA, Clear DS,DK,AS,GK)%
        HOLDB           ="00"
        NREQ            ="00"

        EOBA            ="1"            %EOBA%
        NEOBA           ="0"

        EG              ="1"            %EG%
        NEG             ="0"

        RD              ="1"            %RD%
        NRD             ="0"

        MS0             ="000"          %MS2,MS1,MS0%
        MS1             ="001"
        MS2             ="010"
        MS3             ="011"
        MS4             ="100"
        MS5             ="101"
        MS6             ="110"
        MS7             ="111"

        CWT             ="1"            %CWT%
        XWT             ="0"

        SDK             ="01"           %CDK,SDK%
        CDK             ="10"
        XDK             ="00"

        SDS             ="01"           %CDS,SDS%
        CDS             ="10"
        XDS             ="00"

        SAS             ="01"           %CAS,SAS%
        CAS             ="10"
        XAS             ="00"

        FBDEF           ="000000000000000"      %DEFAULTS FOR U47X40%


%       Macros for Chip 3       %

        PFFEN           ="1"            %PFIFOEN%
        PFFDIS          ="0"
```

```
        SR              ="0"            %SR*%
        NSR             ="1"

        SCLK            ="1"            %SCLK%
        NSCLK           ="0"

        FCLK            ="1"            %FCLK%
        NFCLK           ="0"

        DEOE            ="1"            %DEOE%
        NDEOE           ="0"

        CEOE            ="1"            %CEOE%
        NCEOE           ="0"

        TMR             ="1"            %TMR%
        NTMR            ="0"

        CCLR            ="1"            %CCLEAR%
        NCCLR           ="0"

        STAT0           ="0000"                 %STATE3,STATE2,STATE1,STATE0%
        STAT1           ="0001"
        STAT2           ="0010"
        STAT3           ="0011"
        STAT4           ="0100"
        STAT5           ="0101"
        STAT6           ="0110"
        STAT7           ="0111"
        STAT8           ="1000"
        STAT9           ="1001"
        STAT10          ="1010"
        STAT11          ="1011"
        STAT12          ="1100"
        STAT13          ="1101"
        STAT14          ="1110"
        STAT15          ="1111"

        DSAK            ="0"            %DSACK*%
        NDSAK           ="1"

        MUX2            ="011"          %MUX2,MUX1,MUX0%
        MUX1            ="101"
        MUX0            ="110"

%       RESET0 = All bits false except: FCOE* FREL FCWT FCDK FCDS FCAS MUX1%
%                         12345678901234567890123456789012345678901234567%
        RESET0          ="00010110000110101000000011010100100000000001101"

%       RESET02 = All bits false except: FREL FCWT FCDK FCDS FCAS MUX1%
%                         12345678901234567890123456789012345678901234567%
        RESET02
        ="00010110000111101000000011010100100000000001101"

%-------------------------------------------------------------------------%
%       Here the 448's set all outputs to their default state, then       %
```

```
%       wait for either the 68020 to call, or somebody on FASTBUS        %
%       to want something.  When FBSEL is true, that means the 68020     %
%       is calling, and we need to get the vector from MUX2 and go to     %
%       the routine at that address.  If, on the other hand, FSLV is     %
%       true, we go to the slave handler.                                 %
%-------------------------------------------------------------------------%


PROGRAM:

RESET:  [RESETO2] JUMP SELECT;

%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%
%       Routine entry points.                                             %
%       Entry points are the first instruction of each routine.           %
%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%

%-------------------------------------------------------------------------%
%       Routine ARBITRATE                                                 %
%       --SLOW-- --WRITE--                                                %
%-------------------------------------------------------------------------%
ARB:    IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF I2       %FRDY%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT1 DSAK MUX1]
                JUMP ARBX;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD NFDOE FCOE NIRQ DFFDIS
                REQ
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT1 NDSAK
MUX0]
                JUMP ARB;


%-------------------------------------------------------------------------%
%       Routine BUS_RELEASE                                               %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%                                                                         %
%       This routine sets FREL true, clears DS, DK, and AS, then goes     %
%       home.  Note that since no data is needed, the control lines       %
%       latching the 68020 data       are returned to their default states, %
%       discarding the data.                                              %
%-------------------------------------------------------------------------%


BUSREL:                 [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
```

```
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT2 NDSAK
MUX1]
                JUMP BUSRELX;


%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%
%       Primary Address Cycles                                            %
        %
%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%

%-------------------------------------------------------------------------%
%       Routine ADDRESS_DATA_GEOGRAPHICAL                                 %
%                                                                         %
%       Geographical primary address cycle to data space.                 %
%       (we assert EG)                                                    %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%                                                                         %
%       IF FBREQ Release bus, TMR=0, JUMP WAYHOME.                        %
%       ELSEIF WT=1, loop here.                                           %
%       ELSEIF AK=1 THEN turn off FASTBUS AD line drivers, negate EG,     %
%              MS, and go home.                                           %
%       ELSE set AS, TMR (turn on short timer), and loop here.            %
%                                                                         %
%-------------------------------------------------------------------------%
ADDDG:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA EG NRD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDDG;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA EG NRD MS0 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDDG;


%-------------------------------------------------------------------------%
%       Routine ADDRESS_CSR_GEOGRAPHICAL                                  %
%                                                                         %
%       Geographical primary address cycle to CSR space.                  %
%       (We assert EG)                                                    %
%       --FAST-- --WRITE--                                                %
```

```
%-------------------------------------------------------------------------%
%                                                                         %
%       IF FBREQ THEN Release bus, JUMP WAYHOME.                          %
%       ELSEIF WT=1, loop here.                                           %
%       ELSEIF AK=1 THEN turn off FASTBUS AD line drivers, negate EG,     %
%              MS, and go home.                                           %
%       ELSE set AS, TMR (turn on short timer), and loop here.            %
%                                                                         %
%-------------------------------------------------------------------------%
ADDCG:  IF /I6          %FBREQ%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA EG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDCG;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA EG NRD MS1 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDCG;


%-------------------------------------------------------------------------%
%       Routine ADDRESS_DATA_LOGICAL                                      %
%                                                                         %
%       This routine is identical to ADDRESS_DATA_GEOGRAPHICAL            %
%       except that EG is not driven.                                     %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%                                                                         %
%       IF FBREQ THEN Release bus, TMR=0, JUMP WAYHOME.                   %
%       ELSEIF WT=1, loop here.                                           %
%       ELSEIF AK=1 THEN turn off FASTBUS AD line drivers, negate MS,     %
%              and go home.                                               %
%       ELSE set AS, TMR (turn on short timer), and loop here.            %
%                                                                         %
%-------------------------------------------------------------------------%
ADDDL:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS0 XWT XDK XDS XAS
```

```
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDDL;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS0 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDDL;

%-------------------------------------------------------------------------%
%       Routine ADDRESS_CSR_LOGICAL                                       %
%-------------------------------------------------------------------------%
%       This routine is identical to ADDRESS_CSR_GEOGRAPHICAL except      %
%       that EG is not driven.                                            %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%                                                                         %
%       IF FBREQ THEN Release bus, TMR=0, JUMP WAYHOME.                   %
%       ELSEIF WT=1, loop here.                                           %
%       ELSEIF AK=1 THEN turn off FASTBUS AD line drivers, negate MS,     %
%       and go home.                                                      %
%       ELSE set AS, TMR (turn on short timer), and loop here.            %
%                                                                         %
%-------------------------------------------------------------------------%
ADDCL:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX0]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDCL;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDCL;

%-------------------------------------------------------------------------%
%       Routine ADDRESS_DATA_BROADCAST                                    %
%                                                                         %
%       This routine does a broadcast primary address cycle to            %
%       DATA space.                                                       %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
```

```
%       IF FBREQ THEN Release bus, TMR=0, jump WAYHOME.                  %
%       ELSEIF WT=1, then loop here.                                     %
%       ELSEIF AK=1, Then turn off FASTBUS AD line drivers, negate MS,   %
%              and go home.                                              %
%       ELSE set AS, TMR (turn on short timer), and loop here.           %
%                                                                        %
%------------------------------------------------------------------------%
ADDDB:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDDB;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDDB;

%------------------------------------------------------------------------%
%       Routine ADDRESS_CSR_BROADCAST                                    %
%                                                                        %
%       This routine does a broadcast primary address cycle to           %
%       CSR space.                                                       %
%       --FAST-- --WRITE--                                               %
%------------------------------------------------------------------------%
%       IF FBREQ THEN Release bus, TMR=0, JUMP WAYHOME.                  %
%       ELSEIF WT=1, loop here.                                          %
%       ELSEIF AK=1 THEN turn off FASTBUS AD line drivers, negate MS,    %
%              and go home.                                              %
%       ELSE set AS, TMR (turn on short timer), and loop here.           %
%                                                                        %
%------------------------------------------------------------------------%
ADDCB:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK
MUX0]
                JUMP ADDCB;
        ELSEIF I5               %FRAK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
```

```
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT3 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK XDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT3 NDSAK MUX0]
                JUMP ADDCB;

%-------------------------------------------------------------------------%
%       Routine ADDRESS_RELEASE                                           %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%               EG=0 RD=0 MS=0 WT=0 Clear DK,DS,AS
ADDREL:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT4 NDSAK
MUX1]
                JUMP ADDREL;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT CDK CDS CAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT4 NDSAK
MUX0]
                JUMP ADDRELX;

%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%
%       Data Cycle Routines                                               %
%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_RANDOM_READ                                %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
DATPRR:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPRR;
        ELSEIF I4               %FRDK%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 DSAK MUX0]
```

```
                JUMP DATPRRX;
        ELSE    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS0 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DATPRR;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_RANDOM_WRITE                               %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
DATPRW:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPRW;
        ELSEIF I4               %FRDK%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPRWX;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS0 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DATPRW;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_SEC_ADDRESS_READ                           %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
DATPSR:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK
MUX0]
                JUMP DATPSR;
        ELSEIF I4               %FRDK%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 DSAK MUX0]
                JUMP DATPSRX;
        ELSE    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS2 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT6 NDSAK MUX0]
```

```
                JUMP DATPSR;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_SEC_ADDRESS_WRITE                          %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
DATPSW:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK
MUX0]
                JUMP DATPSW;
        ELSEIF I4               %FRDK%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK
MUX0]
                JUMP DATPSWX;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS2 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT6 NDSAK MUX0]
                JUMP DATPSW;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_BLOCK_TRANSFER_READ                        %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
DATPBR:                 [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPBRX;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_BLOCK_TRANSFER_WRITE                       %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
DATPBW:                 [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPBWX;


%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_BLOCK_TRANSFER_TERMINATE                   %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%       This routine does a graceful termination of a block transfer      %
%       where DS and DK are brought to their idle states without          %
%       effecting the slave.                                              %
```

```
%               MS=0 RD=0                                                    %
%               then                                                         %
%               DS=0                                                         %
DATPBT:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX1]
                JUMP DATPBT;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT XDK CDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Routine LOCAL_COUNTER_LOAD                                        %
%       --SLOW-- --WRITE--                                                %
%-------------------------------------------------------------------------%
LC_LOAD:        [CLOAD NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT9 DSAK MUX1]
                JUMP LC_LOADX;


%-------------------------------------------------------------------------%
%       Routine LOCAL_COUNTER_READ                                        %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
LC_READ:        [CREAD NSRT NDW COUNTIN NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT9 DSAK MUX1]
                JUMP LC_READX;

%-------------------------------------------------------------------------%
%       Routine FIFO_WRITE                                                %
%       --SLOW-- --WRITE--                                                %
%-------------------------------------------------------------------------%
FIFO_WR:        [CDEF NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT10 DSAK MUX1]
                JUMP FIFO_WRX;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_BLOCK_TRANSFER_READ                             %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
DATFBR:         IF /I6                  %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
```

```
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP DATFBR;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK SDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP DATFBRX;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_PIPELINED_READ_1  (100ns transfers)             %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
DATFP1:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP1;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP1X;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_PIPELINED_READ_2  (200ns transfers)             %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
DATFP2:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP2;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP2X;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_PIPELINED_READ_4  (400ns transfers)             %
%       --FAST-- --READ or WRITE--                                        %
%-------------------------------------------------------------------------%
DATFP4:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
```

```
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP4;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DATFP4X;

%-------------------------------------------------------------------------%
%       SLAVE_DATA_INPUT                                                  %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
SLVDIN:                 [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 CWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 NDSAK
MUX0]
                JUMP SLVDINX;

%-------------------------------------------------------------------------%
%       SLAVE_DATA_OUTPUT                                                 %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%       (WT has been cleared back at LETSGO).                             %
%       IF FBREQ* THEN release bus, jump WAYHOME.                         %
%       ELSEIF WT=1 THEN wait here.                                       %
%       ELSEIF DS=1 THEN set DK jump HOME.                                %
%       ELSE clear DK jump HOME.                                          %

SLVDOUT:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP WAYHOME;
        ELSEIF I7       %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 NDSAK
MUX0]
                JUMP SLVDOUT;
        ELSEIF I1       %FRDS%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 XWT SDK XDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 NDSAK
MUX1]
                JUMP WAYHOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 XWT CDK XDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 NDSAK
MUX1]
                JUMP WAYHOME;
```

```
%-------------------------------------------------------------------------%
%       Null                                                              %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
NULL:           [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT12 NDSAK
MUX1]
                JUMP NULLX;


%-------------------------------------------------------------------------%
%       End Of Event                                                      %
%       --SLOW-- --WRITE --                                               %
%-------------------------------------------------------------------------%
EOE:            [CHOLD NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR NSCLK NFCLK DEOE CEOE NTMR NCCLR STAT14 DSAK MUX1]
                JUMP EOEX;


%-------------------------------------------------------------------------%
%       Bulb Test                                                         %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
BULBTST:        [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE IRQ DFFDIS
                HOLDB NEOBA EG RD MS7 XWT SDK SDS SAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT11 NDSAK
MUX1]
                JUMP BULBTST;


%-------------------------------------------------------------------------%
%       Sequencer Select                                                  %
%       Wait here to be selected by the 68020                             %
%               FB1S* + FB2S means that the FASTBUS standard sequencer    %
%                       has been selected.                                %
%               FB1S* * FB2S* means that the USER sequencer has been      %
%                       selected.                                         %
%                                                                         %
%       Whichever sequencer has not been selected, puts it's outputs      %
%               into high Z, and is not accessable unless it is reset.    %
%-------------------------------------------------------------------------%
SELECT:         IF /I1 * /I5            %FB1S* * FB2S*%
        THEN [Z] CONTINUE;
        ELSEIF (/I1 * I5) + (I1 * /I5)          %FB2S* + FB1S*%
        THEN [RESETO2] JUMP HOME;
        ELSE [RESETO2] JUMP SELECT;

TURNOFF:[Z] JUMP TURNOFF;
%-------------------------------------------------------------------------%
%       If, in the cycle executed just previous to returning here,        %
%       DSACK was asserted, then the select line (FBxS*)                  %
%       will still be asserted (it takes the 68020 and it's PAL's a       %
%       certain amount of time to negate a select line after it sees      %
%       DSACK).  HOME would then see FBxS* true and think that it is      %
```

```
%       being selected again, when, in fact, FBxS* is just lingering     %
%       from the previous cycle.  As you might imagine, if this were     %
%       to happen, the sequencer would jump to what ever address         %
%       just happened to be on the 68020's address bus at that time      %
%       and anarchy would reign.  So, to keep order, if we               %
%       want to come HOME right after DSACK, then we instead return      %
%       here where three NOP type instructions are executed.  This       %
%       allows the 68020 enough time to negate FBxS* from any            %
%       preveously executed cycle.                                       %
%-------------------------------------------------------------------------%
WAYHOME:        [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX1]
                CONTINUE;

                [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX1]
                CONTINUE;

                [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX1]
                CONTINUE;

%       Since the IOD lines could be latched on the next cycle (if there%
%       is a pending instruction), the Processor transceivers must be    %
%       set to DATAOUT before coming HOME.  Any instructions leading to %
%       HOME must have the processor transceivers set in this way or it %
%       will be         bad.                                             %

%       When HOME, all outputs are default                        %
HOME:   IF /I1          %FB1S* (Fast cycle, return DSACK now.)%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 DSAK MUX2]
                JUMP LETSGOF;
        ELSEIF /I5      %FB2S* (Slow cycle, return DSACK in routine.%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX2]
                JUMP LETSGOS;
        ELSE    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX1]
                JUMP HOME;

%-------------------------------------------------------------------------%
%       These are the fast and slow jumps to the appropriate FASTBUS     %
%       primative routines as determined by the vector received from     %
%       the '020 at MUX2.                                                %
%-------------------------------------------------------------------------%
```

```
LETSGOF:        [CDEF NSRT NDW DATAOUTLATCH NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 DSAK MUX2]
                PUSHI;

%       Clear WT here is for Slave write routine, dosn't effect anything
else.%
                [CDEF NSRT NDW DATAOUTHOLD NFDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 CWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX0]
                RETURN;

LETSGOS:        [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX2]
                PUSHI;

                [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT0 NDSAK
MUX0]
                RETURN;

%-------------------------------------------------------------------------%
%       Routine ARBITRATE                                                 %
%       --SLOW-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%                                                                         %
%       Starting with something simple, this routine checks to make       %
%       sure that FBREQ is false, then sets FREQ true, waits for          %
%       FRDY (means that we have the bus), then returns DSACK and goes    %
%       home.                                                             %
%-------------------------------------------------------------------------%
ARBX:           [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT1 DSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Bus Release                                                       %
%-------------------------------------------------------------------------%
%       Release Bus, Status=2                                             %
%                                                                         %
%       Status=2, Status clock = 1                                        %
BUSRELX:        [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT2 NDSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Routine ADDRESS_RELEASE                                           %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%               Wait for AK to go low, then go HOME.                      %
```

```
ADDRELX:IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I5               %FRAK%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT CDK CDS CAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT4 NDSAK MUX0]
                JUMP ADDRELX;
        ELSE    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT4 NDSAK MUX1]
                JUMP HOME;

%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%
%       Data Cycle Routines                                               %
%-------------------------------------------------------------------------%
%-------------------------------------------------------------------------%

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_RANDOM_READ                                %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, JUMP TO INTS.           %
%       ELSEIF WT=1 THEN Enable FASTBUS transceivers and drivers          %
%               (FDOE*=0,DDIR=1,CPBA=0, DSAB=0,DSBA=0,DG*=0), enable      %
%               processor transceivers (POE*=0), Assert RD=1, MS=0 and    %
%               loop here.                                                %
%       ELSEIF DK=1 THEN return DSACK, and continue.                      %
%       ELSE Enable FASTBUS AD line drivers, enable proccessor xceivers   %
%               Assert RD=1, MS=0, DS(u), TMR, and loop here.             %
%                                                                         %
%       Hold DSACK true for one more cycle, and continue.                 %
%                                                                         %
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, NDSACK, JUMP TO         %
%               WAYHOME.                                                  %
%       ELSEIF WT=1 THEN negate DSACK, disable processor and FASTBUS      %
%               transceivers, continue to assert RD=1, MS=0, and DS=1,    %
%               and loop here.                                            %
%       ELSEIF DK=0 THEN negate DSACK, disable processor and FASTBUS      %
%               transceivers, deassert TMR, RD, and MS, and go WAYHOME.   %
%       ELSE negate DSACK, disable processor and FASTBUS transceivers,    %
%               deassert RD, MS, and DS, assert TMR, and loop here.       %
%                                                                         %
%-------------------------------------------------------------------------%
DATPRRX:        [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 DSAK MUX0]
                CONTINUE;

WAIT1:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
```

```
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP WAIT1;
        ELSEIF /I4              %/FRDK%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP WAYHOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP WAIT1;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_RANDOM_WRITE                               %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, JUMP TO WAYHOME.        %
%       ELSEIF WT=1 THEN Enable FASTBUS transceivers and drivers          %
%               (FDOE*=0, DDIR=0, CPBA=0, DSAB=0, DSBA=1, DG*=0),         %
%               Assert RD=0, MS=0 and loop here.                          %
%       ELSEIF DK=1 THEN disable FASTBUS transceivers and drivers, and    %
%               continue.                                                 %
%       ELSE Assert DS(u), TMR, keep FASTBUS lines asserted, and loop     %
%               here.                                                     %
%       be bop a lula she's my baby                                       %
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, JUMP TO HOME.           %
%       ELSEIF WT=1 THEN continue to assert DS=1, and loop here.          %
%       ELSEIF DK=0 THEN go HOME.                                         %
%       ELSE deassert DS, assert TMR, and loop here.                      %
%-------------------------------------------------------------------------%
DATPRWX:IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPRWX;
        ELSEIF /I4              %/FRDK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
```

```
                JUMP DATPRWX;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_SEC_ADDRESS_READ                           %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, JUMP TO INTS.           %
%       ELSEIF WT=1 THEN Enable FASTBUS transceivers and drivers          %
%               (FDOE*=0, DDIR=1,CPBA=0, DSAB=0,DSBA=0,DG*=0), enable     %
%               processor transceivers (POE*=0), Assert RD=1, MS=2 and    %
%               loop here.                                                %
%       ELSEIF DK=1 THEN return DSACK, and continue.                      %
%       ELSE Assert DS(u), TMR, and loop here.                            %
%                                                                         %
%       Hold DSACK true for one more cycle, and continue.                 %
%                                                                         %
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, DEassert DSACK, jump    %
%               to WAYHOME.                                               %
%       ELSEIF WT=1 THEN negate DSACK, disable processor and FASTBUS      %
%               transceivers, continue to assert RD=1, MS=2, and DS=1,    %
%               and loop here.                                            %
%       ELSEIF DK=0 THEN negate DSACK, disable processor and FASTBUS      %
%               transceivers, deassert RD, and MS, and go WAYHOME.        %
%       ELSE negate DSACK, disable processor and FASTBUS transceivers,    %
%               deassert RD, MS, and DS, assert TMR, and loop here.       %
%-------------------------------------------------------------------------%
DATPSRX:        [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 DSAK MUX0]
                CONTINUE;

WAIT2:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG RD MS2 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK
MUX0]
                JUMP WAIT2;
        ELSEIF /I4              %/FRDK%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK MUX1]
                JUMP WAYHOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS2 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT6 NDSAK MUX0]
                JUMP WAIT2;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_SEC_ADDRESS_WRITE                          %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
```

```
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, jump to WAYHOME.      %
%       ELSEIF WT=1 THEN Enable FASTBUS transceivers and drivers        %
%              (FDOE*=0, DDIR=0, CPBA=0, DSAB=0, DSBA=1, DG*=0),        %
%              Assert RD=0, MS=2 and loop here.                         %
%       ELSEIF DK=1 THEN disable FASTBUS transceivers and drivers, and  %
%              continue.                                                %
%       ELSE Assert DS(u), keep FASTBUS lines asserted, assert TMR,     %
%              and loop here.                                           %
%       IF FBREQ THEN clear FASTBUS lines, TMR=0, jump to HOME.         %
%       ELSEIF WT=1 THEN continue to assert DS=1, and loop here.        %
%       ELSEIF DK=0 THEN go HOME.                                       %
%       ELSE deassert DS, assert TMR, and loop here.                    %
%-----------------------------------------------------------------------%
DATPSWX:IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK
MUX0]
                JUMP DATPSWX;
        ELSEIF /I4              %/FRDK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT6 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT6 NDSAK MUX0]
                JUMP DATPSWX;

%-----------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_BLOCK_TRANSFER_READ                      %
%       --SLOW-- --READ--                                               %
%-----------------------------------------------------------------------%
%       This routine is going to be a little bit different than any of  %
%              the others, in that, DSACK will have to be returned      %
%              after each DK(t) to allow the 68020 to read the data.    %
%       Here goes:                                                      %
%                                                                       %
%       Turn on FASTBUS AD line receivers, Assert RD=1, MS=1,switch     %
%              to MUX1 and continue.                                    %
%       IF FBREQ THEN Release bus, JUMP TO INTS.                        %
%       ELSEIF WT=1 THEN Assert RD=1, MS=1, on FASTBUS  Turn on FASTBUS %
%              xceivers and enable processor xceivers                   %
%              (POE*=0, DDIR=1, CPBA=0, DSAB=0, DSBA=0, DG*=0, FDOE*=0)%
%              and loop here.                                           %
%       ELSEIF DS=0 THEN Assert RD=1, MS=1, DS(u), enable FASTBUS and   %
%              processor transceivers (POE*=0, DDIR=1, CPBA=0,          %
%              DSAB=0, DSBA=0, DG*=0, FDOE*=0)and assert TMR, and JUMP  %
%              to DKUPR.                                                %
%       ELSE Assert RD=1, MS=1, DS(d), enable FASTBUS and processor     %
%              transceivers (POE*=0, DDIR=1, CPBA=0, DSAB=0,            %
```

```
%               DSBA=0, DG*=0, FDOE*=0) assert TMR, and JUMP to DKDWNR.  %
%DKDWNR                                                                  %
%       IF FBREQ THEN RD=1, MS=1, Clear DS, jump to INTS.                %
%       ELSEIF DK=0 THEN Assert DSACK, continue to drive FASTBUS lines,  %
%               and processor and FASTBUS transceivers, deassert TMR,    %
%               and JUMP to next1.                                       %
%       ELSE Continue to drive FASTBUS lines, and transceivers, and      %
%               TMR, and loop here.                                      %
%DKUPR                                                                   %
%       IF FBREQ THEN RD=1, MS=1, Clear DS, assert DSACK, jump to        %
%               NEXT2.                                                   %
%       ELSEIF DK=1 THEN Assert DSACK, continue to drive FASTBUS lines,  %
%               and processor and FASTBUS transceivers, deassert TMR,    %
%               and JUMP to next1.                                       %
%       ELSE Continue to drive FASTBUS lines, and transceivers,          %
%               assert TMR, and loop here.                               %
%NEXT1                                                                   %
%       Continue to drive FASTBUS lines, DSACK, and transceivers, and    %
%               Continue.                                                %
%       Continue to drive FASTBUS lines, and transceivers negate DSACK   %
%               and JUMP WAYHOME.                                        %
%                                                                        %
%NEXT2  Clear FASTBUS lines, release bus, keep DSACK asserted, jump       %
%               to WAYHOME.                                              %
%------------------------------------------------------------------------%
DATPBRX:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPBRX;
        ELSEIF /I1              %/FRDS%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKUPR;
        ELSE    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKDWNR;

DKDWNR:         IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG RD MS1 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF /I4              %/FRDK%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 DSAK MUX1]
```

```
                JUMP NEXT1;
        ELSE    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKDWNR;

DKUPR:  IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG RD MS1 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 DSAK
MUX1]
                JUMP NEXT2;
        ELSEIF I4               %FRDK%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 DSAK MUX1]
                JUMP NEXT1;
        ELSE    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKUPR;

NEXT1:          [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 DSAK MUX1]
                CONTINUE;

                [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP WAYHOME;

NEXT2:          [CTRLDEF
                REL
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 DSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Routine DATA_PROCESSOR_BLOCK_TRANSFER_WRITE                       %
%       --FAST-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%       As with DATPBR, DSACK will have to be returned after each DK(t)   %
%               to allow the 68020 to output new data.                    %
%                                                                         %
%       Keep FASTBUS AD line drivers enabled, assert MS=1, set MUX1,      %
%               and continue.                                             %
%       IF FBREQ THEN Release bus, JUMP TO WAYHOME.                       %
%       ELSEIF WT=1 THEN Assert MS=1, keep FASTBUS transceivers enabled,  %
%               and enable FASTBUS drivers (POE*=1, DDIR=0,               %
%               CPBA=0, DSAB=0, DSBA=1, DG*=0, FDOE*=0), and loop here.   %
%       ELSEIF DS=0 THEN Assert MS=1, DS(u), TMR, keep FASTBUS            %
%               transceivers enabled and enable FASTBUS drivers (POE*=1,  %
%               DDIR=0, CPBA=0, DSAB=0, DSBA=1, DG*=0, FDOE*=0), and      %
%               JUMP to DKUPW.                                            %
%       ELSE Assert MS=1, DS(d), TMR, keep FASTBUS transceivers enabled,  %
%               and enable FASTBUS drivers (POE*=1, DDIR=0,               %
%               CPBA=0, DSAB=0, DSBA=1, DG*=0, FDOE*=0), and JUMP to      %
```

```
%               DKDWNW.                                                 %
%DKDWNW                                                                 %
%       IF FBREQ THEN Release bus, JUMP TO HOME.                        %
%       ELSEIF DK=0 THEN deassert MS, TMR, disable FASTBUS transceivers %
%               and drivers (POE*=0, DDIR=0, CPBA=0, DSAB=0, DSBA=0,    %
%               DG*=1, FDOE*=1), and JUMP home.                         %
%       ELSE continue to assert MS, TMR, and keep FASTBUS transceivers  %
%               and drivers enabled, and loop here.                     %
%DKUPW                                                                  %
%       IF FBREQ THEN Release bus, JUMP TO HOME.                        %
%       ELSEIF DK=1 THEN deassert MS, TMR, disable FASTBUS transceivers %
%               and drivers (POE*=0, DDIR=0, CPBA=0, DSAB=0, DSBA=0,    %
%               DG*=1, FDOE*=1), and JUMP home.                         %
%       ELSE continue to assert MS, TMR, and keep FASTBUS transceivers  %
%               and drivers enabled, and loop here.                     %
%-----------------------------------------------------------------------%
DATPBWX:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK
MUX0]
                JUMP DATPBWX;
        ELSEIF /I1              %/FRDS%
        THEN    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK SDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKUPW;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKDWNW;

DKDWNW:         IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF /I4              %/FRDK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKDWNW;

DKUPW:  IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
```

```
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I4               %FRDK%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT5 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CDEF NSRT NDW DATAOUTHOLD FDOE FCOE NIRQ DFFDIS
                HOLDB NEOBA NEG NRD MS1 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT5 NDSAK MUX0]
                JUMP DKUPW;

%-------------------------------------------------------------------------%
%       Routine LOCAL_COUNTER_LOAD                                        %
%       --SLOW-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%       This routine loads the Local Word Counter with data from          %
%       the processor.  Only the lower 12 bits of the 32 bit word         %
%       are used.                                                         %
%                                                                         %
%       Set POE* true, put Local Word Counter into load mode,             %
%               assert DSACK and continue.                                %
%       Negate POE*, put Local Word Counter into hold, keep DSACK         %
%               asserted, and continue.                                   %
%       Negate DSACK (hold DSACK false for two cycles to prevent          %
%               sequencer from seeing FBSEL from this cycle when          %
%               returning home).                                          %
%                                                                         %
LC_LOADX:       [CHOLD NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT9 DSAK MUX1]
                CONTINUE;

                [CHOLD NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT9 NDSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Routine LOCAL_COUNTER_READ                                        %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
%       Routine to allow the processor to read the value of the Local     %
%               Word Counter.                                             %
%                                                                         %
%       Set POE* true, put Local Word Counter into Read mode, assert      %
%               DSACK, and continue.                                      %
%       Keep all outputs the same for one more cycle.                     %
%       Negate DSACK (hold DSACK false for two cycles to prevent          %
%               sequencer from seeing FBSEL from this cycle when          %
%               returning home).                                          %
%                                                                         %
LC_READX:       [CREAD NSRT NDW COUNTIN NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT9 DSAK MUX1]
                JUMP WAYHOME;
```

```
%-------------------------------------------------------------------------%
%       Routine FIFO_WRITE                                                %
%       --SLOW-- --WRITE--                                                %
%-------------------------------------------------------------------------%
%       Routine to write a word from the 68020 into the processor         %
%       and data FIFO's.                                                  %
%                                                                         %
%       Enable processor transceivers (POE*=0), set FIFO Write true       %
%              (DW=1), enable FIFO's, set DSACK true.                     %
%       Set FCLK true.                                                    %
%       Negate DSACK, DW, FCLK, and POE*, JUMP TO WAYHOME.                %
%                                                                         %
FIFO_WRX:       [CDEF NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFEN NSR SCLK FCLK NDEOE NCEOE NTMR NCCLR STAT10 DSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_BLOCK_TRANSFER_READ                             %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
%       This routine does a FASTBUS handshake block transfer read from    %
%       the FASTBUS crate segment to the FSCC's Output Port Data FIFO,    %
%       and also the Processor Data FIFO.                                 %
%-------------------------------------------------------------------------%
%       Note: DDIR, and SAB must be held high throughout the block        %
%              transfer operation.                                        %
%                                                                         %
%DATFBR:IF FBREQ                                                          %
%       THEN release bus jump to INTF                                     %
%       ELSEIF WT=1                                                       %
%       THEN Enable FASTBUS xceivers, enable processor and Data FIFO's    %
%              MS=1, RD=1, Place local word counter into hold, and        %
%              loop here.                                                 %
%       ELSE Turn on FASTBUS xcievers, enable processor and Data FIFO's   %
%              MS=1, RD=1, Place local word counter into count, set       %
%              DS=1, NO FCLK!, and jump to DATFBRX.                       %
%                                                                         %
%BXFERLP:IF (/FRDK * FRSS1) + EOB + FBREQ                                 %
%       THEN Enable FASTBUS xceivers, enable processor and Data FIFO's    %
%              MS=1, RD=1, Place local word counter into hold mode,       %
%              and JUMP to EOBIRQ.                                        %
%       ELSEIF WT=1                                                       %
%       THEN Enable FASTBUS xceivers, enable processor and Data FIFO's    %
%              MS=1, RD=1, Place local word counter into hold, and        %
%              loop here.                                                 %
%       ELSEIF DK=0                                                       %
%       THEN Turn on FASTBUS xcievers, enable processor and Data FIFO's   %
%              MS=1, RD=1, Place local word counter into count, set       %
%              DS=1, set FCLK, and continue.                              %
%       ELSE Place local word counter into hold, and loop here.           %
%                                                                         %
%DATFBRX:Place counter into hold mode, negate FCLK, set TMR, and          %
%              continue.  This       cycle ensures that the FSCC's minimum %
%              FASTBUS cycle time (100ns) is not violated.  If some       %
%              slave returns DK too quickly, the FSCC could               %
```

```
%               try to do another DS before the FIFO's were ready for    %
%               the next word.  This cycle keeps the sequencer from      %
%               looking for DK before it should.                         %
%                                                                        %
%       Keep TMR true, and do another NOP for the same reason as above.  %
%                                                                        %
%       IF (FRDK* FRSS1) + EOB + FBREQ                                   %
%       THEN counter=hold, NTMR, keep FIFO's enabled and jump to EOBIRQ.%
%       ELSEIF WT=1                                                      %
%       THEN counter=hold, NTMR, keep both FIFO's enabled and loop here.%
%       ELSEIF DK=1                                                      %
%       THEN counter=decr, set DS=0, set FCLK, keep FIFO's enabled,      %
%               Clear TMR, and continue.                                 %
%       ELSE counter=hold, TMR, keep FIFO's enabled, and loop here.      %
%                                                                        %
%       Place local word counter in hold, negate FCLK, keep processor    %
%               and oport FIFO's enabled, set TMR, and JUMP to DATFBR.   %
%               This cycle ensures that the FSCC's minimum cycle time    %
%               is not violated in the same way as above.                %
%       Keep TMR set, do another NOP for the same reason as above, and   %
%               jump to BXFERLP.                                         %
%                                                                        %
%EOBIRQ:IF FBREQ THEN Release bus, jump to HOME.                          %
%       ELSEIF DS=1                                                      %
%       THEN counter=hold, TMR, continue.                                %
%       ELSE counter=hold, TMR, JUMP DSLOWH.                             %
%                                                                        %
%       --case DS=1--                                                    %
%       IF FBREQ THEN Release bus, jump to HOME.                         %
%       ELSEIF DK=1 THEN Clear TMR, continue.                            %
%       ELSE keep TMR true, and wait here.                               %
%                                                                        %
%       Wait two cycles to allow for FIFO write.                         %
%                                                                        %
%       IF FBREQ THEN Release bus, jump to HOME.                         %
%       ELSEIF WT=1                                                      %
%       THEN RD=1, MS=0, counter=hold, FIFO's disabled loop here.        %
%       ELSE DS=0, RD=0, MS=0, FIFO's Disabled continue.                 %
%                                                                        %
%       IF FBREQ THEN Release bus, jump HOME.                            %
%       ELSEIF DK=0 THEN NTMR, EOBA, IRQ, jump HOME.                     %
%       ELSE TMR, wait here.                                             %
%                                                                        %
%DSLOWH:IF FBREQ THEN Release bus, jump to HOME.                          %
%       ELSEIF DK=0 THEN clear TMR, continue.                            %
%       ELSE TMR, wait here.                                             %
%                                                                        %
%       wait three cycles to ensure FIFO write, then IRQ, EOBA, jump     %
%       home.                                                            %
%                                                                        %
BXFERLP:IF (/I4 * I0) + I3 + /I6      %(/FRDK * FRSS1) + FEOB + FBREQ*%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP EOBIRQ;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
```

```
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP BXFERLP;
        ELSEIF /I4              %/FRDK%
        THEN    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK SDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP BXFERLP;

DATFBRX:        [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

WAIT3:  IF (I4 * I0) + I3 + /I6                 %(FRDK * FRSS1) + FEOB + FBREQ*%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP EOBIRQ;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                JUMP WAIT3;
        ELSEIF I4               %FRDK%
        THEN    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK CDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP WAIT3;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP BXFERLP;

EOBIRQ:         IF /I0          %/FRSS1%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
```

```
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

        IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX0]
                JUMP INTF;
        ELSEIF I1               %FRDS%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP DSLOWH;

%       Case--DS=1
DSHIGHH:IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX0]
                JUMP INTF;
        ELSEIF I4               %FRDK%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP DSHIGHH;

                [CHOLD NSRT NDW DATAINFIFO NFDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO NFDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

WAIT5:  IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
```

```
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK
MUX0]
                JUMP WAIT5;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG NRD MS0 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                CONTINUE;

WAIT6:  IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF /I4              %FRDK=0%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE IRQ DFFDIS
                HOLDB EOBA NEG NRD MS0 XWT XDK XDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX1]
                JUMP HOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP WAIT6;

%       Case--DS=0
DSLOWH:         IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF /I4              %FRDK=0%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX1]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT7 NDSAK MUX0]
                JUMP DSLOWH;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX1]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX1]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE IRQ DFFEN
                HOLDB EOBA NEG RD MS1 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT7 NDSAK MUX1]
                JUMP HOME;
```

```
%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_PIPELINED_READ_1  (100ns transfers)             %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
%       FASTBUS pipeline block transfer read from the FASTBUS crate       %
%       segment to the FSCC's Output Port Data FIFO, and also the         %
%       Processor Data FIFO if enabled through the parrallel port.        %
%                                                                         %
%       Set FASTBUS xcievers for input (DDIR=1, SAB=1, FDOE*=0),          %
%               enable data and processor FIFO's (DFIFOEN=1, PFIFOEN=1),  %
%               MS=3, RD=1, and continue.                                 %
%                                                                         %
%       Note: DDIR, and SAB must be held high throughout the block        %
%               transfer operation.                                       %
%                                                                         %
%DATFP1:IF FBREQ THEN Release bus, jump to INTS.                          %
%       ELSEIF WT=1                                                       %
%       THEN Enable FASTBUS xceivers for input, MS=3, RD=1, Place local   %
%               word counter into hold, and loop here.                    %
%       ELSE Enable FASTBUS xceivers for input, MS=3, RD=1, Place local   %
%               word counter into count, set DS=1, and JUMP to DATFP1X:    %
%                                                                         %
%PXFR1LP:IF FBREQ THEN Release bus, jump to INTS.                         %
%       ELSEIF WT=1                                                       %
%       THEN Enable FASTBUS xceivers for input, MS=3, RD=1, Place local   %
%               word counter into hold, and loop here.                    %
%       ELSEIF EOB                                                        %
%       THEN Enable FASTBUS xceivers for input, MS=3, RD=1, Place local   %
%               word counter into hold mode, and JUMP to EOBPIPE.         %
%       ELSE Enable FASTBUS xceivers for input, MS=3, RD=1, Place local   %
%               word counter into count, set DS=1, set FCLK, and          %
%               continue.                                                 %
%                                                                         %
%DATFP1X:Place counter into hold mode, negate FCLK, and                   %
%               continue.  This       cycle provides the FSCC's           %
%               FASTBUS cycle time (100ns).                               %
%                                                                         %
%       IF FBREQ THEN Release bus, jump to HOME.                          %
%       ELSEIF WT=1                                                       %
%       THEN counter=hold, keep both FIFO's enabled and loop here.        %
%       ELSEIF EOB                                                        %
%       THEN counter=hold, keep FIFO's enabled and jump to EOBPIPE.       %
%       ELSE counter=decr, set DS=0, set FCLK, keep FIFO's enabled,       %
%               and continue.                                             %
%                                                                         %
%       Place local word counter in hold, negate FCLK, keep processor     %
%               and oport FIFO's enabled, and JUMP to PXFR1LP.            %
%               This cycle provides the FSCC's cycle time.                %
%                                                                         %
%EOBPIPE:IF SS <> 2 (Local counter terminated the block transfer)         %
%       THEN FCLK continue.                                               %
%       ELSE (SS=2 terminated the block transfer) continue.               %
%                                                                         %
%       IF FBREQ THEN Release bus, jump to HOME.                          %
%       ELSEIF DS=1                                                       %
%       THEN counter=hold, TMR, continue.                                 %
%       ELSE counter=hold, TMR, JUMP DSLOWP.                              %
```

```
%                                                                    %
%       --case  DS=1--                                               %
%DSHIGHP:IF FBREQ THEN Release bus, jump to HOME.                     %
%       ELSEIF DK=1 THEN Clear TMR, continue.                        %
%       ELSE keep TMR true, and wait here.                           %
%                                                                    %
%       Wait two cycles to allow for FIFO write.                     %
%                                                                    %
%       IF FBREQ THEN Release bus, jump to HOME.                     %
%       ELSEIF WT=1 THEN RD=1, counter=hold, disable FIFO's, loop here.  %
%       ELSE RD=0, MS=0, DS=0, disable FIFO', continue.              %
%                                                                    %
%       IF FBREQ THEN Release bus, jump to HOME.                     %
%       ELSEIF DK=0 THEN clear TMR, EOBA, IRQ, continue.             %
%       ELSE TMR, wait here.                                         %
%                                                                    %
%       --case  DS=0--                                               %
%DSLOWP:IF FBREQ THEN Release bus, jump to HOME.                      %
%       ELSEIF DK=0 THEN clear TMR, CONTINUE.                        %
%       ELSE TMR, wait here.                                         %
%                                                                    %
%       Wait three cycles to ensure FIFO write, then IRQ, EOBA, jump %
%       home.                                                        %
%                                                                    %
PXFR1LP:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP PXFR1LP;
        ELSEIF I3               %FEOB%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP EOBPIPE;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

DATFP1X:        [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

WAIT7:  IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
```

```
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP WAIT7;
        ELSEIF I3               %FEOB%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP EOBPIPE;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK CDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP PXFR1LP;

EOBPIPE:IF /I0          %/FRSS1%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

        IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I1               %FRDS%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX0]
                JUMP DSLOWP;

%       Case  DS=1                                                      %
DSHIGHP:IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I4               %FRDK%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;
```

```
        ELSE    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX0]
                JUMP DSHIGHP;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

WAIT9:  IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CTRLDEF
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK
MUX0]
                JUMP WAIT9;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG RD MS0 XWT XDK CDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

WAIT10:         IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF /I4      %FRDK=0%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE IRQ DFFDIS
                HOLDB EOBA NEG NRD MS0 XWT XDK XDS XAS
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR          ;AK MUX1]
                JUMP HOME;
        ELSE    [CTRLDEF
                HOLDB NEOBA NEG RD MS0 XWT XDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX0]
                JUMP WAIT10;

%       Case  DS=0                                                      %
DSLOWP:         IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF /I4              %FRDK=0%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
```

```
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX1]
                CONTINUE;
        ELSE    [CHOLD NSRT NDW DATAINFIFO NFDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP DSLOWP;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX1]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX1]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE IRQ DFFEN
                HOLDB EOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE TMR NCCLR STAT8 NDSAK MUX1]
                JUMP HOME;

%-------------------------------------------------------------------------%
%       Routine DATA_FIFO_PIPELINED_READ_2  (200ns transfers)             %
%       --FAST-- --WRITE or READ--                                        %
%-------------------------------------------------------------------------%
%       This routine does a FASTBUS pipeline block transfer read from     %
%       the FASTBUS crate segment to the FSCC's Output Port Data FIFO,    %
%       and also the Processor Data FIFO.                                 %
%                                                                         %
%       Note: DDIR, and SAB must be held high throughout the block        %
%               transfer operation.                                       %
%                                                                         %
%DATFP2:IF FBREQ THEN Release bus, JUMP WAYHOME.                          %
%       ELSEIF WT=1                                                       %
%       THEN Enable FASTBUS xceivers for input, Enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into hold, and loop here.%
%       ELSEIF EOB                                                        %
%       THEN Enable FASTBUS xceivers for input, enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into hold mode, and        %
%               JUMP to EOBPIPE.                                          %
%       ELSE Enable FASTBUS xceivers for input, enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into count, set DS=1,      %
%               and JUMP to DATFP2X.                                      %
%                                                                         %
%PXFR2LP:IF FBREQ THEN Release bus, JUMP WAYHOME.                         %
%       ELSEIF WT=1                                                       %
%       THEN Enable FASTBUS xceivers for input, Enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into hold, and loop here.%
%       ELSEIF EOB                                                        %
%       THEN Enable FASTBUS xceivers for input, enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into hold mode, and        %
%               JUMP to EOBPIPE.                                          %
%       ELSE Enable FASTBUS xceivers for input, enable FIFO's, MS=3,      %
%               RD=1, Place local word counter into count, set DS=1,      %
%               set FCLK, and continue.                                   %
```

```
%                                                                    %
%DATFP2X:Place counter into hold mode, negate FCLK, and              %
%               continue.  These cycles provide the FSCC's           %
%               FASTBUS cycle time (200ns).                          %
%                                                                    %
%       Continue (add 50ns to cycle time).                           %
%                                                                    %
%       Continue (add 50ns to cycle time).                           %
%                                                                    %
%       IF FBREQ THEN Release bus, jump to HOME.                     %
%       ELSEIF WT=1                                                  %
%       THEN counter=hold, keep both FIFO's enabled and loop here.   %
%       ELSEIF EOB                                                   %
%       THEN counter=hold, keep FIFO's enabled and jump to EOBPIPE.  %
%       ELSE counter=decr, set DS=0, set FCLK, keep FIFO's enabled,  %
%               and continue.                                        %
%                                                                    %
%       Place local word counter in hold, negate FCLK, keep processor%
%               and oport FIFO's enabled, and Continue.              %
%               These cycles provide the FSCC's cycle time.          %
%                                                                    %
%       Continue (add 50ns to cycle time).                           %
%                                                                    %
%       Jump to PXFR2LP. (add 50ns to cycle time).                   %
%                                                                    %
PXFR2LP:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP PXFR2LP;
        ELSEIF I3               %FEOB%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP EOBPIPE;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

DATFP2X:        [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
```

```
               HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
               PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
               CONTINUE;

WAIT11:        IF /I6         %FBREQ*%
       THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
               REL
               PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
               JUMP INTF;
       ELSEIF I7              %FRWT%
       THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
               HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
               PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
               JUMP WAIT11;
       ELSEIF I3              %FEOB%
       THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
               HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
               PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
               JUMP EOBPIPE;
       ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
               HOLDB NEOBA NEG RD MS3 XWT XDK CDS XAS
               PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
               CONTINUE;

               [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
               HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
               PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
               CONTINUE;

               [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
               HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
               PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
               CONTINUE;

               [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
               HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
               PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
               JUMP PXFR2LP;

%---------------------------------------------------------------------------%
%       Routine DATA_FIFO_PIPELINED_READ_4  (400ns transfers)              %
%       --FAST-- --READ or WRITE--                                          %
%---------------------------------------------------------------------------%
%       This routine does a FASTBUS pipeline block transfer read from      %
%       the FASTBUS crate segment to the FSCC's Output Port Data FIFO,     %
%       and also the Processor Data FIFO.                                   %
%                                                                           %
%       Note: DDIR, and SAB must be held high throughout the block          %
%               transfer operation.                                         %
%                                                                           %
%DATFP4:IF FBREQ THEN Release bus, jump to WAYHOME.                         %
%       ELSEIF WT=1                                                         %
%       THEN Enable FASTBUS xceivers for input, enable data and             %
%               processor FIFO's, MS=3, RD=1, Place local word counter      %
%               into hold, and loop here.                                   %
%       ELSEIF EOB                                                          %
```

```
%       THEN Enable FASTBUS xceivers for input, enable FIFO's, Place     %
%               local word counter into hold mode, MS=3, RD=1, and JUMP %
%               to EOBPIPE.                                               %
%       ELSE Enable FASTBUS xceivers for input, enable FIFO's, Place     %
%               local word counter into count, MS=3, RD=1, set DS=1,      %
%               and JUMP to DATFP4X.                                      %
%                                                                         %
%PXFR4LP:IF FBREQ THEN Release bus, jump to WAYHOME.                      %
%       ELSEIF WT=1                                                       %
%       THEN Enable FASTBUS xceivers for input, enable data and           %
%               processor FIFO's, MS=3, RD=1, Place local word counter    %
%               into hold, and loop here.                                 %
%       ELSEIF EOB                                                        %
%       THEN Enable FASTBUS xceivers for input, enable FIFO's, Place     %
%               local word counter into hold mode, MS=3, RD=1, and JUMP %
%               to EOBPIPE.                                               %
%       ELSE Enable FASTBUS xceivers for input, enable FIFO's, Place     %
%               local word counter into count, MS=3, RD=1, set DS=1,      %
%               set FCLK, and continue.                                   %
%                                                                         %
%       Place counter into hold mode, negate FCLK, and                    %
%               continue.  These cycles provide the FSCC's                %
%               FASTBUS cycle time (400ns).                               %
%                                                                         %
%       Continue (add 50ns to cycle time).                                %
%                                                                         %
%       Continue (add 50ns to cycle time).                                %
%                                                                         %
%       Continue (add 50ns to cycle time).                                %
%                                                                         %
%       Continue (add 50ns to cycle time).                                %
%                                                                         %
%       Continue (add 50ns to cycle time).                                %
%                                                                         %
%       Continue (add 50ns to cycle time).                                %
%                                                                         %
%       IF FBREQ THEN Release bus, jump to HOME.                          %
%       ELSEIF WT=1                                                       %
%       THEN counter=hold, keep both FIFO's enabled and loop here.        %
%       ELSEIF EOB                                                        %
%       THEN counter=hold, keep FIFO's enabled and jump to EOBPIPE.       %
%       ELSE counter=decr, set DS=0, set FCLK, keep FIFO's enabled,       %
%               and continue.                                             %
%                                                                         %
%       Place local word counter in hold, negate FCLK, keep processor     %
%               and oport FIFO's enabled, and Continue.                   %
%               These cycles provide the FSCC's cycle time.               %
%                                                                         %
%       Continue (add 50ns to cycle time).                                %
%                                                                         %
%       Continue (add 50ns to cycle time).                                %
%                                                                         %
%       Continue (add 50ns to cycle time).                                %
%                                                                         %
%       Continue (add 50ns to cycle time).                                %
%                                                                         %
%       Continue (add 50ns to cycle time).                                %
```

```
%                                                                        %
%       Jump to PXFR4LP.(add 50ns to cycle time).                        %
%                                                                        %
PXFR4LP:IF /I6          %FBREQ*%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7              %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP PXFR4LP;
        ELSEIF I3              %FEOB%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP EOBPIPE;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK SDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

DATFP4X:        [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;


                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
```

```
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

WAIT12:         IF /I6          %FBREQ*%
        THEN    [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFDIS
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTF;
        ELSEIF I7               %FRWT%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP WAIT12;
        ELSEIF I3               %FEOB%
        THEN    [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                JUMP EOBPIPE;
        ELSE    [COUNT NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK CDS XAS
                PFFEN NSR NSCLK FCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
                CONTINUE;

                [CHOLD NSRT NDW DATAINFIFO FDOE FCOE NIRQ DFFEN
                HOLDB NEOBA NEG RD MS3 XWT XDK XDS XAS
                PFFEN NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT8 NDSAK MUX0]
```

```
                JUMP PXFR4LP;

%-------------------------------------------------------------------------%
%       SLAVE_DATA_INPUT                                                  %
%       --SLOW-- --READ--                                                 %
%-------------------------------------------------------------------------%
%       Enable FASTBUS AD line receivers, Clear WT, jump to SLVDINX.      %
%                                                                         %
%       IF FBREQ then release bus, jump to INTS.                          %
%       ELSEIF WT=1, THEN wait here.                                      %
%       ELSEIF DS=1 THEN latch FASTBUS AD lines, set DK, DSACK,           %
%               continue.                                                 %
%       ELSE (DS=0) THEN latch FASTBUS AD lines, clear DK, DSACK,         %
%               continue.                                                 %
%                                                                         %
%       DSACK, Hold latched FASTBUS data, jump WAYHOME.                   %
SLVDINX:IF /I6          %FBREQ%
        THEN    [CTRLDEF
                REL
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP INTS;
        ELSEIF I7       %FRWT%
        THEN    [CDEF NSRT NDW DATAINPROC FDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 NDSAK
MUX0]
                JUMP SLVDINX;
        ELSEIF I1       %FRDS%
        THEN    [CDEF NSRT NDW DATAINLATCH FDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 XWT SDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 DSAK
MUX0]
                CONTINUE;
        ELSE    [CDEF NSRT NDW DATAINLATCH FDOE FCOE NIRQ DFFDIS
                NREQ NEOBA NEG NRD MS0 XWT CDK XDS XAS
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 DSAK
MUX0]
                CONTINUE;

                [CDEF NSRT NDW DATAINHOLD NFDOE FCOE NIRQ DFFDIS
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT13 DSAK MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       Null                                                              %
%       --FAST-- WRITE or READ--                                          %
%-------------------------------------------------------------------------%
NULLX:          [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT12 NDSAK
MUX1]
                JUMP WAYHOME;

%-------------------------------------------------------------------------%
%       End Of Event                                                      %
```

```
%       --SLOW-- --WRITE--                                                  %
%-------------------------------------------------------------------------%
%       --Two dummy words--                                                 %
%       DW (Data FIFO Write), DEOE, CEOE, DFFEN, Set FASTBUS interface      %
%               transceivers for output to FIFO (Data and Processor).       %
%               DSACK, continue.                                            %
%                                                                           %
%       Hold all lines for one more cycle (make sure that the write         %
%               pulse to the FIFO's is long enough).                        %
%                                                                           %
%       Hold DEOE true for one more cycle while DW is false.  This          %
%               provides the "hold" time for the fifo's.                    %
%                                                                           %
%       Set DW, for two cycles to insert another dummy word with DEOE       %
%               false.                                                      %
%                                                                           %
%       Hold DEOE true for one more cycle while DW is false, to             %
%               ensure that the fifo data hold time spec is met.            %
%                                                                           %
EOEX:           [CDEF NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR NSCLK NFCLK DEOE CEOE NTMR NCCLR STAT14 DSAK MUX1]
                CONTINUE;

                [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR NSCLK NFCLK DEOE NCEOE NTMR NCCLR STAT14 NDSAK
MUX1]
                CONTINUE;

                [CDEF NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT14 NDSAK
MUX1]
                CONTINUE;

                [CDEF NSRT DW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT14 NDSAK
MUX1]
                CONTINUE;

                [CDEF NSRT NDW DATAOUT NFDOE FCOE NIRQ DFFEN
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT14 NDSAK
MUX1]
                JUMP HOME;

%-------------------------------------------------------------------------%
%       Interrupt handler                                                   %
%-------------------------------------------------------------------------%
%       Being interrupted means that there was a FASTBUS error, or          %
%       the RB (Reset Bus) line has been driven true.  In either case       %
%       the sequencer's assigned task is simply to abort any operation      %
%       it is currently doing, and return to idle state.                    %
%                                                                           %
%       Returning DSACK to the 68020, is a prcisely timed function          %
```

```
%       which the sequencer must perform.  DSACK is asserted true       %
%       for exactly two sequencer cycles.  Then it is deasserted, and   %
%       the FBxS* lines may not be sampled for three sequencer cycles.  %
%       What this means is that when a routine wants to return DSACK    %
%       and return to idle, it must do three nop type cycles before     %
%       testing either of the two FBxS* lines.                          %
%                                                                       %
%       The interrupt handler is a strange routine since the IRQ line   %
%       is tested at almost every "IF" instruction there is.  If IRQ is %
%       true at any of these places, the IH must make sure that DSACK   %
%       is returned with the proper timing in mind.                     %
%                                                                       %
INTS:           [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 DSAK MUX1]
                CONTINUE;

                [CTRLDEF
                FBDEF
                PFFDIS NSR NSCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 DSAK
MUX1]
                JUMP WAYHOME;

INTF:           [CTRLDEF
                FBDEF
                PFFDIS NSR SCLK NFCLK NDEOE NCEOE NTMR NCCLR STAT15 NDSAK
MUX1]
                JUMP WAYHOME;
```

FF_CTRL.ABL-2/28/90

```
MODULE FB_CTRL_FF
TITLE 'FB_CTRL_FF        VERSION 14-AUG-89 1000 hrs'
FF_CTRL DEVICE 'P22V10';

FSAS    PIN     1;
FCAS    PIN     2;
FSDS    PIN     3;
FCDS    PIN     4;
FSDK    PIN     5;
FCDK    PIN     6;
!SPOE   PIN     7;
SLCOE   PIN     8;
!SDG    PIN     9;
!SR     PIN     10;
DDIR    PIN     11;
!FCOE   PIN     13;
!DG     PIN     14;
!LCOE   PIN     15;
!POE    PIN     16;
!FB2S   PIN     18;
!FB1S   PIN     19;
!PWRITE         PIN     20;
FDDK    PIN     21;
FDDS    PIN     22;
FDAS    PIN     23;

EQUATIONS

FDAS    =       (FSAS # (FDAS & !FCAS)) & FCOE;

FDDS    =       (FSDS # (FDDS & !FCDS)) & FCOE;

FDDK    =       (FSDK # (FDDK & !FCDK)) & FCOE;

"       Equations to prevent bus conflicts on the IOD bus, by preventing
"       any two enables from being true at one time.
"       POE* is only true during a processor write when selected (SPOE &
"       PWRITE & (FB1S # FB2S)), when the 646's (!(SDG & DDIR)), the Data
"       FIFO (!SR), and the Local Counter (!SLCOE), are not driving the IOD
"       bus.  POE* is also true during processor reads (# (SPOE & !PWRITE).
POE     = ((SPOE & PWRITE) & !(SDG & DDIR) & !SR & !SLCOE # (SPOE & !PWRITE))
& (FB1S # FB2S);

"       In similar fashion to POE*, LCOE is only true during processor reads
"       (!(SPOE & PWRITE)), when the 646's !(SDG & DDIR), and the Data FIFO
"       (!SR), are not enabled.
LCOE    =       SLCOE & !(SDG & DDIR) & !SR & !(SPOE & PWRITE);

"       Again, as above, DG* (646's) can only be true during processor reads
"       (!(SPOE & PWRITE)), when the Data FIFO (!SR), and the Local Counter
"       (!SLCOE), are not enabled.  Also, DG* can be true when driving the
"       FASTBUS AD lines (# (SDG & !DDIR)).
DG      = (SDG & DDIR) & !SR & !SLCOE & !(SPOE & PWRITE) # (SDG & !DDIR);

END FB_CTRL_FF
```

FIFO_CTR.ABL-2/28/90

```
MODULE FIFO_CTRL
TITLE 'FIFO_CTRL        VERSION 28-FEB-90 1500 hrs'
FIFO_CTR DEVICE 'P22V10';

CLK     PIN     1;
DFIFOEN         PIN     2;
PFIFOEN PIN     3;
COPYEN  PIN     4;
FRDK    PIN     5;
FW      PIN     6;
SDCPAB  PIN     7;
FRSS1   PIN     8;

DKT     PIN     15;
FRDKZ   PIN     16;
SA      PIN     17;
SB      PIN     18;
SC      PIN     19;
SD      PIN     20;
DCPAB   PIN     21;

!DW     PIN     23;
!PW     PIN     22;

NSTATE  =       [SA,SB]
PSTATE  =       [SC,SD]
S0      =       [0,0];
S1      =       [0,1];
S2      =       [1,0];
S3      =       [1,1];

EQUATIONS

FRDKZ   :=      FRDK;

DKT     :=      ((!FRSS1 & (FRDK != FRDKZ)) # DKT) & !(NSTATE == 0);

WHEN (NSTATE == S0) & ((!FRSS1 & (FRDK != FRDKZ)) # DKT) THEN NSTATE := S1;

WHEN (NSTATE == S1) THEN NSTATE := S2;

WHEN (NSTATE == S2) THEN NSTATE := S3;

WHEN (NSTATE == S3) THEN NSTATE := S0;

DW      :=      ((((NSTATE == S0) & ((!FRSS1 & (FRDK != FRDKZ)) # DKT))
                # (NSTATE == S1)
                # (NSTATE == S2))
                & (DFIFOEN == 1))
                # (FW & DFIFOEN);

WHEN ((PSTATE == S0) & (!COPYEN)) # ((PSTATE == S3) & (!PFIFOEN))
THEN PSTATE := S0;

WHEN ((PSTATE == S1) & (PFIFOEN)) # ((PSTATE == S0) & (COPYEN))
THEN PSTATE := S1;
```

```
WHEN ((PSTATE == S2) & (!PFIFOEN)) # ((PSTATE == S1) & (!PFIFOEN))
THEN PSTATE := S2;

WHEN ((PSTATE == S3) & (PFIFOEN)) # ((PSTATE == S2) & (PFIFOEN))
THEN PSTATE := S3;

PW      :=      ((((NSTATE == S0) & ((!FRSS1 & (FRDK != FRDKZ)) # DKT))
                # (NSTATE == S1)
                # (NSTATE == S2))
                & (PSTATE == S3))
                # (FW & PFIFOEN);

DCPAB   :=      ((NSTATE == S0) & ((DFIFOEN == 1) # (PFIFOEN == 1))
                & ((!FRSS1 & (FRDK != FRDKZ)) # DKT))
                # SDCPAB;

END FIFO_CTRL
```

HEADAV2.ADF-7/11/90

```
module counter
Gustavo I. Cancelo
Fermilab
10/9/89
title header latch & 12 bit event total counter
%       Changed PL equation from registered output to combinatorial.  RK %
%       11-dec-89  1400hrs%
%       --Added state 7 (CLEAR2) to the state machine to lengthen the clear%
%       pulse to the counter.  11-JUL-90  RK%

PART: EPB1400

INPUTS: DS@25, CS@12, A1@13, RW@36, RESET@9, EOE@8, CK@14, CLK@7

OUTPUTS: CARRY@17, CKE@6, WRITE@37, RC@38
         P0@24, P1@23, P2@22, P3@21, P4@20, P5@19, P6@18, %counter%
         DAT0@26, DAT1@27, DAT2@28, DAT3@29, DAT4@32, DAT5@33, DAT6@34,
DAT7@35
         Q3@39, Q2@40, Q1@1, Q0@2, PL@5

NETWORK:
        DS=INP(DS)
        CS=INP(CS)
        A1=INP(A1)
        RW=INP(RW)
        RESET=INP(RESET)
        EOE=INP(EOE)
        CK=INP(CK)
        CLK=INP(CLK)

% data tranceiver bus %
DAT0,DAT1,DAT2,DAT3,DAT4,DAT5,DAT6,DAT7 = BUSX (IBUS,,XOE)

% low order counter. input register #1 LBUSI %
B0,B1,B2,B3,B4,B5,B6,B7 = LBUSI (IBUS,DS,WE1)

% control register. input register #2 LBUSI %
SE0,SE1,RF,WRT,QE2,QE1,QE0,X0 = LBUSI (IBUS,DS,WE2)

% low order counter.  output register #1 %
IBUS = LBUSO(P0,P1,P2,P3,P4,P5,P6,CARRY,,OE1,RE1)

% status register. output register #2 %
IBUS = LBUSO (S0,S1,Q0,Q1,Q2,Q3,RC,WRITE,,OE2,RE2)

%macrocell connection %
S0 = NOCF(S0c)
S1 = NOCF(S1c)

RC,RC = RORF(RCd,CLK,,,)          %software reset macrocell%

Q3,Q3 = RORF(Q3d,CLK,RESET,,)

Q2,Q2 = RORF(Q2d,CLK,RESET,,)
Q1,Q1 = RORF(Q1d,CLK,RESET,,)
Q0,Q0 = RORF(Q0d,CLK,RESET,,)
```

```
%PL,PL = RORF(PLd,CLK,RES,,)       %%enables Preload/Transparent modes for
HEAD1%
PL,PL = COCF(PLd,)     %Changed from registered to combinatorial rk 11-dec-89%

WRITE,WRITE = RORF(WRITEd,CLK,RESET,,WOE)  %control FIFO's write pulse%

CKE = RONF(CKEd,CLK,RES,,)              %Fastbus clock enable for HEAD1%

% counter macrocells %
P0,P0 = TOTF(P0t,RELO,RES,,)
P1,P1 = TOTF(P1t,RELO,RES,,)
P2,P2 = TOTF(P2t,RELO,RES,,)
P3,P3 = TOTF(P3t,RELO,RES,,)
P4,P4 = TOTF(P4t,RELO,RES,,)
P5,P5 = TOTF(P5t,RELO,RES,,)
P6,P6 = TOTF(P6t,RELO,RES,,)

CARRY,CARRY = COCF(CARRYc,)

EQUATIONS:
          RE1 = /DS * /CS * RW * /A1;       % low_counter read enable %
          RE2 = /DS * /CS * RW *  A1;       % status register read enable %
          % RE = RE1 + RE2  read function is activated for either RE1 or RE2%

          WE1 = /RW * /CS * /A1;            % low_counter write enable %
          WE2 = /RW * /CS * A1;             % control register write enable %

          OE1 = /A1 + A1;          % output registers tri state always enabled

          OE2 = /A1 + A1;

          XOE = /CS * RW;

% counter equations %
% INCR mode == S1 * !s0 %
% if (mode=INCR) RELO=ck   else RELO=clk   %

RELO =  Q3 * /Q2 * Q1 * /Q0 * CK + /Q1 *  CLK;

RES = RC;

P0t = Q3 * /Q2 *  Q1 * /Q0 *  S1 * /S0% counter increments if (mode==INCR)  %
 + Q3 * /Q2 * /Q1 *  Q0 *  B0 * /P0    % preload mode !S1 * S0, state mach=9 %
 + Q3 * /Q2 * /Q1 *  Q0 * /B0 *  P0    % toggle if (mode==LOAD & Bi != Pi)   %
 + Q3 *  Q2 * /Q1 *  Q0 *  B0 * /P0    % transparent mode S1 * S0, SM=13     %
 + Q3 *  Q2 * /Q1 *  Q0 * /B0 *  P0;   % toggle if (mode==WRDT & Bi != Pi)   %
P1t = Q3 * /Q2 *  Q1 * /Q0 *  S1 * /S0 *  P0
    + Q3 * /Q2 * /Q1 *  Q0 *  B1 * /P1 + Q3 * /Q2 * /Q1 *  Q0 * /B1 *  P1
    + Q3 *  Q2 * /Q1 *  Q0 *  B1 * /P1 + Q3 *  Q2 * /Q1 *  Q0 * /B1 *  P1;
P2t = Q3 * /Q2 *  Q1 * /Q0 *  S1 * /S0 *  P0 *  P1
    + Q3 * /Q2 * /Q1 *  Q0 *  B2 * /P2 + Q3 * /Q2 * /Q1 *  Q0 * /B2 *  P2
    + Q3 *  Q2 * /Q1 *  Q0 *  B2 * /P2 + Q3 *  Q2 * /Q1 *  Q0 * /B2 *  P2;
P3t = Q3 * /Q2 *  Q1 * /Q0 *  S1 * /S0 *  P2 *  P1 *  P0
    + Q3 * /Q2 * /Q1 *  Q0 *  B3 * /P3 + Q3 * /Q2 * /Q1 *  Q0 * /B3 *  P3
    + Q3 *  Q2 * /Q1 *  Q0 *  B3 * /P3 + Q3 *  Q2 * /Q1 *  Q0 * /B3 *  P3;
P4t = Q3 * /Q2 *  Q1 * /Q0 *  S1 * /S0 *  P3 * P2 * P1 * P0
```

```
    + Q3 * /Q2 * /Q1 *  Q0 *  B4 * /P4 + Q3 * /Q2 * /Q1 *  Q0 * /B4 *  P4
    + Q3 *  Q2 * /Q1 *  Q0 *  B4 * /P4 + Q3 *  Q2 * /Q1 *  Q0 * /B4 *  P4;
P5t = Q3 * /Q2 *  Q1 * /Q0 *  S1 * /S0 *  P4 *  P3 *  P2 *  P1 *  P0
    + Q3 * /Q2 * /Q1 *  Q0 *  B5 * /P5 + Q3 * /Q2 * /Q1 *  Q0 * /B5 *  P5
    + Q3 *  Q2 * /Q1 *  Q0 *  B5 * /P5 + Q3 *  Q2 * /Q1 *  Q0 * /B5 *  P5;
P6t = P6 * /P6;

S0c = SE0;
S1c = SE1;

% state machine, controls the counter's operation  %
% states: Off_line=0, Idle=8, Preload=9, Incr=10, Write=11, %
%         Pass=12, pass_1=13, Clear=14, DS_High=15 %
Q0d =  Q3 * /Q2 * /Q1 * /Q0 * /S1 *  S0 +  Q3 * /Q2 * /Q1 *  Q0 * /S1 *  S0
    +  Q3 *  Q2 * /Q1 * /Q0 * /DS * /A1 * /RW * /CS +  Q3 *  Q2 * /Q1 *  Q0
    +  Q3 *  Q2 *  Q1 *  Q0 +  Q3 * /Q2 *  Q1 * /Q0 *  S1 * /S0 *  EOE
    +  Q3 *  Q2 *  Q1 * /Q0 * /EOE;
Q1d =  Q3 *  Q2 * /Q1 *  Q0 *  DS +  Q3 *  Q2 *  Q1 *  Q0
    +  Q3 * /Q2 * /Q1 * /Q0 *  S1 * /S0 +  Q3 * /Q2 *  Q1 *  Q0 *  EOE
    +  Q3 *  Q2 *  Q1 * /Q0 +  /Q2 *  Q1 * /Q0 *  S1 * /S0
    + /Q3 *  Q2 *  Q1 *  Q0;
Q2d =  Q3 * /Q2 * /Q1 * /Q0 *  S1 *  S0 +  Q3 *  Q2 * /Q1 * /Q0 *  S1 *  S0
    +  Q3 *  Q2 * /Q1 * /Q0 * /DS +  Q3 *  Q2 * /Q1 *  Q0
    +  Q3 * /Q2 *  Q1 *  Q0 *  S1 *  S0 * /EOE +  Q3 * /Q2 *  Q1 *  Q0 *  EOE
    +  Q3 *  Q2 *  Q1 * /Q0;
Q3d = /CS * /RW * /DS * A1
    +  Q3 * /Q2 * /Q1 * /Q0          %Q=8%
    +  Q3 * /Q2 * /Q1 *  Q0          %Q=9%
    +  Q3 * /Q2 *  Q1 * /Q0          %Q=A%
    +  Q3 * /Q2 *  Q1 *  Q0          %Q=B%
    +  Q3 *  Q2 * /Q1 * /Q0          %Q=C%
    +  Q3 *  Q2 * /Q1 *  Q0          %Q=D%
    +  Q3 *  Q2 *  Q1 * /Q0 *  EOE   %Q=E * EOE=1%
    +  Q3 *  Q2 *  Q1 *  Q0          %Q=F%
    + /Q3 *  Q2 *  Q1 *  Q0;         %Q=7%

PLd =  Q3 * /Q2 * /Q1 * /Q0 * /S1 *  S0 +  Q3 * /Q2 * /Q1 * /Q0 *  S1 *  S0
    +  Q3 *  Q2 * /Q1 * /Q0 *  S1 *  S0 +  Q3 *  Q2 * /Q1 *  Q0
    +  Q3 *  Q2 *  Q1 *  Q0 +  Q3 * /Q2 *  Q1 *  Q0 *  S1 *  S0 * /EOE;

WOE =  Q3 + /Q3 *  Q2 *  Q1 *  Q0;

WRITEd = /(Q3 *  Q2 *  Q1 *  Q0 +   Q3 *  Q2 * /Q1 *  Q0 *  DS
       +   Q3 * /Q2 *  Q1 * /Q0 *  S1 * /S0 *  EOE
       +   Q3 * /Q2 *  Q1 *  Q0 *  S1 * /S0);    %CAMBIE EOE POR S1 * !S0%

CLR =  Q3 * /Q2 * /Q1 * Q0;

CARRYc = P5 * P4 * P3 * P2 * P1 * P0;

CKEd =  Q3 * /Q2 * /Q1 * /Q0 *  S1 * /S0 +  Q3 * /Q2 *  Q1 * /Q0 *  S1 * !S0
* /EOE + Q3 * Q2 *  Q1 * /Q0 *  S1 * !S0 + /Q3 *  Q2 *  Q1 *  Q0 *  S1 * /S0;

RCd =   Q3 *  Q2 *  Q1 * /Q0 + /Q3 *  Q2 *  Q1 *  Q0 + RESET + RF;

ZEROc =  A1 * /A1;
END$
```

HEADBV1.ADF-12/9/89

```
module counter
Gustavo I. Cancelo
Fermilab
10/9/89
 Added counter reset (RC) input.      Richard Kwarciany  17-NOV-89
 Removed HAB1 from Header pin equations,  Header outputs are now always
enabled.
  Reversed WE2, with WE1, RE2, with RE1, and OE2 with OE1, in input and
output
  register equations.   9-dec-89  rk.
title header latch & 12 bit event total counter

PART: EPB1400

INPUTS: DS@25, CS@12, A1@13, RW@36, RESET@9, CK@14, CKE@8, PL@5, CARRY@6,
CLK@7,
        RC@15

OUTPUTS: H0@2, H1@1, H2@40, H3@39, H4@38  %header%
         P6@24, P7@23, P8@22, P9@21, P10@20, P11@19, %counter%
         DAT0@26, DAT1@27, DAT2@28, DAT3@29, DAT4@32, DAT5@33, DAT6@34,
DAT7@35

NETWORK:
        DS=INP(DS)
        CS=INP(CS)
        A1=INP(A1)
        RW=INP(RW)
        RESET=INP(RESET)
        CKE=INP(CKE)
        PL=INP(PL)
        CARRY=INP(CARRY)
        CK=INP(CK)
        CLK=INP(CLK)
        RC=INP(RC)      %added 17-NOV-89  RK%

% data tranceiver bus %
DAT0,DAT1,DAT2,DAT3,DAT4,DAT5,DAT6,DAT7 = BUSX (IBUS,,XOE)

% header. input register #2 LBUSI %
XE1,XE2,HE0,HE1,HE2,HE3,HE4,XE3 = LBUSI (IBUS,DS,WE2)   %diferente nombre%

% counter's 6 highest bits. input register #1 LBUSI %
B0,B1,B2,B3,B4,B5,B6,B7 = LBUSI (IBUS,DS,WE1)

% output register #2 %
IBUS = LBUSO (X1,X2,H0,H1,H2,H3,H4,X3,,OE2,RE2)

% output register #1 %
IBUS = LBUSO (P6,P7,P8,P9,P10,P11,N1,N2,,OE1,RE1)

% header outputs%      %Removed HAB1 from OE   9-dec-89  RK%
H0,H0 = COCF (HE0,)
H1,H1 = COCF (HE1,)
H2,H2 = COCF (HE2,)
H3,H3 = COCF (HE3,)
```

```
H4,H4 = COCF (HE4,)

% junk. just to give a name to unused macrocells %
X1 = NOCF(X1c)
X2 = NOCF(X2c)
X3 = NOCF(X3c)

% counter outputs %              %changed RESET to CRESET in counter eqs.%
                                 %17-nov-89  RK%
P6,P6 = TOTF(P6t,RELO,CRESET,,)
P7,P7 = TOTF(P7t,RELO,CRESET,,)
P8,P8 = TOTF(P8t,RELO,CRESET,,)
P9,P9 = TOTF(P9t,RELO,CRESET,,)
P10,P10 = TOTF(P10t,RELO,CRESET,,)
P11,P11 = TOTF(P11t,RELO,CRESET,,)

N1 = NOCF(N2c)
N2 = NOCF(N2c)

RELO = NOCF(RELOc)

CRESET = NOCF(CRESETc)

EQUATIONS:
          RE1 = !DS * !CS * RW * !A1;
          RE2 = !DS * !CS * RW *  A1;
% RE = RE1 + RE2 %

          WE1 = !RW * !CS * !A1;
          WE2 = !RW * !CS * A1;

          OE1 = !DS * !CS * RW * !A1;
          OE2 = !DS * !CS * RW * A1;

          XOE = !CS * RW;

           HAB1 = !CS;                  9-DEC-89 RK%

         CRESETc = RESET + RC;                %added 17-NOV-89  RK%

% counter equations %

RELOc = CLK * !CKE
      + CK  *  CKE;

P6t = !PL * CKE * CARRY
     + B0 * !P6 * PL
     + !B0 * P6 * PL;

P7t = P6 * !PL * CKE * CARRY
    + B1 * !P7 * PL
    + !B1 * P7 * PL;

P8t = P6 * P7 * !PL * CKE * CARRY
    + B2 * !P8 * PL
    + P8 * !B2 * PL;
```

```
P9t = P6 * P7 * P8 * !PL * CKE * CARRY
    + B3 * !P9 * PL
    + P9 * !B3 * PL;

P10t = P9 * P8 * P7 * P6 * !PL * CKE * CARRY
     + B4 * !P10 * PL
     + P10 * !B4 * PL;

P11t = P10 * P9 * P8 * P7 * P6 * !PL * CKE * CARRY
     + B5 * !P11 * PL
     + P11 * !B5 * PL;

X2c = PL * !PL;

X3c = PL * !PL;

N1c = PL * !PL;

N2c = PL * !PL;

X1c = PL * !PL;

END$_
```

INTF3.ADF-9/06/90

```
module oport_interface
Gustavo Cancelo
Fermilab
10/5/89                INTF3
-Changed RES equation.   Rick Kwarciany  16-NOV-89
-Changed clock on START, INHIB, MODE, RES, pins, from RELO (CLK2C), to CS;
-Changed polarity of WE2.  Kwar  16-DEC-89
-Changes from INTF2.... added RPERMIN, and RWAIT to register PERMIN, and
       WAIT signals.  Kwar  4-20-90
-Changed interrupt, and interrupt mask equations.  kwar  9-4-90
-Changed INCOMP interrupt to PERMIN interrupt.  kwar 9-4-90
-Added clear to IRQ line.  kwar 9-6-90

title interface the Oport controller to the CPU system

PART:EPB1400

INPUTS: CS@12,RW@36,DS@25,A0@13,CLK@14,CLK2@7,RESET@9
        PERMIN@8,PRMOUT@40,STPACK@18,ERROR@17,SM3@21,SM2@22,SM1@23,SM0@24
        WAIT@5

OUTPUTS: DB0@26,DB1@27,DB2@28,DB3@29,DB4@32,DB5@33,DB6@34,DB7@35
         MODE2@2,MODE0@4,MODE1@3,RES@1,STPMK@39,PERMINMK@38,ERRMK@37
         IRQ@16,RPERMIN@19,RWAIT@6,IRQS@20

NETWORK:

A0=INP(A0)
CLK=INP(CLK)
CLK2=INP(CLK2)
CS=INP(CS)
RW=INP(RW)
DS=INP(DS)
RESET=INP(RESET)

SM3=INP(SM3) %SM3,SM2,SM1,SM0: OPORT's state machine status%
SM2=INP(SM2)
SM1=INP(SM1)
SM0=INP(SM0)
PERMIN=INP(PERMIN)
PRMOUT=INP(PRMOUT)
STPACK=INP(STPACK)
ERROR=INP(ERROR)
WAIT=INP(WAIT)

MODE0,MODE0=RORF(MODE0d,RELO,RESET,,)
MODE1,MODE1=RORF(MODE1d,RELO,RESET,,)
MODE2,MODE2=RORF(MODE2d,RELO,RESET,,)
RES,RES=RORF(RESd,CLK2,,,)

IRQ,IRQ=RORF(IRQd,CLK,,,)
STPMK,STPMK=RORF(STPMKc,RELO,RESET,,)
PERMINMK,PERMINMK=RORF(PERMINMKc,RELO,RESET,,)
ERRMK,ERRMK=RORF(ERRMKc,RELO,RESET,,)
IRQS,IRQS=COCF(IRQSd,)
```

```
RPERMIN,RPERMIN=RORF(PERMINd,CLK,RESET,,)
RWAIT,RWAIT=RORF(WAITd,CLK2,RESET,,)

%high add/data byte tranceiver w/tri_state enable%
DB0,DB1,DB2,DB3,DB4,DB5,DB6,DB7 = BUSX (IBUS,,XOE)

%high address byte input register w/tri_state enable 2%
MODE01,MODE11,MODE21,RES1,OR10,STPMK1,PERMINMK1,ERRMK1 = LBUSI (IBUS,,WE2)

%high data byte output register 1 %
IBUS = LBUSO (SM0,SM1,SM2,SM3,IRQS,RPERMIN,STPACK,ERROR,,OLE1,RE1)

%high data byte output register 2 %
IBUS = LBUSO (MODE0,MODE1,MODE2,RES,PRMOUT,STPMK,PERMINMK,ERRMK,,OLE2,RE2)


EQUATIONS:

          RELO = !DS * !CS * !RW * A0 * CLK2;

          RE1 = !DS * !CS * RW * !A0;
          RE2 = !DS * !CS * RW *  A0;
          % RE = RE1 + RE2 %

          WE2 = (!RW * !CS * A0);     %Changed polarity 16-dec-89 rk%

           OLE1 = CLK;
           OLE2 = CLK2;

          XOE = !CS * RW;


MODE0d = MODE01;
MODE1d = MODE11;
MODE2d = MODE21;

!RESd = RESET + RES1;
!IRQd = (STPACK * STPMK)
       + (!PERMIN * PERMINMK)
       + (ERROR * ERRMK)
       + (!IRQ * RES * !(!DS * !CS * RW * !A0));

STPMKc = STPMK1;
PERMINMKc = PERMINMK1;
ERRMKc = ERRMK1;
IRQSd = IRQ;

WAITd = WAIT;
PERMINd = PERMIN;

END$
```

LC_HIGH.ABL-2/5/90

```
MODULE LOCAL_COUNTER_HIGH
TITLE 'LOCAL_COUNTER_HIGH  PC2 VERSION 5-FEB-90 0900 hrs'
LC_HIGH DEVICE 'P22V10';

CLK     PIN     1;
LC0     PIN     2;
LC1     PIN     3;

!CIN    PIN     22;
!TC     PIN     23;

!OE     PIN     13;

I0      PIN     4;
I1      PIN     5;
I2      PIN     6;
I3      PIN     7;
I4      PIN     8;
I5      PIN     9;

Q0      PIN     16;
Q1      PIN     17;
Q2      PIN     18;
Q3      PIN     19;
Q4      PIN     20;
Q5      PIN     21;

DATA    =       [Q5..Q0];
INPUT   =       [I5..I0];
MODE    =       [LC1,LC0];
HOLD    =       [0,.X.];
WRITE   =       [1,0];
COUNT   =       [1,1];
MATCH   =       0;

EQUATIONS


ENABLE DATA     =       OE;

WHEN (MODE == COUNT) & (CIN == 1) THEN DATA:=DATA-1;

WHEN (MODE == WRITE) THEN DATA:=INPUT;

WHEN (MODE == HOLD) # ((MODE == COUNT) & (CIN == 0)) THEN DATA:=DATA;

TC      =       (DATA == MATCH);

ENABLE CIN=0;

END LOCAL_COUNTER_HIGH
```

LC_LOW.ABL-2/5/90

```
MODULE LOCAL_COUNTER_LOW
TITLE 'LOCAL_COUNTER_LOW   PC2 VERSION 5-FEB-90  0900hrs'
LC_LOW DEVICE 'P22V10';

CLK     PIN     1;
LC0     PIN     2;
LC1     PIN     3;

!COUT   PIN     22;
!TC     PIN     23;

!OE     PIN     13;

I0      PIN     4;
I1      PIN     5;
I2      PIN     6;
I3      PIN     7;
I4      PIN     8;
I5      PIN     9;

Q0      PIN     16;
Q1      PIN     17;
Q2      PIN     18;
Q3      PIN     19;
Q4      PIN     20;
Q5      PIN     21;

DATA    =       [Q5..Q0];
INPUT   =       [I5..I0];
MODE    =       [LC1,LC0];
HOLD    =       [0,.X.];
WRITE   =       [1,0];
COUNT   =       [1,1];
MATCH   =       0;

EQUATIONS


ENABLE DATA     =       OE;

WHEN (MODE == COUNT) THEN DATA:=DATA-1;

WHEN (MODE == WRITE) THEN DATA:=INPUT;

WHEN (MODE == HOLD) THEN DATA:=DATA;

TC      =       (DATA == MATCH);

COUT    =       (MODE == COUNT) & (DATA == 0);

END LOCAL_COUNTER_LOW
```

OPOV7.02.ASM-6/4/90

```
OPORT State Machine SAM format file
VDAS controller
V7.02  Monday 4-JUN-90        1530hrs
Change from V3: Keep oe true during reset to prevent 485 drivers from
              oscillating on cable.
Change from V4: Change Start, and Step inputs to Mode0, and Mode1.
              Four possible modes are now:
              M1      M0      Mode
              0       0       Hold
              0       1       Start
              1       0       Force_Start
              1       1       CPU
Change from V5:Reverse polarity of Strobe output, and remove Strobe, and
              OE from RESET state.  Now a negative going ~25 ns pulse
              will be generated by a one shot on the AUX card from the
              rising edge of the the VDAS controller Strobe output.
              Change made 21-Mar-90  by Richard Kwarciany.
Change from V6:Inverted PERMIN/PERMOUT signals as per E771's request, and
              corrected error in macros for vdas_oe line.  24-Mar-90  RK
Change from V6.01:Change polarity of WAIT input to match VDAS's /WAIT signal.
              10-APR-90, RK.
Change from V6.02: Change INHIB line to MODE2.  Encode the three mode lines
              to allow for two new modes.  The new modes are; EMPO, and PO.
              (Event With Manual Permit Out, and Permit Out).  The Eight
              possible modes are now:
              Mode            M2      M1      M0
              Hold            0       0       0
              Event           0       0       1
              Force_Event     0       1       0
              CPU (Step)      0       1       1
              Inhibit         0       1       0
              EMPO            1       0       1
              PO              1       1       0
              Reserved        1       1       1
Change from V7.00: Added two new states; Active2, and Wait3. Active2 is the
              same as Active, except that EOE is not tested. This state will
              only be used on the first data cycle, where EOE is not valid.
              Wait3 is the same as Wait, except that it returns to Active2
              instead of Active.
Change from V7.01: Made changes to fix bugs in Step mode operation.

PART: EPS448

INPUTS: wait@2, eoe@3, def@4, cef@5, permin@9, mode2@10, mode1@11, mode0@12

OUTPUTS: Q3@25, Q2@24, Q1@23, Q0@22,
       lck@13, strobe@20, /cfifor@17, /dfifor@18,
       /loe@14, error@26, stpack@27, /prmout@28,
       /lclr@16, vdas_oe@1

MACROS:  % outNo   state    outputs %
          RESET = "0000    0011100100"      %LCLR*%
           OUT0  = "0000    0011000101"      %LCLR*,LOE*,VDAS_OE%
           OUT1  = "0001    0001000111"      %CFR*,LOE*,VDAS_OE%
           OUT2  = "0010    0011000101"      %LCLR*,LOE*,VDAS_OE%
           OUT3  = "0011    0001000111"      %CFR*,LOE*,VDAS_OE%
```

```
            OUT4  = "0100    1010000111"      %LCK,DFR*,LOE*,VDAS_OE%
            OUT5  = "0101    0011001011"      %STPACK,LOE*,VDAS_OE,PERMOUT*%
            OUT6  = "0110    0001000111"      %CFR*,LOE*,VDAS_OE%
            OUT7  = "0111    0011110100"      %ERROR,LCLR*%
            OUT8  = "1000    0011100110"      %no outputs%
            OUT9A = "1001    0011000101"      %LCLR*,LOE*,VDAS_OE%
            OUT9B = "1001    0010000001"      %DFR*,PERMOUT*,LOE*,VDAS_OE%
            OUT9C = "1001    0011000001"      %PERMOUT*,LCLR*,LOE*,VDAS_OE%
            OUT9D = "1001    0010100100"      %DFR*,LCLR*%
            OUT10 = "1010    1011000111"      %LCK,LOE*,VDAS_OE%
            OUT11 = "1011    1011000111"      %LCK,LOE*,VDAS_OE%
            OUT12 = "1100    0110000111"      %STROBE,DFR*,LOE*,VDAS_OE%
            OUT13 = "1101    0011001011"      %STPACK,LOE*,VDAS_OE,PERMOUT*%
            OUT14 = "1110    1011000111"      %LCK,LOE*,VDAS_OE%
            OUT15 = "1111    1011000111"      %LCK,LOE*,VDAS_OE%

EQUATIONS:
        MODE_0  =       /mode2  *  /mode1  *  /mode0;
        %hold%
        MODE_1  =       /mode2  *  /mode1  *   mode0;
        %Event%
        MODE_2  =       /mode2  *   mode1  *  /mode0;
        %Force_Event%
        MODE_3  =       /mode2  *   mode1  *   mode0;
        %CPU%
        MODE_4  =        mode2  *  /mode1  *  /mode0;
        %Inhibit%
        MODE_5  =        mode2  *  /mode1  *   mode0;
        %EMPO%
        MODE_6  =        mode2  *   mode1  *  /mode0;
        %PO%
        MODE_7  =        mode2  *   mode1  *   mode0;
        %Reserved%

PROGRAM:
%************************************************************************%
%     State = IDLE, CK=L, strobe=H, cfifor=H, dfifor=H, oe=H, error=0,   %
%     stpack=L,  prmout=L                                                %
%************************************************************************%
        [RESET] CONTINUE;

RES:    IF (MODE_1 + MODE_3 + MODE_5)          %Event + CPU + EMPO%
        THEN [RESET] CONTINUE;
        ELSEIF MODE_2 THEN [OUT0] JUMP IDLES;
        %Force_Event%
        ELSEIF MODE_4 THEN [OUT8] JUMP INHIBS;
        %inhibit%
        ELSE [RESET] JUMP RES;

PINWAIT1:IF /permin + MODE_2                   %permin* + Force_Event%
        THEN [OUT0] JUMP IDLES;
        ELSEIF MODE_4 THEN [OUT8] JUMP INHIBS;
        %inhibit%
        ELSE [RESET] JUMP PINWAIT1;
```

```
%*************************************************************************%
%    State = IDLE, CK=L, strobe=H, cfifor=H, dfifor=H, oe=H, error=0,      %
%    stpack=L,  prmout=L                                                  %
%    IF inhib THEN State=INHIB (8)                                        %
%    ELSEIF ((Event + Force_Event + CPU + EMPO) * cef) State=SETUP (2)    %
%    ELSE state=IDLES (0)                                                 %
%*************************************************************************%
IDLES:  IF MODE_4                                  %Inhibit%
        THEN [OUT8] JUMP INHIBS;
        ELSEIF (MODE_1 + MODE_2 + MODE_3 + MODE_5) * cef
        THEN [OUT2] JUMP SETUPS;
        ELSE [OUT0] JUMP IDLES;

%*************************************************************************%
%    State = INHIB, CK=L, strobe=H, cfifor=H, dfifor=H, oe=H, error=0,     %
%    stpack=L,  prmout=L                                                  %
%*************************************************************************%
INHIBS: IF MODE_4 THEN [OUT8] JUMP INHIBS;   %Inhibit%
        ELSE [OUT0] JUMP RES;

%*************************************************************************%
%    State = SETUP, CK=L, strobe=H, cfifor=H, dfifor=H, oe=H, error=0,     %
%    stpack=L,  prmout=L                                                  %
%    IF /wait State=ERROR (7), error go to H                              %
%    ELSE State=CONTS (6), cfifor=L, oe=L                                 %
%*************************************************************************%
SETUPS: IF (/wait) THEN [OUT7] JUMP ERRORS;
        ELSEIF MODE_3 THEN [OUT3] JUMP STEPS;
        %Step%
        ELSE [OUT6] JUMP CONTS;

%*************************************************************************%
%    State = CONTS, CK=L, strobe=H, cfifor=L, dfifor=H, oe=L, error=0,     %
%    stpack=L,  prmout=L                                                  %
%    IF cef State=ACTIVE (14), ck go to H, strobe=L, cfifor=H             %
%*************************************************************************%
CONTS:  IF MODE_0 THEN [OUT4] JUMP HOLD1S;    %hold%
       ELSE [OUT14] JUMP ACTIVE2S;

%*************************************************************************%
%    State = ACTIVES, CK=H, strobe=L, cfifor=H, dfifor=H, oe=L, error=0,   %
%    stpack=L,  prmout=L                                                  %
%    IF /WAIT THEN WAITS                                                  %
%    IF eoe State=EOES (9), ck go to H, oe=H,                             %
%    ELSEIF def THEN State=DATA (12), ck=L, strobe=H, dfifor=H,           %
%    ELSE State = ACTIVES                                                 %
%*************************************************************************%
ACTIVES: IF (/wait) THEN [OUT15] JUMP WAITS;
         ELSEIF (eoe) THEN [OUT9A] JUMP CHKMODE;
         ELSEIF (def) THEN [OUT12] JUMP DATAS;
         ELSE [OUT14] JUMP ACTIVES;

ACTIVE2S:IF (/wait) THEN [OUT15] JUMP WAIT3S;
         ELSEIF (def) THEN [OUT12] JUMP DATAS;
         ELSE [OUT14] JUMP ACTIVE2S;
```

```
%*************************************************************************%
%    State = DATAS, CK=L, strobe=H, cfifor=H, dfifor=H, oe=L, error=0,     %
%    stpack=L,  prmout=L                                                  %
%    IF /start THEN HOLD1S.                                               %
%    ELSE State=ACTIVE (14), ck go to H, strobe=L, cfifor=H               %
%*************************************************************************%
DATAS: IF MODE_0 THEN [OUT4] JUMP HOLD1S;    %hold%
       ELSE [OUT14] JUMP ACTIVES;

%*************************************************************************%
%    State = WAITS, CK=H, strobe=L, cfifor=H, dfifor=H, oe=L, error=0,     %
%    stpack=L,  prmout=L                                                  %
%    IF cef State=ACTIVE (14), same outputs                               %
%*************************************************************************%
WAITS: IF (wait) THEN [OUT14] JUMP ACTIVES;  %/wait*%
       ELSE [OUT15] JUMP WAITS;

WAIT3S:IF (wait) THEN [OUT14] JUMP ACTIVE2S; %/wait*%
       ELSE [OUT15] JUMP WAIT3S;

%*************************************************************************%
%    State = HOLD1S, CK=H, strobe=L, cfifor=H, dfifor=H, oe=L, error=0,    %
%    stpack=L,  prmout=L                                                  %
%    IF inhib State=INHIBS (8), ck go to L, strobe=H, oe=H                %
%    ELSEIF (step * !start) State=IDLE, ck goto L, strobe=H, oe=H         %
%    ELSEIF (start) State=ACTIVE, same outputs                            %
%*************************************************************************%
HOLD1S: IF MODE_4 THEN [OUT8] JUMP INHIBS;
        ELSEIF MODE_3 THEN [OUT0] JUMP IDLES;
        %Step%
        ELSEIF MODE_1 + MODE_2 + MODE_5
        %Event + Force_Event + EMPO%
        THEN [OUT12] JUMP DATAS;
        ELSE [OUT4] JUMP HOLD1S;

%*************************************************************************%
%    State = STEPS, CK=L, strobe=H, cfifor=L, dfifor=H, oe=L, error=0,     %
%    stpack=L,  prmout=L                                                  %
%    IF /wait State=WAIT2S (1), same outputs                              %
%    ELSEIF (!cef) State=STPACK1 cfifor=H, oe=H, stpack=H                 %
%    ELSE State=IDLE, stpack goto H, cfifor=H, oe=H                       %
%*************************************************************************%
STEPS: IF (/wait) THEN [OUT1] JUMP WAIT2S;
       ELSEIF (!cef) THEN [OUT5] JUMP STPACK1S;
       ELSE [OUT11] JUMP SDATAS;

%*************************************************************************%
%    State = SDATAS, CK=H, strobe=L, cfifor=H, dfifor=H, oe=L, error=0,    %
%    stpack=L,  prmout=L                                                  %
%    IF (!step) State=HOLD2S, same outputs                                %
%    ELSEIF (cef) State=STEPS (3), ck go to L, strobe=H, cfifor=L         %
%*************************************************************************%
SDATAS: IF MODE_3 THEN [OUT3] JUMP STEPS;    %step%
        ELSE [OUT10] JUMP HOLD2S;
```

```
%*************************************************************************%
%    State = HOLD2S, CK=H, strobe=L, cfifor=H, dfifor=H, oe=L, error=0,    %
%    stpack=L,  prmout=L                                                  %
%    IF inhib State=INHIBS (8), ck go to L, strobe=H, oe=H                %
%    ELSEIF (/step * /start) State=IDLES, ck goto L, strobe=H, oe=H       %
%    ELSEIF (start * step) State=SDATAS, same outputs                     %
%*************************************************************************%
HOLD2S: IF MODE_4 THEN [OUT8] JUMP INHIBS;   %Inhibit%
        ELSEIF MODE_1 + MODE_2 + MODE_5
        %Event + Force_Event + EMPO%
        THEN [OUT0] JUMP IDLES;
        ELSEIF MODE_3 THEN [OUT11] JUMP SDATAS;
        %step%
        ELSE [OUT10] JUMP HOLD2S;

%*************************************************************************%
%    State = WAIT2S, CK=L, strobe=H, cfifor=L, dfifor=H, oe=L, error=0,    %
%    stpack=L,  prmout=L                                                  %
%    IF cef State=STEP (3), same outputs                                  %
%*************************************************************************%
WAIT2S: IF (wait) THEN [OUT3] JUMP STEPS;
        ELSE [OUT1] JUMP WAIT2S;

%*************************************************************************%
%     EOE goto IDLE                                                       %
%*************************************************************************%
EOES:   [OUT9C] CONTINUE;

PINWAIT2:IF (/permin) THEN [OUT0] JUMP IDLES;
        %/permin*%
        ELSEIF MODE_0 THEN [RESET] JUMP RES;  %hold%
        ELSEIF MODE_4 THEN [OUT8] JUMP INHIBS;
        %inhibit%
        ELSE [RESET] JUMP PINWAIT2;

%*************************************************************************%
%     STPACK1S goto STPACK2                                               %
%*************************************************************************%
STPACK1S:  [OUT13] JUMP STPACK2S;

%*************************************************************************%
%     STPACK2S goto PINWAIT2                                              %
%*************************************************************************%
STPACK2S:   [OUT0] JUMP PINWAIT2;
```

```
%*************************************************************************%
%     State = ERRORS, CK=L, strobe=H, cfifor=H, dfifor=H, oe=H, error=1,   %
%     stpack=L,  prmout=L                                                 %
%     stay in ERRORS until hardware Reset                                 %
%*************************************************************************%
ERRORS: [OUT7] JUMP ERRORS;

CHKMODE:IF MODE_6 + MODE_1 + MODE_2          %PO + Event + Force_Event%
        THEN [OUT9B] JUMP EOES;
        ELSEIF MODE_0 THEN [OUT9D] JUMP CLEANUP;
        %hold%
        ELSEIF MODE_4 THEN [OUT9D] JUMP INHIBS;
               %Inhibit%
        ELSE [OUT9A] JUMP CHKMODE;

CLEANUP:[RESET] JUMP RES;

END$
```

STIMER.ABL-10/13/89

```
MODULE SHORT_TIMER
TITLE 'SHORT_TIMER        VERSION 21-AUG-89 1430 hrs'
STIMER DEVICE 'P22V10';

CLK     PIN     1;
FRBH    PIN     3;
FRRB    PIN     4;
FDRB    PIN     5;
STEN    PIN     8;
TIMER   PIN     9;
FRWT    PIN     10;
FCLERR  PIN     13;

T0      PIN     14;
T1      PIN     15;
T2      PIN     16;
T3      PIN     17;
T4      PIN     18;
!STO    PIN     20;
R0      PIN     21;
R1      PIN     22;
!FRESET         PIN     23;

COUNT   =       [T4..T0];
RCOUNT  =       [R1..R0];
RINIT   =       [1,0];
CINIT   =       [1,1,1,1,1];

EQUATIONS

WHEN (FCLERR == 1) # (FRBH == 1) # (FRRB == 0) # (FDRB == 1) THEN
RCOUNT:=RINIT;
                                                ELSE RCOUNT:=RCOUNT-1;

WHEN (TIMER == 0) # (FCLERR == 1) # (STEN == 0) # (FRWT == 1) THEN
COUNT:=CINIT;
                                                ELSE COUNT:=COUNT-1;

STO     :=      ((COUNT == 0) # (STO == 1)) & (FCLERR == 0);

FRESET  :=      ((RCOUNT == 0) # (FRESET ==1)) & (FCLERR == 0);

END SHORT_TIMER
```

STROBE.ABL-9/28/89

```
module ds_generator
title 'Fastbus Readout Controller, data strobe generator
Gustavo Cancelo, Fermilab, 4/3/89'

"Added DS to LDS and UDS 19-SEP-89 RK
"Changed LDS, and UDS from pins 16 and 17 to pins 15 and 16.  27-SEP-89 RK

        strobe   device    'P22V10';

AD0,AD1,SIZ0,SIZ1,RW,DS      pin     8,7,6,5,4,3;
LDS,UDS                      pin     15,16;
WLDS,WUDS                    pin     22,23;
WUUDS,WUMDS,WLMDS,WLLDS      pin     21,20,19,18;
RDS                          pin     14;

L,H,X,Z = 0,1,.X.,.Z.;

TRANF_TYPE = [SIZ1,SIZ0];
LONG = ^b00;
TRI_BY = ^b11;
WORD = ^b10;
BYTE = ^b01;

AD1_0 = [AD1,AD0];

equations
         !UDS = !AD0 * !DS;
         !LDS =  AD0 * !DS # !SIZ0 * !DS # SIZ1 * !DS;

         !WUDS = !AD0 & !RW;
         !WLDS =  (AD0 # !SIZ0 # SIZ1) & !RW;

         !RDS = RW;


         !WUUDS = !AD0 & !AD1 & !RW;

         !WUMDS = (!AD1 & !SIZ0
                #  AD0 & !AD1
                # !AD1 & SIZ1) & !RW;

         !WLMDS = (!AD0 & AD1
               # !AD1 & !SIZ0 & !SIZ1
               # !AD1 & SIZ0 & SIZ1
               #  AD0 & !AD1 & !SIZ0) & !RW;

         !WLLDS = (AD0 & SIZ0 & SIZ1
               # !SIZ0 & !SIZ1
               #  AD0 & AD1
               #  AD1 & SIZ1) & !RW;

test_vectors

([TRANF_TYPE,AD1,AD0, RW] -> [WUUDS,WUMDS,WLMDS,WLLDS])
 [   LONG,     0, 0,  L ] -> [  L,    L,    L,    L  ];
 [   TRI_BY,   0, 0,  L ] -> [  L,    L,    L,    H  ];
```

```
  [   WORD,      0, 0,   L ] -> [  L,    L,    H,    H  ];
  [   BYTE,      0, 0,   L ] -> [  L,    H,    H,    H  ];
  [   LONG,      0, 1,   L ] -> [  H,    L,    L,    L  ];
  [   TRI_BY,    0, 1,   L ] -> [  H,    L,    L,    L  ];
  [   WORD,      0, 1,   L ] -> [  H,    L,    L,    H  ];
  [   BYTE,      0, 1,   L ] -> [  H,    L,    H,    H  ];

  [   LONG,      1, 1,   H ] -> [  H,    H,    H,    H  ];
  [   TRI_BY,    1, 1,   H ] -> [  H,    H,    H,    H  ];
  [   WORD,      1, 1,   H ] -> [  H,    H,    H,    H  ];
  [   BYTE,      1, 1,   H ] -> [  H,    H,    H,    H  ];
  [   LONG,      1, 0,   H ] -> [  H,    H,    H,    H  ];
  [   TRI_BY,    1, 0,   H ] -> [  H,    H,    H,    H  ];
  [   WORD,      1, 0,   H ] -> [  H,    H,    H,    H  ];
  [   BYTE,      1, 0,   H ] -> [  H,    H,    H,    H  ];


test_vectors
([TRANF_TYPE, DS,AD1,AD0, RW] -> [WUDS,WLDS,UDS,LDS])
  [  LONG,      0, 0,   0,   L ] -> [ L,    L,    L,   L];
  [  WORD,      0, 0,   1,   H ] -> [ H,    H,    H,   L];
  [  BYTE,      0, 1,   0,   H ] -> [ H,    H,    L,   H];
  [  BYTE,      0, 1,   1,   H ] -> [ H,    H,    H,   L];
  [  BYTE,      0, 0,   0,   H ] -> [ H,    H,    L,   H];
  [  BYTE,      0, 0,   1,   H ] -> [ H,    H,    H,   L];


END_
```

Appendix D - FSCC Documentation

Fermilab Drawing Numbers

FNAL #	Title	Description
0882-MB-199070	Mounting Bracket	FASTBUS Module Front Panel Mount
0880.000-ED-215714	FSCC/VDAS Interface	E771 FSCC/VDAS interface schematic
0880.000-ED-269065	FSCC/VDAS Interface	E791 FSCC/VDAS interface schematic
0880.000-AC-269129	FSCC Layer 1	Trace Layer 1
0880.000-AC-269130	FSCC Layer 2	-2.0V & -5.2V Layer 2
0880.000-AC-269131	FSCC Layer 3	Ground Layer 3
0880.000-AC-269132	FSCC Layer 4	+5.0V Layer 4
0880.000-AC-269133	FSCC Layer 5	Trace Layer 5
0880.000-AC-269134	FSCC Top Silk	Top Silkscreen Photo
0880.000-AC-269135	FSCC Bot Silk	Bottom Silkscreen Photo
0880.000-AC-269136	FSCC Solder Mask	Top & Bot. Solder Mask
0880.000-AC-269137	FSCC Front Panel	Front Panel Silkscreen Photo
0880.000-MD-269138	FSCC Assembly	Assembly drawing
0880.000-MD-269139	FSCC Front Panel	Front Panel Mechanical Drawing
0880.000-MD-269140	FSCC Mechanical	Board Dimensions & Pads
0880.000-MD-269141	FSCC OPORT Test Board	OPORT Test Board Schematic
0880.000-MD-269142	FSCC FPP Test Board	Front Panel Port Test Board Schematic
	FSCC FPP Top Layer	Front Panel Port Test Board Top Layer
	FSCC FPP Bottom Layer	Front Panel Port Test Board Bottom Layer
0880.000-MD-269143	FSCC Schematic	11 Page Schematic and Block Diagram

Fermilab Documents

FASTBUS Smart Crate Controller - PC3, Design Specification
Fermilab Computing Division
Mark Bernett - Online and Data Acquisition Software Groups
Mark Bowden, Rick Kwarciany, John Urish - Data Acquisition Electronics Group
Fermilab Physics Department
Gustavo Cancelo

Diagnostics for the FASTBUS Smart Crate Controller - PN417
Fermilab Computing Division
Mark Bernett, Dave Slimmer - Online and Data Acquisition Software Groups
Fermilab Computing Division
Rick Kwarciany, John Urish - Data Acquisition Electronics Group

Release Notes for SCG68K V2.3
PN 376
David M. Berg, Bryan MacKinnon - Fermilab Computing Division
Online Systems Software Group

SCG68K User's Guide and Reference
PN369
Peter Heinicke, David Berg, Bryan MacKinnon, Tom Nicinski, Gene Oleynik -
Fermilab Computing Division, Online Systems and Data Acquisition Software Groups

Serial Port Driver for the PAN-DA pSOS Environment
PN379.2

Bryan MacKinnon - Fermilab Computing Division, Data Acquisition Software Group
FSCC Parts List, Front Panel Port Test Board Parts List, OPORT Parts List and FSCC/VDAS interface Parts List
The parts lists are implemented in a Microsoft EXCEL spreadsheet program to allow easy updating and sorting. They are available from John Urish at Fermilab as a printed copy or on a Macintosh floppy disk.

Non-Fermilab Documents

pSOS-68K, pROBE-68K, pRISM-68K
Software Components Group, Inc.
4655 Old Ironsides Drive
Santa Clara, CA 95054

IEEE Standard FASTBUS
IEEE 960
Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, New York - 10017

68020 32-Bit Microprocessor Manual
Motorola Literature Distribution
P.O. Box 20912
Phoenix, AZ 85036

FASTBUS Standard Routines
DOE/ER 0325
National Technical Information Service, U.S. Dept. of Commerce
Springfield, Virginia 22161

FSCC Software

FSCC System Software
Software Components Group, Inc.
4655 Old Ironsides Drive
Santa Clara, CA 95054
(Refer to "SCG68K User's Guide and Reference" or contact David Berg at Fermilab)

FSCC Diagnostic Software
The diagnostic software is supplied by Fermilab and described in "Diagnostics for the FASTBUS Smart Crate Controller". The source code or compiled code may be obtained by contacting Dave Slimmer at Fermilab.

FSCC Microcode

PAL/PLD Equations for FSCC
The PAL/PLD sources are listed in Appendix C of this document. The source listings and JEDEC files may be obtained on a Macintosh or IBM format floppy disk from John Urish at Fermilab.

FPORT Equations for FSCC
The EPS448 sources are listed in Appendix C of this document. The source listings and JEDEC files may be obtained on a Macintosh or IBM format floppy disk from John Urish at Fermilab.

PAL/PLD Equations for OPORT Test Board
The source listings and JEDEC files may be obtained on a Macintosh or IBM format floppy disk from John Urish at Fermilab.

Appendix E - FSCC Front Panel

FASTBUS Slave: Yellow LED

FASTBUS Master: Green LED

Front Panel Output (20 pin IDC):

Pin#	RS422 Funciton
1	Front Panel 0 +
2	Front Panel 0 -
3	Front Panel 1 +
4	Front Panel 1 -
5	Front Panel 2 +
6	Front Panel 2 -
7	Front Panel 3 +
8	Front Panel 3 -
9-20	Reserved

Trigger Input (20 pin IDC):

Pin#	RS422 Funciton
1-10	Reserved
11	Trigger Strobe +
12	Trigger Strobe -
13	Trigger ID0 +
14	Trigger ID0 -
15	Trigger ID1 +
16	Trigger ID1 -
17	Trigger ID2 +
18	Trigger ID2 -
19	Trigger ID3 +
20	Trigger ID3 -

Reset Pushbutton: Hard processor reset.

Remote Reset:: Hard processor reset. This is an active low TTL input. Shorting the connetor or applying a TTL low will cause a reset. This input may be Daisy-Chained.

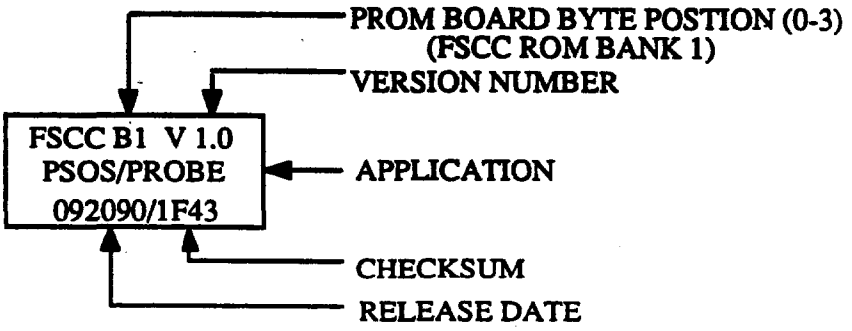
Perm_In/Perm_Out: Serial "daisy-chain" signals for FSCC auxiliary port bussing, LEMO connectors.

2 Serial Ports: RS232 signal levels (4 pin LEMO connectors) one for Host connection and one for Terminal connection.
Terminal port
Host port (Null Modem)

Ethernet Port: Cheapernet signal levels, Isolated BNC Connector.

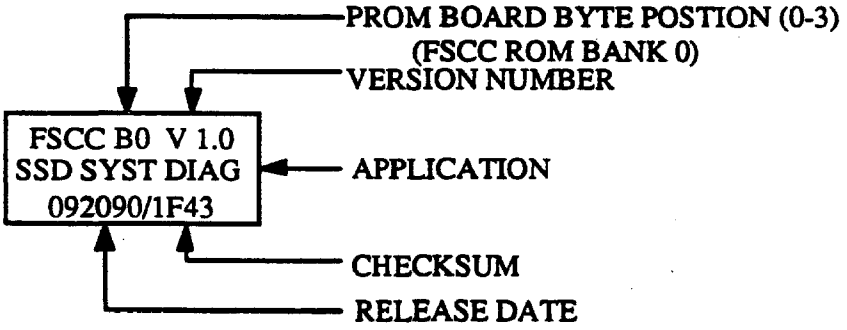
Appendix F - FSCC EPROMs

ROM BANK 1; OPERATING SYSTEM AND FSCC DIAGNOSTICS

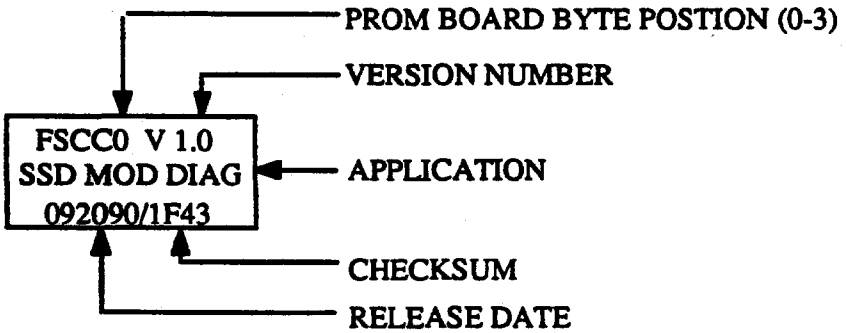


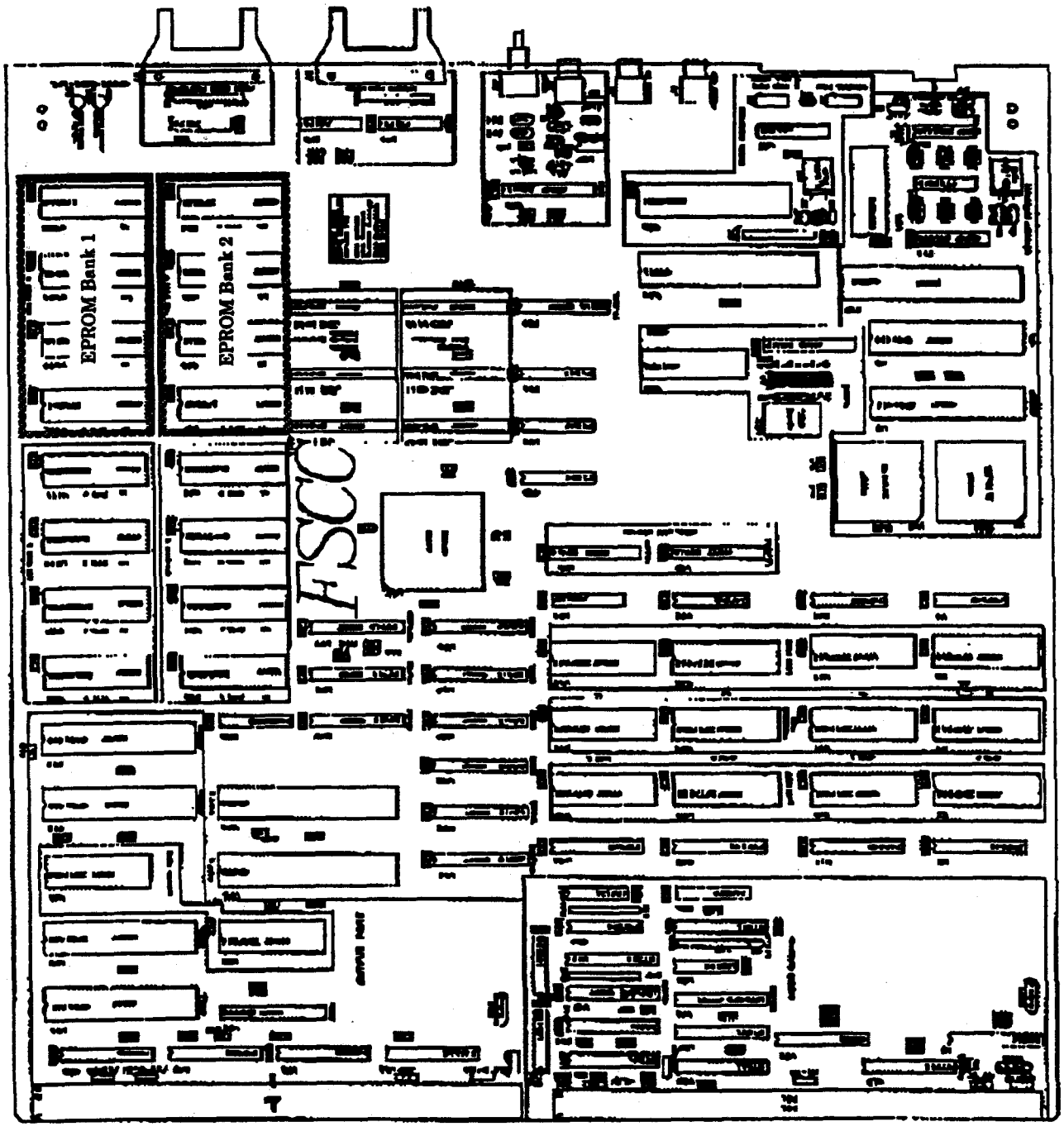
ROM BANK 2; SYSTEM OR INDIVIDUAL MODULE DIAGNOSTICS

OPTION 1; System Diagnostic Tests



OPTION 2; Individual Module Tests





FASTBUS SMART CRATE CONTROLLER

VERSION PC3

EPROM Assembly Drawing

Fermi National Accelerator Laboratory

MASTER TIMING CONTROLLER

HARDWARE DESCRIPTION

M.Fachin, C. Rotolo, C. Needles, P. Spentzouris

October 19, 1990

Table of Contents

1. INTRODUCTION

The Master Timing Controller (MTC) module is part of the Silicon Strip Detector System, intended for use in experiments E771 and E789. In addition to generating the system clock coherent to the beam, the MTC provides the mechanism to maintain system synchronization, and acts as an interface between the readout system and the 1st level trigger system, generating hit addresses in response to trigger requests. The MTC provides a pipeline for triggers and is responsible for system reset (initialization). Figure 1 shows the MTC connections to the Sequencer modules and to the trigger system.

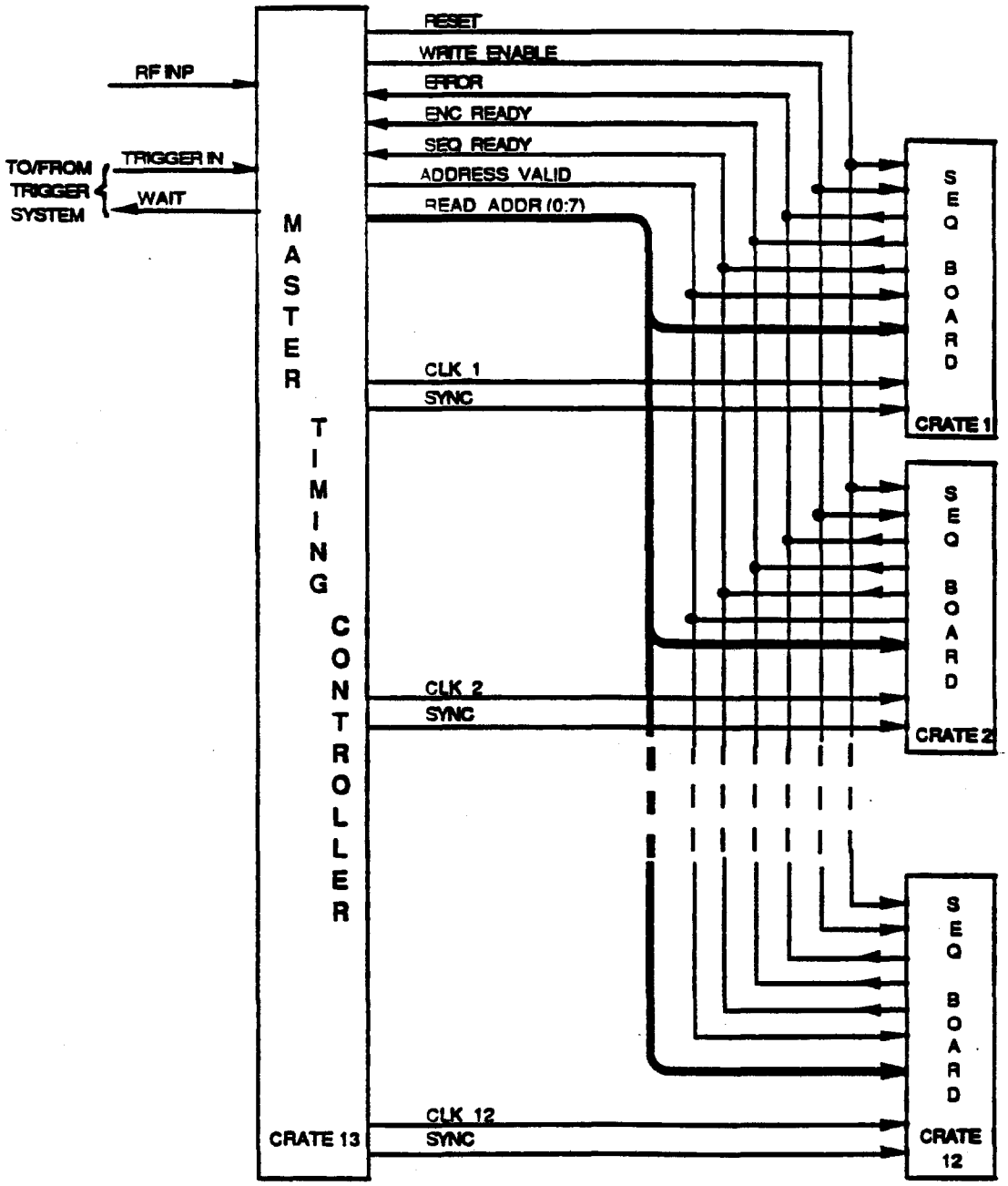


Fig. 1 - MTC connections to the Sequencers and 1st level trigger system.

2. GENERAL DESCRIPTION

The Master Timing Controller generates the system clock, controls system synchronization, and generates the hit addresses upon the receipt of Level 1 triggers from the experiments Trigger system. A block diagram of the MTC is shown in Figure (2). The MTC receives the 53 MHz Tevatron RF and establishes a near 50% duty cycle clock whose phase is adjustable relative to the incoming RF. This CLOCK along with a SYNC pulse (occurring every 256 clock cycles) are distributed to each Sequencer and eventually to all Delay/Encoders (D/Es). D/Es use this clock and sync to determine write addresses for incoming data. Being synchronized, the MTC knows the current D/E write address and generates a read or "hit" address when a trigger is received. The hit address generated is offset from the write address based upon a calibration of the Trigger decision time.

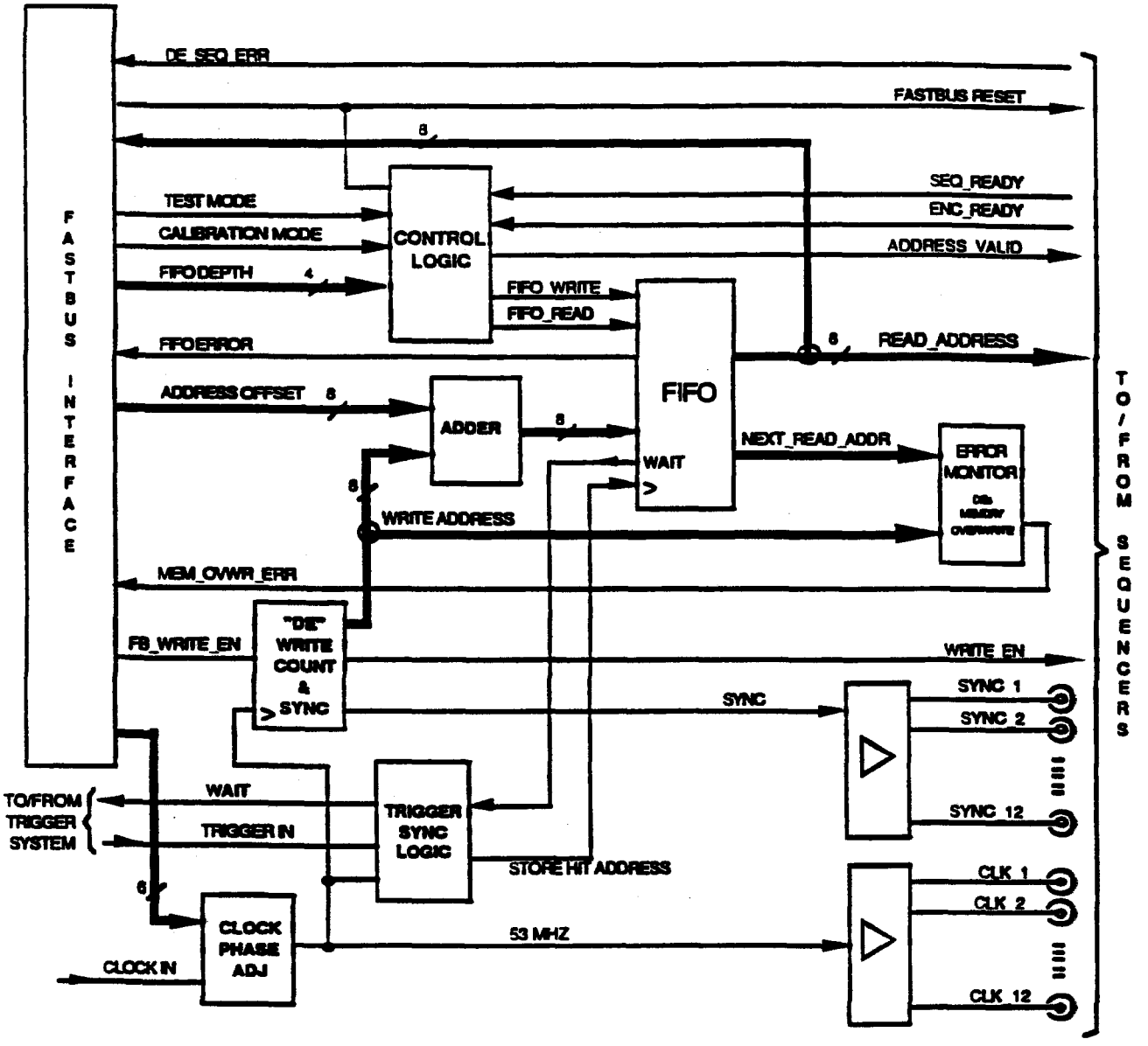


Fig. 2 - MTC internal block diagram

In the case of D/Es being busy, hit addresses are placed into a high speed FIFO queueing up to seven trigger requests which can occur as close together as successive RF buckets. The read or "hit" address output from the FIFO is broadcast to all D/E modules at a rate determined by the ENC_READY signal summed from all D/E modules.

The ability to pipeline triggers makes the system truly deadtimeless at trigger rates up to the readout bandwidth. With knowledge of the read address and the D/E's current write address, the MTC detects fatal D/E memory overwrite errors (256 clock cycles, or 4.8 μs, is how much time it takes to overwrite the memory). However, to prevent such a condition, the system is throttled by sending a WAIT signal to the Trigger system — the MTC trigger pipeline depth can be set by Fastbus and will generate a trigger WAIT at this depth.

The MTC provides trigger synchronization monitoring and incorporates features for system calibration and debugging.

3. CLOCK GENERATION AND SYSTEM SYNCHRONIZATION

Figure 3 shows the Clock and the Sync generation block diagram.

The MTC expects a continuous NIM 53 MHz RF from the accelerator, and generates a fixed duty cycle near 50% that is fanned out individually to each crate. In order to maintain constant phase relationship to the RF, it is suggested that the sinusoidal RF signal be processed by EG&G Model 140/N Zero Crossing Discriminator, producing NIM level pulses, prior to being fed to the MTC, which in turn reshapes the 53 MHz pulses with a 8 ns delay line to achieve the desired duty cycle. To account for slow drifts of the RF with respect to the actual beam, as well as equipment changes, the MTC provides a 6-bit programmable delay line with 0.5 ns resolution for clock phase adjustment. This delay line is programmed through FASTBUS.

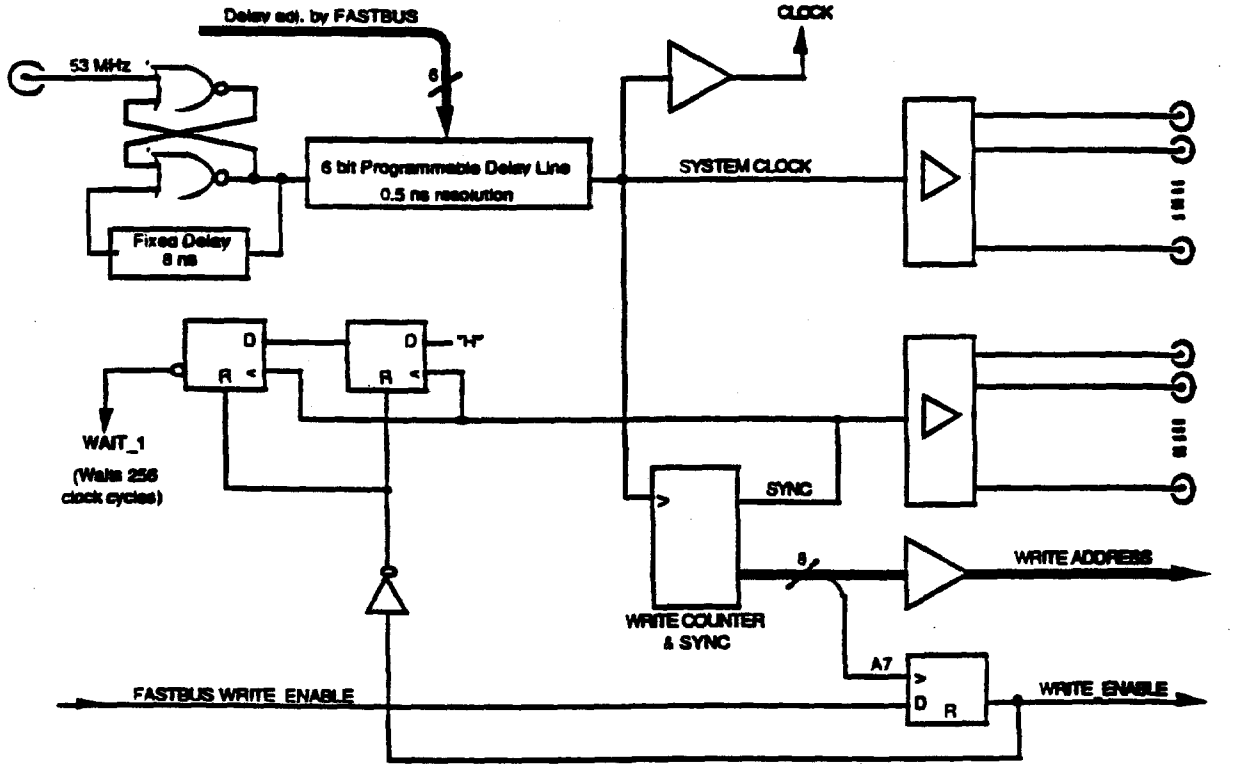
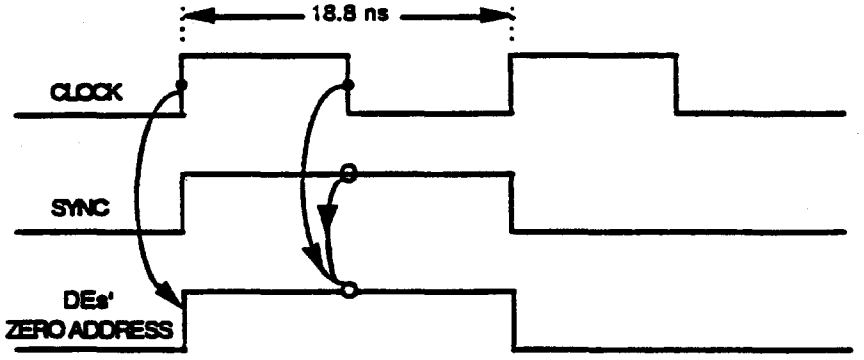


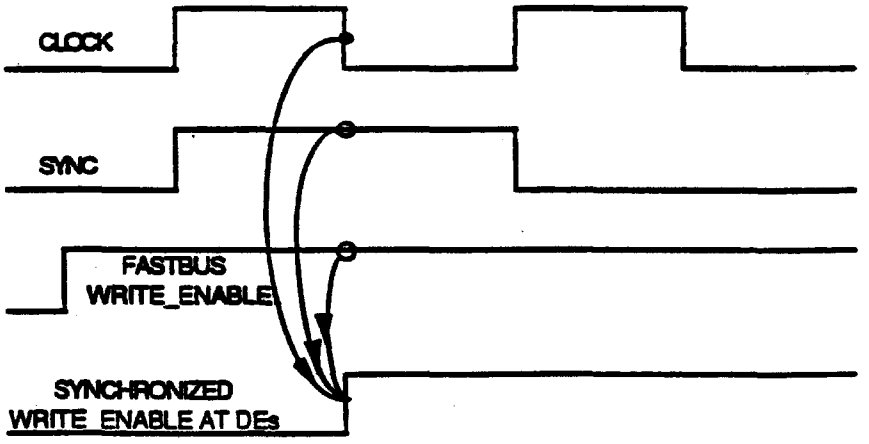
Fig. 3 - Clock Generation and System Synchronization

The MTC is thus capable of delivering a fixed phase 53 MHz clock relative to the beam. This clock is the absolute timing signal for the system and can be used as a reference throughout the system. If individual modules of the Data Acquisition System need a different phase relationship, they have to adjust the phase locally.

The MTC generates a SYNC signal with fixed phase relative to the clock on every 256 clock cycles. The MTC keeps a copy of the D/Es write counters for use as a reference to evaluate the read address (hit address). The MTC thus knows when the write counters in every D/E are expected to be at address zero. At this precise time the MTC broadcasts a 18.8 ns SYNC pulse to all Sequencers in the system. The Sequencer will then bus this signal to all the D/Es in the crate, with each D/E checking its synchronization to the SYNC pulse. If a loss of sync is detected, the faulty D/E will inform the Sequencer in the crate by asserting the SYNC_ERROR line (wired-or of all D/Es in a crate). Each faulty D/E will identify itself by latching the error and displaying it on the front panel. Although the SYNC signal has a close phase match to the system clock, it is not used as a timing signal. The SYNC signal is sampled in the D/Es on the trailing edge of the system clock, as shown in Figure 4a, and is allowed some phase skew without affecting D/Es synchronization. The Sequencers, which have control over the clock phase in each crate, have to adjust the SYNC signal phase if they change the clock phase.



(a) Synchronism checking in the D/Es



(b) WRITE_EN Synchronization

Fig. 4 - D/E and WRITE_ENABLE timing diagrams

The SYNC signal has to be calibrated internally in the MTC to phase match the clock signal. A tapped delay line is provided for this purpose, with dip switch SW3 being used to select the tap which gives the right phase. This calibration is done only once and remains fixed thereafter.

The WRITE_EN signal programmed through FASTBUS is intended to start the D/Es to acquire data, as well as allowing the MTC to accept trigger requests. The WRITE_EN is sent to the Sequencers/D/E 128 clock cycles before the SYNC pulse, so that all D/Es can see it before the next SYNC pulse.

On system start up, the WAIT signal to the trigger system is deasserted only after WRITE_EN is asserted, and 256 clock cycles after the D/Es have started taking data.

A flow chart of the system initialization is shown in Figure 5.

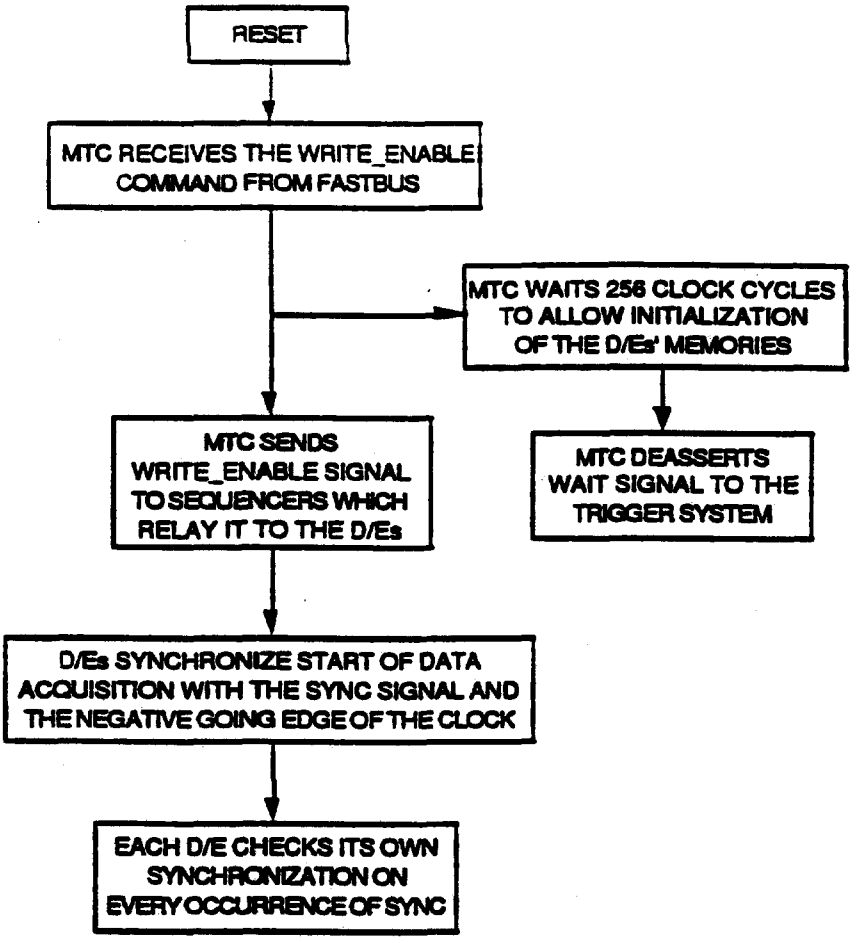


Fig. 5 - System initialization

4. READ ADDRESS

Each trigger pulse received by the MTC generates a READ_ADDRESS — address where the hit data corresponding to the trigger is stored in the D/Es memories —, obtained by subtracting an offset to the reference D/E write counter of the MTC. This offset is software programmable and its value is determined by a calibration procedure. The offset accounts for the trigger decision time, which is expected to be close to 1 μs (for example, for a trigger decision

time of 1 μs, the offset would be 1000 ns/18.8 ns ≈ 53 clock cycles). The READ_ADDRESSes are sent asynchronously to the Sequencers (and D/Es) along with the ADDRESS_VALID signal. If the D/Es are busy, the MTC stores the trigger addresses in a FIFO for later delivery. An error is issued if the NEXT_READ_ADDR to be output from the FIFO is too close (dip switch programmable in SW2) to the current write address in the D/Es, such that a MEM_OVWR error could occur. An opened switch on SW2 means a logic one, and the quantity programmed in SW2 should be equivalent to the time (number of clock cycles) needed for the READ_ADDRESS to reach the D/Es and be processed before that memory location is overwritten with new data. The memory overwrite error monitor is inhibited when the FIFO is empty, a situation in which the hardware would flag a non-existing error.

5. TRIGGERS

Triggers can be external or internal. External triggers come from the 1st level trigger system and signal an event (external triggers are inhibited in Test mode and under error conditions in other modes). Internal triggers are software generated triggers and are intended for test purposes only. There are no hardware restrictions as to when an internal trigger is allowed. Triggers produce READ_ADDRESSes that are loaded into the trigger FIFO.

5.1. Trigger Phase

The incoming trigger pulse must hold a certain phase relationship to the system clock for the MTC to work properly. Specifically, a trigger must not arrive close to the rising edge of the clock, which would cause the trigger synchronization logic to malfunction, by not honoring the setup time of the logic. If the trigger phase is not set right, the MTC will flag an error named TRIG_PHASE_ERR, which will halt the system by discarding new trigger requests and by not sending out READ_ADDRESSes.The MTC produces the signals TRG WIN (trigger window) and TRG MON (trigger monitor) to help adjust the trigger timing. By observing these signals with a scope, one should externally delay the trigger pulse input such that the TRIGGER MONITOR signal lies inside the range presented by the TRIGGER WINDOW signal. It is recommended that the TRIGGER MONITOR signal be positioned in the center of this time window, to allow for eventual timing drifts.

An alternative way of setting the trigger timing is to monitor the TRIG_PHASE_ERR LED while making the delay adjustments. By noting the range for which no errors are flagged, one can set the trigger delay to be centered to the window.

The width of the window described above can be trimmed to allow a narrower or looser phase error check for the incoming triggers. A rotary switch (SW5) is provided for setting the window width.

In test mode, triggers are generated by software and have a random phase. The trigger phase errors are disregarded in this case, and do not halt the system.

5.2. Trigger Interface

The MTC accepts pulses from the 1st level trigger system and broadcasts the addresses of the data corresponding to those triggers (READ_ADDRESSes) to the Sequencers/D/E. The MTC is capable of pipelining trigger requests if the D/Es are busy encoding the previous trigger. The pipeline depth, programmable through FASTBUS, depends on the trigger delay, and on the expected average number of hits in the detector, which in turn causes different encoding times in the D/Es. A wise selection of the number of stages in the pipeline prevents the D/Es from wrapping around their memories and consequently overwriting data; the MTC tests for this type of error as said in the previous section. When the number of events waiting to be serviced exceeds the maximum programmed number of stages in the pipeline, the MTC sends a WAIT signal to the trigger system. The MTC, however, doesn't block out new trigger requests, even though it has sent the WAIT to the trigger system. This feature is important in the case of trigger

requests that are in the process of being delivered when the WAIT signal is asserted. The trigger request FIFO can store up to 7 trigger requests.

For test purposes, triggers can be generated by software (FB_TRIGGER) and are OR'ed to external triggers.

5.3. Trigger WAIT

The way the MTC throttles the trigger system is by sending a WAIT signal. This signal is asserted under four conditions: whenever the system is in stand-by, is running in Test mode, the FIFO depth exceeds the programmed FIFO depth, or the Sequencers are not ready (the Sequencers' FIFOs used to store the encoded data from the D/Es become more than half full with encoded data from the D/Es).

5.4. Trigger FIFO

When the D/Es are busy encoding the previous trigger, the MTC stores the READ_ADDRESSes corresponding to the incoming triggers in a FIFO. Commercial FIFOs available today have limitations in speed and functionality. To meet the requirements of the SSD trigger pipelining, a discrete ECL FIFO is built in the MTC, with the following characteristics:

- 53 MHz input frequency
- depth (number of stages) programmable through FASTBUS
- generation of a WAIT signal if the current number of stages in the FIFO is greater than the FASTBUS programmed depth
- generation of status such as *empty, full, and error*
- generation of the NEXT_READ_ADDRESS

The number of stages in the FIFO was limited to 7. Figure 6 shows the FIFO connections to other MTC blocks.

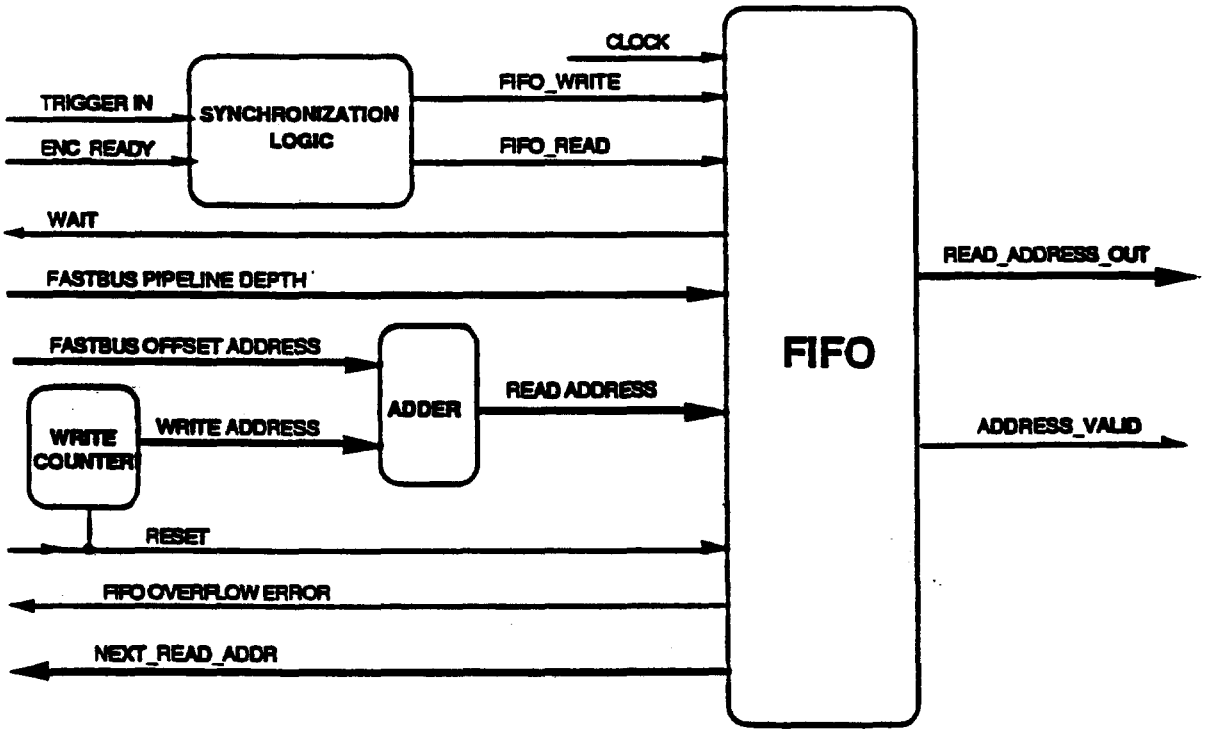


Fig. 6 - FIFO signals

The pipelining functional characteristics are:

- *fifo empty*: do nothing; wait for trigger.
- *fifo not empty*: if the D/Es are READY, retrieve a read address stored in the pipe; point to the next read address in the pipe.
- *fifo depth > programmed number of stages*: send WAIT to the trigger system. This is a very important feature that makes the FIFO more flexible, allowing the trigger pipelining to be tuned to the detector hit rate.
- *fifo depth > 7*: error.

A more detailed block diagram for the FIFO is depicted in Figure 7.

The FIFO is implemented as 8 registers, 2 pointers (write and read pointers) and a status generation logic.

The write pointer is always pointing to the next free location in the FIFO. Upon receiving a trigger pulse, the FIFO writes the hit address to the current free location and increments the pointer to the next one. The write operation is synchronized internally to the falling edge of the clock; the READ_ADDRESSes are generated after the rising edge of the clock.

The read pointer is always pointing to the register where the data in the pipe is to be read. The contents of this register is available and is called NEXT_READ_ADDR, used in the MEM_OVWR error monitor. In response to a FIFO_READ, this data is latched and becomes the READ_ADDRESS, which is sent out to the Sequencers/D/Es. The read pointer is then incremented, pointing to the NEXT_READ_ADDR. In order to allow minimum time for the error and status circuits, the FIFO_READ is synchronized internally to the rising edge of the clock. This provides one half clock cycle for the error checking circuit, since the writes occur on the falling edge of the clock. The difference between the write pointer and the read pointer is the FIFO depth.

6. MODES OF OPERATION

The MTC has 3 modes of operation: Run (Acquisition) mode, Test mode, and Calibration mode.

6.1. Run Mode

The Run mode is the normal mode for data acquisition. The MTC basically broadcasts READ_ADDRESSes in response to trigger requests, and is capable of pipelining triggers (read addresses) when the system is busy. The MTC checks for system integrity, halting the system on the occurrence of an error. A FB_WRITE_EN causes the MTC to enter Run mode.

6.2. Test (Debugging) Mode

The Test mode feature serves many purposes, the most important one being the system test. In TEST_MODE, the D/Es acquire data in the usual fashion, the data now being a known pattern generated by the Post-Amp/Comparator board. The TEST_MODE capability allows one to read the contents of the D/Es' memories and compare them to the known pattern being written into the memories.

In this mode, the write counter in the MTC is shut down (SYNC pulses are generated by an alternate counter used solely for this purpose), and the read address is controlled by the trigger offset (TOFFSET) setting alone. The actual read address that is broadcast is achieved by subtracting an offset from the current write counter contents (which is zero in this case).

"Test" READ_ADDRESS = 256 - TOFFSET

In this way, one has control over which addresses to read from the D/Es. A software program can scan over the entire D/E memory and check for correct data.

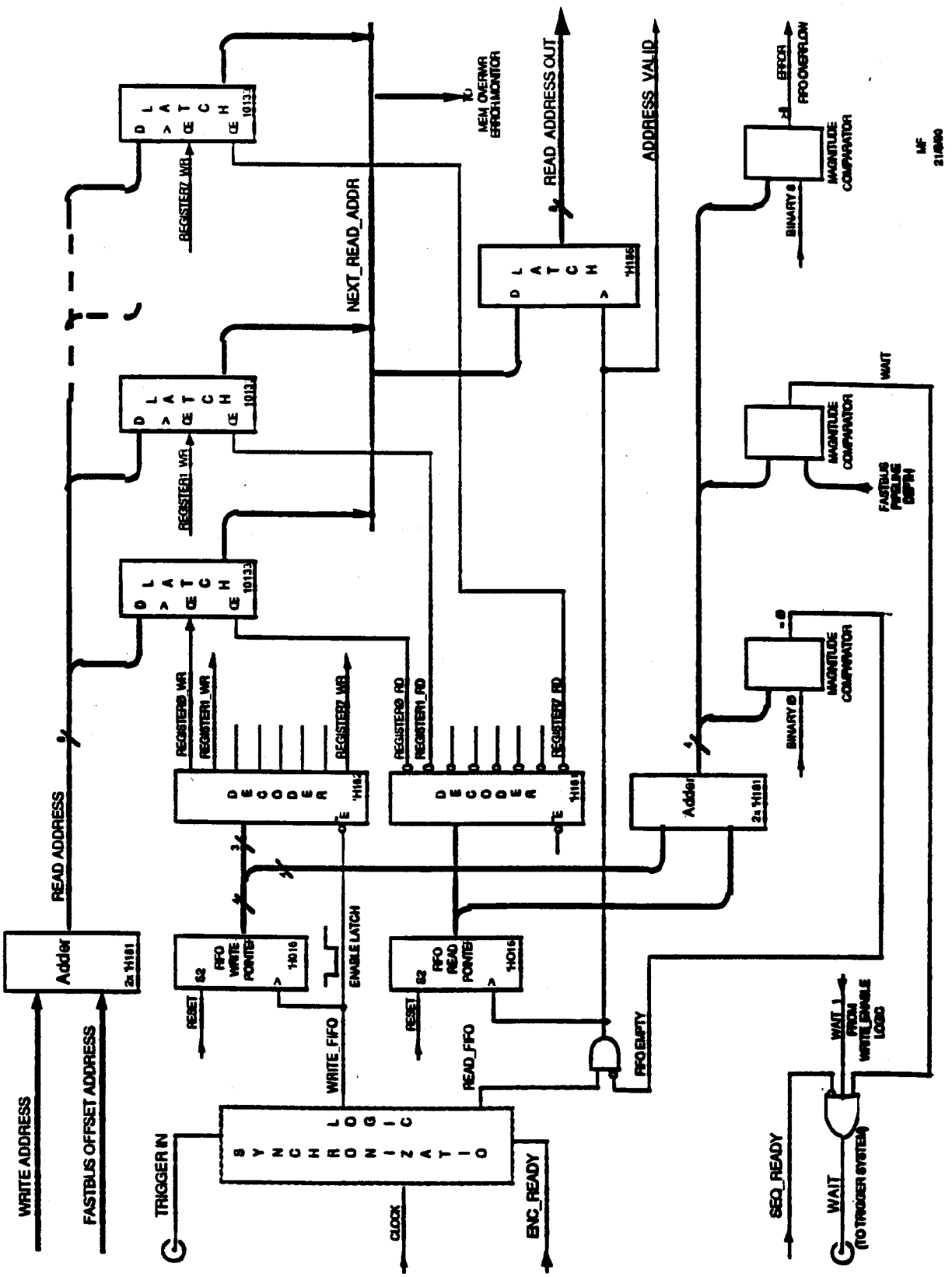


Fig. 7 - Fifo internal logic

External trigger requests are disregarded in this mode, and triggers are generated by software (FB_TRIGGER). The software generated triggers are not phased to the clock, and their role is to load test READ_ADDRESS into the trigger FIFO. A WAIT condition can be read back through Fastbus, caused by either a FIFO_WAIT or by the Sequencers not being ready, and has to be observed by the software routine that generates triggers.

The procedures to execute the test are:

. Assert the TEST_MODE signal.

. Assert FB_WRITE_EN. This will initiate the D/Es accepting hit data.

. Load an offset that will give the desired READ_ADDRES:

READ_ADDRESS = 256 - TOFFSET

. Generate the FB_TRIGGER pulse by software.

. Change offset and wait for ENC_READY to issue another FB_TRIGGER

The Test mode can be also used to test the MTC alone. A great amount of the MTC's hardware can be checked under Test mode. The generation of a FB_ENC_READY closes the loop on the board, by allowing triggers stored in the FIFO to be read back through Fastbus. This checks the circuitry that evaluates the read address, the FIFO itself, and the Fastbus interface. The routine can be extended to further check the FIFO depth (generates Fastbus readable WAIT signals if the number of stages in the FIFO exceeds the programmed depth), the memory overwrite error monitor, and the hardware used in the calibration feature of the MTC.

6.3. Calibration Mode

This mode is used for evaluating the correct TOFFSET (number of clock cycles) necessary to accomplish for trigger decision time and other intrinsic delays. When set to Calibration mode, the MTC waits for the first external trigger pulse to arrive and generates N (switch setable) consecutive read addresses, using an estimated offset. The number of triggers produced is equal do the switch setting plus 1.
Fig. 8 shows a diagram of this switch, which is set to the number 4 in the figure (5 trigger pulses produced)

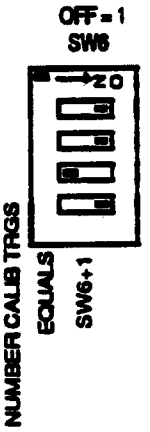


Fig. 8 - Diagram for the number of calibration triggers switch

These READ_ADDRESSes are stored in the MTC's trigger FIFO, and are delivered on demand to the D/Es. Additional trigger pulses cause the MTC to generate a new burst of N trigger addresses synchronized to the arrival of the trigger pulse input. In other words, each trigger pulse will generate one burst of N consecutive trigger addresses. However, if a new trigger arrives before the N calibration triggers corresponding to the previous trigger have been stored in FIFO, the FIFO_OVFL_ERR is generated.

The procedure to perform the calibration cycle is:

. RESET the module to clear the error latches and reset the FIFO.

. Set trigger pipeline depth (FIFO_DEPTH) to 1, so that the FIFO_WAIT will inhibit external triggers if the first burst of N read addresses is not serviced yet.

. Assert CALIB_MODE

. Set WRITE_EN

If any error occurs. external triggers are blocked out and ADDRESS_VALIDs are not produced.

Performing this calibration procedure with different clock phases (adjusted in the Sequencers) and different offsets, the system is able to determine the correct offset by looking into the data collected in the D/Es. This offset then becomes a constant to the system and will be used by the MTC to determine the READ_ADDRESSes that are broadcasted to all Sequencers/ D/Es in the system.

The Calibration mode internal hardware of the MTC can be checked by operating the MTC with both the Calibration mode and Test mode asserted. This allows one to have control of the READ_ADDRESS to be stored in the trigger FIFO (see Test mode) when a a calibration cycle is initiated by a software generated trigger. The expected contents of the FIFO are N (switch setable) consecutive numbers, the first one being equal to the loaded TOFFSET subtracted from 256.

7. ERROR MONITORS

The MTC monitors errors in the system. Any error is fatal, and the mechanism by which the MTC informs the control system of an error in the DAS or itself is to halt the system, i.e., external triggers are disregarded and ADDRESS_VALIDs are not sent out, except when the MTC is in Test mode. In addition, the MTC provides a NIM output indicating the presence of an error, and front panel LEDs for visualization of the errors. All errors are latched, and can be cleared by software or by pressing the front panel switch CLEAR ERRORS (provided the error conditions has been corrected).

The errors can be read through FASTBUS and include:

DE_SEQ_ERROR : means that a D/E module has lost synchronization or that a Sequencer FIFO has overflowed. To identify the exact cause of the error, one needs to read the Sequencers' error registers.

MEM_OVWR_ERROR : indicates that a trigger address cannot be broadcast due to data being overwritten in the D/E memories. The MTC compares the D/E' write address against the NEXT_READ_ADDR; if they are too close to one another, as programmed in dip switch SW2 (memory overwrite margin), the MEM_OVWR_ERR is flagged.

FIFO_OVFL_ERR : indicates that the trigger pipelining FIFO in the MTC has overflowed. This error may occur if the trigger system disregards the WAIT signal the MTC sends to it informing the pipe is almost full. Normally, the WAIT signal is sent out before the FIFO becomes full, whenever the programmed depth is exceeded.

TRIG_PHASE_ERR : signals that the trigger input from the first level trigger system has drifted by an amount that causes it to fall outside a pre-established time window.

CLOCK_MISSING : detects that at least one clock cycle was not received from the accelerator.

ERROR_SUM : it's the OR logic of all errors.

8. FRONT PANEL I/O AND DISPLAYS

Figure 8 shows a section of the MTC's front panel.

8.1. Front panel LEDs

MOD SEL : it is a yellow LED that flashes when module address is selected by Fastbus.

FIFO DEPTH : Is composed of 8 green LEDs numbered 0-7. They monitor the difference between the read and write pointers of the FIFO. If at 0, the read and write pointers are

pointing to the same register in the FIFO, which means the FIFO is *empty*. The maximum depth is 7, and an additional write to the FIFO will cause it to overflow, flagging an error.

- Errors -

MEM OVWR : indicates a MEM_OVWR_ERR in the D/Es.

FIFO OVF : indicates that the FIFO has overflowed.

TRG PHZ : monitors the external trigger phase to be within a pre-established time window.

DE/SEQ : signals that at least one D/Es is out of sync or that some Sequencer FIFO has overflowed.

CLK MIS : indicates the 53 MHz system clock is missing or was not present for at least one clock cycle.

- Status -

WRITE EN : status LED indicating the MTC is ready to accept triggers. The WRITE_EN is required also when the MTC is being operated in Test or Calibration modes.

CALIB : informs the MTC is in Calibration mode..

TEST : Test mode indication.

TRG IN : alongside TRG IN connector, shows when the MTC is receiving external triggers.

TRG WAIT : alongside the WAIT output, shows WAIT signals being sent to the trigger system. A persistent WAIT condition will leave the LED constantly lighted.

- At the front panel bottom -

See Fig 10.

+5 : monitors the 5 volts power in the MTC board.

-5.2 : monitors the -5.2 volts power for the ECL logic.

-2 : monitors the -2 volts power supply for the ECL terminations.

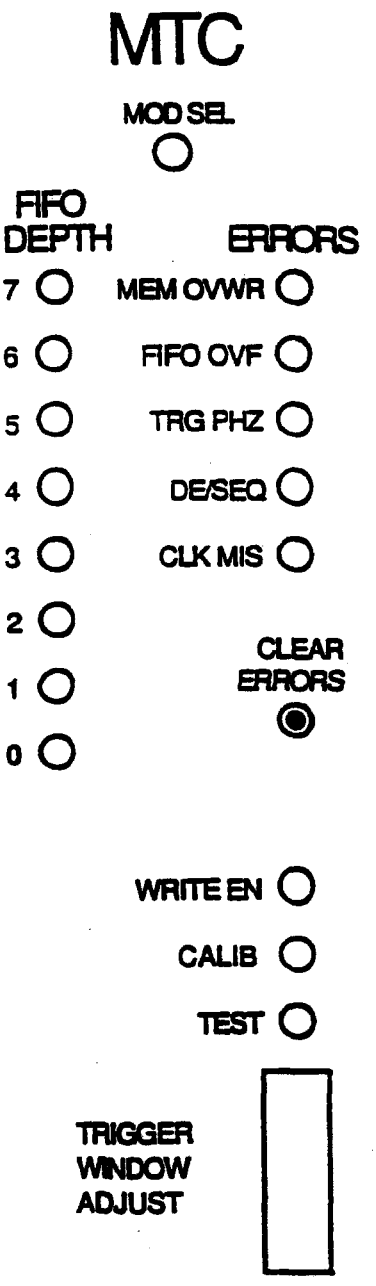


Fig 8 - MTC's front panel partial view

8.2. Coaxial connectors:

Fig 9 depicts the MTC's front panel coaxial connectors.

CLOCK IN : a NIM 53 MHz clock input which is derived from the accelerator RF via the CATV system. The only concern here is to have a signal that has a fixed phase relationship to the RF. This signal may experience slow timing drifts over long periods of time, which can be compensated by reprogramming (Fast-bus operation) the delay line internal to the MTC.

CLOCK OUT : a NIM 50% duty cycle clock output reference that has a constant phase to the beam. This signal is fanned out individually to each of the 12 crates. The 13th output is for monitoring.

SYNC OUT : NIM output pulse, synchronous to CLOCK, to test write counters sync at each zero count. This signal is delivered individually to each Sequencer. The 13th output is for monitoring.

ERROR : NIM output intended for immediate signalization of an error condition.

TRG IN : NIM trigger pulse input from the 1st level trigger system.

TRG MON : output used in conjunction with the TRG WIN signal to adjust the phase of the trigger signal.TRG WIN : presents the time window for phasing the trigger.

TRG WAIT : NIM output signal informing the 1st level trigger that the system is busy and cannot accept new trigger requests

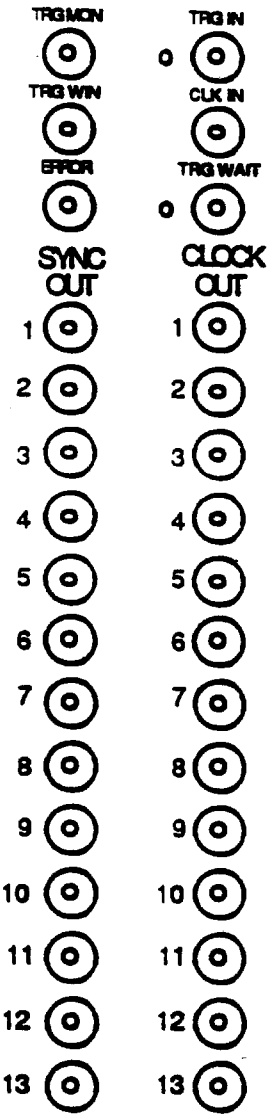


Fig 9 - Front panel Coaxial Connectors

8.3. Ribbon cable connector:

All signal in the ribbon cable connector are bussed to all crates. It is shown in Fig. 10.

RESET : initiated from Fastbus, it accomplishes system resetting and initialization. It is a differential ECL output signal, asynchronous to CLOCK.

WRITE_EN : output signal, differential ECL, asynchronous to CLOCK, for enabling the D/E modules to start data acquisition, i. e., accept hit data and increment write counters.

ENC_READY : single-ended wire OR'd ECL input signal from the Sequencers informing the D/Es status. This signal is used in determining when to deliver another

READ_ADDRESS to the Sequencers/ D/Es. It is the OR logic of the encoders not being ready.

SEQ_READY : input from the Sequencers. single-ended wire OR'd ECL, it signals that the Sequencers have room in its FIFOs for encoded events. If the Sequencers are not READY, the MTC immediately sends a WAIT to the trigger system. SEQ_READY performs the OR logic of the Sequencers not being ready.

DE_SEQ_ERR : single-ended wire OR'd ECL input from the Sequencers informing that the event FIFOs in the Sequencers have been overfilled by the encoders or that a D/E has lost synchronization.

READ_ADDRESS : 8 bit differential ECL output, asynchronous to CLOCK.

ADDRESS_VALID : it is a 80 ns wide differential ECL output pulse, asynchronous to CLOCK, occurring 80 ns after the READ_ADDRESS is asserted. It signals the Sequencers that a new READ_ADDRESS is available.

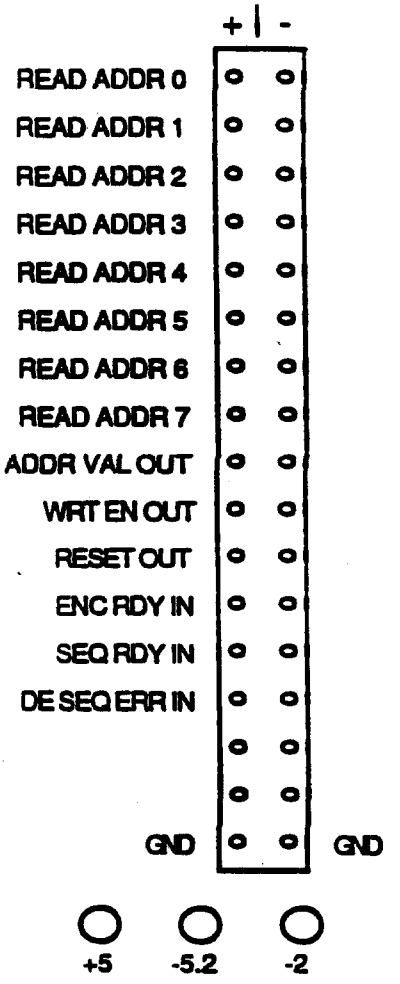


Fig 10 - Ribbon Cable connector

9. FASTBUS INTERFACE

The MTC communicates to the external world through FASTBUS. Functions executed through FASTBUS include system resetting, system calibration, system debugging, clock phase adjustment, pipeline depth setting, read address offset programming and error reporting.

The MTC is a slave device that responds to geographical address in the CSR space. The module ID is 01A2. The signals and addresses are listed below.

9.1. CSR0

CSR0 is used to control the MTC. Its signals are listed below:

bit	write	read
CSR0(00)	Set Error Flag	Error Flag
CSR0(16)	Clear Error Flag	ID
CSR0(02)	WRITE_EN Set	WRITE_EN
CSR0(18)	WRITE_EN Clear	ID
CSR0(06)	Calibration Mode set	Calibration Mode status
CSR0(07)	START_CALIB	ENC_READY

CSR0(22)	Calibration mode clear	ID
CSR0(08)	Test Mode set	Test Mode status
CSR0(09)	not used	CLOCK_MISSING
CSR0(24)	Test Mode clear	ID
CSR0(10)	not used	DE_SEQ_ERR
CSR0(11)	not used	MEM_OVWR_ERR
CSR0(12)	not used	FIFO_OVFL_ERR
CSR0(13)	not used	TRIG_PHASE_ERR
CSR0(30)	Reset	ID
CSR0(16:31)		MODULE ID (01A2)

RESET is not latched internally, and the RESET pulse is the result of writing a one to bit 30 of CSR0. A hardware reset is performed on system power up, and a software reset is issued in system initialization or after a fatal error had halted the system. RESET causes the MTC to go to the following state:

. WRITE_EN is reset
. Calibration mode is reset.
. Test mode is reset.
. All errors are cleared.
. TOFFSET, CKPHASE, and PRDPTH (programmable FIFO depth) are left unchanged.

The FB_WRITE_EN command signal to start data acquisition is set by CSR0(02) and reset by CSR0(18).

The Calibration mode is set by CSR0(06) and reset by CSR0(22).

The Test mode is set by CSR0(08) and reset by CSR0(24).

9.2. CSR10 : READ_ADDRESS

CSR10 is used in Test mode to read back the READ_ADDRESSes and to generate FB_TRIGGERs and FB_ENC_READYs.

bit	write	read
CSR10(0:7)	don't care	FIFO_RA(0:7)
CSR10(08)	FB_TRIGGER	not used
CSR10(09)	FB_ENC_READY	not used

FB_TRIGGERs and FB_ENC_READYs are not latched internally. Pulses on these lines are produced by writing a one to bit 8 and 9 of CSR10, respectively.

9.3. CSR11 : PRDPTH

CSR11 programs the trigger pipeline depth (PRDPTH). Legitimate values are 1 to 7. CSR11 is also used to monitor status used in Test mode.

bit	write	read
CSR11(0:2)	PRDPTH(0:2)	same
CSR11(4)	don't care	TRIG_WAIT
CSR11(5)	don't care	FIFO_NOT_EMPTY

The status of WAIT output on the front panel can be read through CSR11(4) which reads 1 whenever WAIT is asserted. This is intended for module diagnostic purposes.

The NOT EMPTY/EMPTY status of the FIFO can be read in CSR11(5) which reads 1 when the FIFO is NOT empty (the FIFO has READ_ADDRESSes stored in it) and 0 when the FIFO is EMPTY. This bit is intended for module diagnostic purposes.

9.4. CSR12 : TOFFSET

CSR12 holds the 8-bit trigger address offset (TOFFSET).

bit	write	read
CSR12	TOFFSET(0:7)	same

9.5. CSR13 : CKPHASE

CSR13 is used to adjust the the internal clock delay. Six bits are used to program the ELMEC PDH 6500 delay line with a .5 ns resolution.

bit	write	read
CSR13	CKPHASE(0:5)	same

9.6. Fastbus Error Responses

SS=7 Bad NTA R/W

SS=6 R/W to invalid address

SS=2 End of Block (Although not normally used, the MTC is capable of Block Transfers)

10. MODULE CALIBRATION

As said in section 3, the MTC requires the phase of the SYNC signal to phase match the clock phase. This is accomplished in SW3 (only one switch should be closed at a time) and needs to be performed only once.

11. MODULE INTERNAL SETTINGS

In addition to SW3 (SYNC phase) adjustment which is fixed for a particular module, two other dip switch adjustments are required. They are dependant on the experiment, which means that they have to be tuned to a particular system. These dip switches are SW6 for adjusting the number of triggers for Calibration mode, and SW2, used for the memory over-write margin adjustment.

SW6 is shown in Fig. 8 and explained in section 6.3 above.

SW2 setting depends on the number of clock cycles required for a READ_ADDRESS to be recognized by the D/Es, after a ADDRESS_VALID signal was issued. This number compensates for all delays due to cables, Sequencer processing and D/E acknowledging. In other words, this number should be set to some safe number (margin) that would guarantee that the D/Es' memories will retrieve the stored data before being overwritten by new data. The MTC monitors this situation by knowing the current D/Es' write address and the NEXT_READ_ADDRESS that is to be sent out.

Another switch, SW5, is provided for adjusting a time window for incoming triggers. SW5 is a rotary switch and is located in the front panel. Positions 1 to 5 are used. SW5 setting depends on how tight the external trigger phase is to be monitored.

12. POWER REQUIREMENTS

5 Volts @ 2 A

-5.2 Volts @ 8 A

-2 Volts @ 3.5 A

13. MTC PRELIMINARY TESTING

After assembling the MTC module, there are several tests that are required before automated software tests can be performed.

A Test Module (Appendix D) was built to provide the clock and trigger inputs to the MTC, so that the board can be tested. By providing a 53 MHz clock input to the MTC, the signals CLOCK OUT, SYNC, and TRG WIN should be present at the front panel coaxial connectors. It is important to check that all the 13 CLOCK and SYNC outputs deliver a nice NIM output. The CLOCK outputs should have approximately 50% duty cycle; the TRG_WIN width can be adjusted by a front panel rotary switch. This is the right moment to calibrate the SYNC phase delay, as explained in section 10.

14. AUTOMATED TESTS

The hardware requirements for testing the MTC are a standard Fastbus crate, with a Fastbus Smart Crate Controller, and the MTC Test Module, which provides the 53 MHz clock and the external trigger input to the MTC. Fig. 11 shows the connections between the MTC and the Test Module. The circuit diagram of the MTC Test Module is found in Appendix E.

The trigger phase, as observed in the trigger monitor output, has to be adjusted to reside within the time window presented by the TRIG_WINDOW signal (the leading edge of the window is dependent on the leading edge of the clock, and the width is set by the SW5 front panel rotary switch). See section 5.1 for a more detailed description. If the trigger phase is not correct, an error is produced and some tests will not run, since the MTC was designed to stop at an error condition.

The normal situation is when the external trigger produced by the 1st level trigger system is delayed externally in order to have the right phase. For the tests, however, the absolute phase is not important and the phase adjustment can be done by adjusting the clock phase, which changes the clock delay and, consequently, the window delay with respect to the trigger input. The trigger could also be delayed externally, using a delay line module or cable.

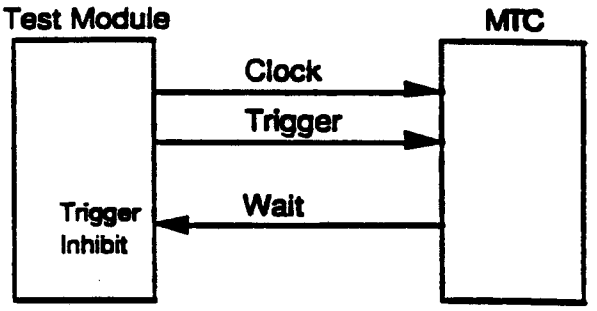


Fig. 11 - The MTC Test Module

The front panel TRIG PHZ LED goes on if the trigger phase is not correct. By providing the MTC with an external clock and an appropriate trigger input, the front panel error LEDs should go off by pressing the CLEAR ERRORS front panel push-button switch.

A program, called MTC Test Software, was developed to test the MTC. This program tests all MTC features, being able to perform system tests as well.

14.1. Test Software

The MTC Test Software is resident in the FSCC (Fastbus Smart Crate Controller), burned in EPROM. Below is presented a brief summary of what is expected from the hardware and software to accomplish the tests on the MTC.

Test Mode: External trigger requests are disregarded, triggers are generated by software (FB triggers).

Write counter is set to zero (256) and the READ_ADDRESS is controlled by the trigger offset (READ_ADDRESS = 256 - TOFFSET)

WAIT is always set in this mode.

FB_ENC_READY reads back the READ_ADDRESSes.

ADDRESS_VALID continues to be generated in the presence of errors.

Calibration Mode: Each external trigger pulse generates N (switch setable) consecutive READ_ADDRESSes. They are stored in the Trigger FIFO and send to the Sequencers/ D/Es on demand (upon receiving of a ENC_READY signal).

ADDRESS_VALID will not be generated in the presence of an error.

Calibration Mode may also be used in combination with Test Mode. This is useful primarily in checking if the MTC is generating the correct READ_ADDRESSes. The first READ_ADDRESS, in this way, can be determine by software.

Run Mode: Each external trigger generates a READ_ADDRESS; if the D/Es are not ready, the READ_ADDRESS is stored in the trigger FIFO.

The FIFO depth (number of stages in the FIFO) is programmable and WAIT is asserted if the number of FIFO stages is greater than the programmed depth. If the number of triggers exceed the maximum FIFO capacity (equal to 7), then the FIFO OVFL error will be generated.

If the read address (obtained by subtracting the offset — determined by the calibration procedure — to the reference D/E write counter of the MTC) is close by a switch setable amount to the current write address in the D/Es, a MEM_OVWR error will be issued.

Any of the above errors, or TRIGGER PHASE and D/E errors, will stop external triggers.

ADDRESS_VALID is inhibited until all errors are cleared.

MTC features that are tested

1) Fastbus interface

2) Fifo read address

3) Fifo overflow

4) Clock phase adjustment

5) Memory overwrite

6) Wait

7) Calibration

Description of tests

Pseudo code for each of the tests are:

Fastbus interface : Write and read to (from) CSR10 - CSR13;
set and reset flags (bits) and read status of CSR0.

Fifo read address :
Reset
Error flag reset
Test mode
Write enable
Trigger address offset to 256 - N
FB trigger
FB encoder ready
Read CSR10
Compare
Reset

The above test checks the arithmetic unit that evaluates the READ_ADDRESS, the trigger FIFO where this address is stored, and the control pulses FB_TRIGGER and FB_ENC_READY.

Fifo overflow :
Reset
Error flag reset
Test mode
Write enable
Set trigger pipeline depth to 7
While not overflow
  Set trigger address offset
  FB trigger
End While
Check number of triggers generated
Loop on trigger number
  Read back with Encoder Ready and
  check address
End Loop
Reset

The current FIFO depth can be found at any time by writing to the programmable depth register and observing the WAIT signal. The WAIT signal is asserted whenever the FIFO depth exceeds the programmed depth.

Clock phase adjustment : Change the clock phase over a range of values, checking for trigger phase error. The change in the clock phase causes the trigger window to move, throwing the trigger input outside the window, producing the error. For automatic tests make sure that trigger remains in phase (inside window).

Memory Overwrite :
Reset
Error flag Reset
Test mode
Write Enable
Loop on the 255 possible read addresses
  Error flag Reset
  Set trigger address offset
  Assert FB trigger
  Check Memory Overflow
  If not when address agrees with the Dip Switch, report it
  Assert FB trigger
  Read and compare address

In the Memory Overwrite test, the memory overwrite margin switch SW2 has to be hardcoded to the value 8, or one can change the default during the initialization time to agree with the switch setting.

```
Wait :  Reset
        Error flag reset
        Loop over fifo depth
          Set fifo depth
          Set write enable
          Loop until wait or error or timeout
          When wait or error, external triggers are inhibited
          Set test mode
          Assert Fb encoder ready until fifo is empty
          Compare with depth or if error report
        End Loop
        Reset

Calibrate :  Reset
             Error flag reset
             Set trigger pipeline to 1 so wait will inhibit all  but the first external trigger
             Set calibration
             Set Write enable
             Loop until Wait
                Set Test mode
                Loop
                   Assert Fb encoder ready
                   Fill array
                   Break if fifo empty
             End Loop
             Compare and check array, report errors      ; the FIFO contents should
                                                         ; read consecutive numbers
                                                         ; representing the read addresses
             Reset
```

Notes:

Test mode is required for reading back READ_ADDRESSes in order to prevent the blocking of ADDRESS_VALIDs due to MEM_OVWR error. For every FB write to a register, the corresponding read is performed in order to check the FB interface.

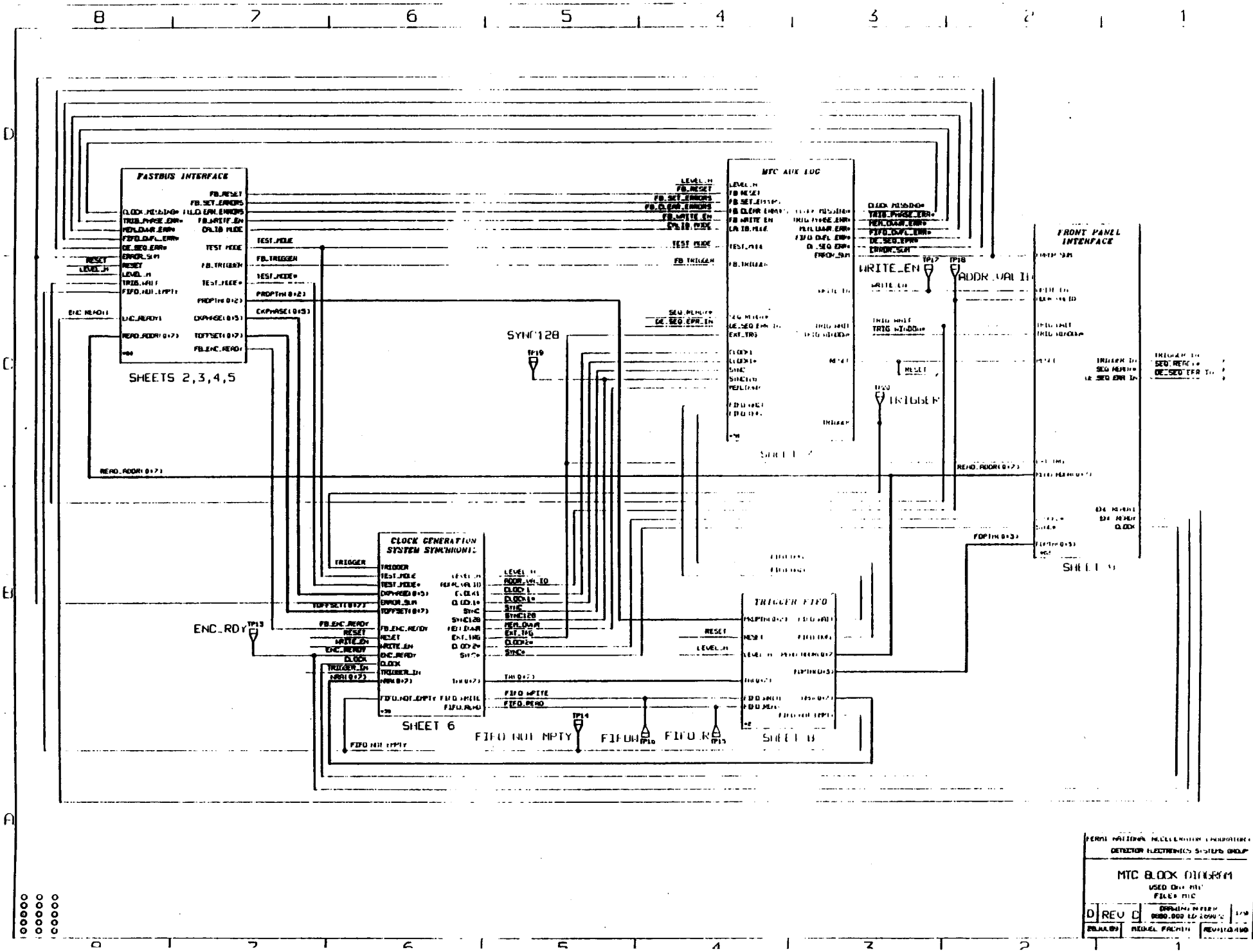
During the initialization, the default values that correspond to dip switches can be changed (main menu), or the hardcoded values can be tested to set the defaults.

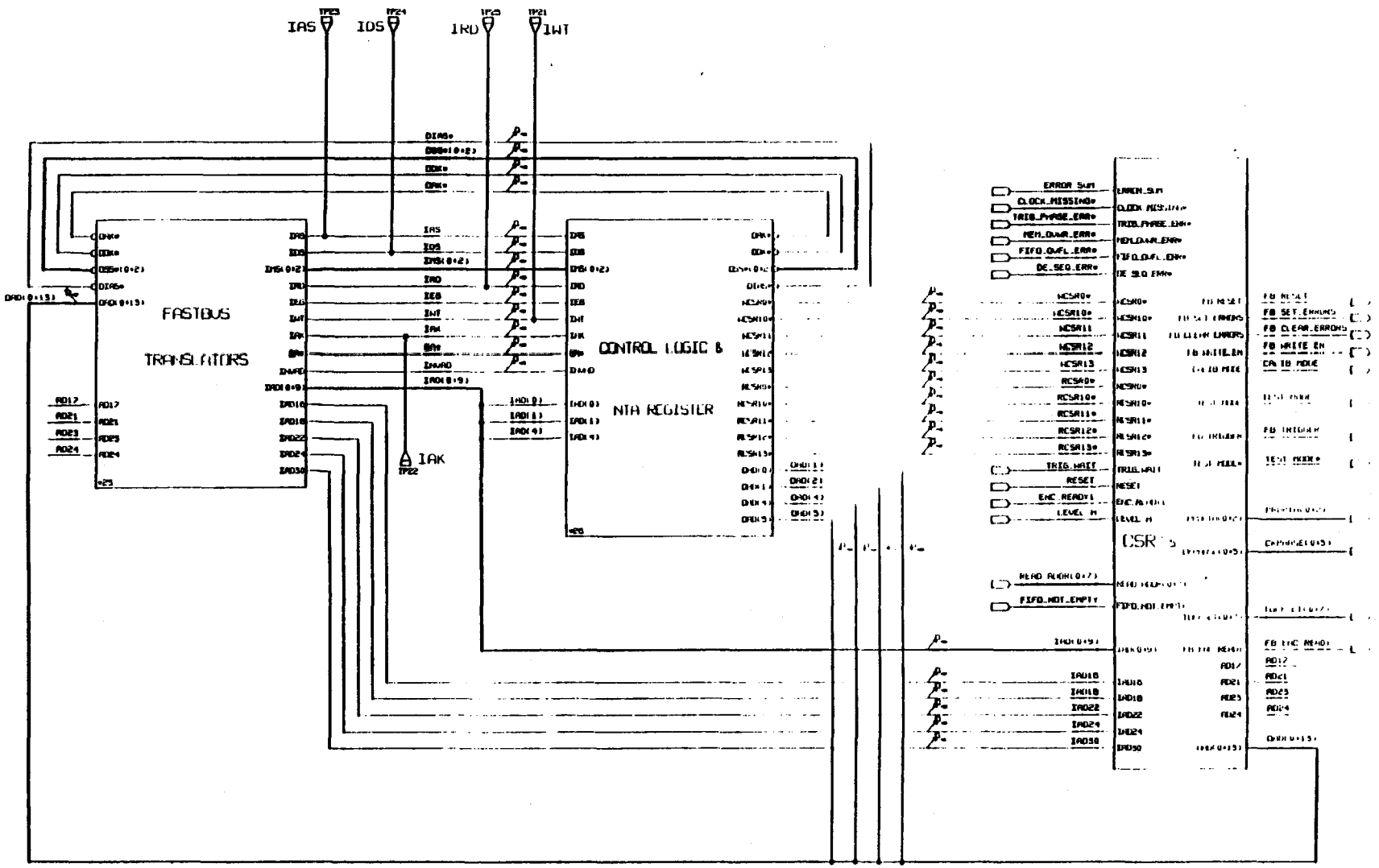
The default values (i.e trigger offset, pipeline depth,...) can be modified for each test if the program is being runned in the interactive mode.

Refer to document _____ for a complete description of the MTC Test Software.

APPENDIX A

CIRCUIT DIAGRAMS


FASTBUS INTERFACE
SHEETS 2,3,4,5
MTC AUX LOG
FRONT PANEL INTERFACE
SHEET 7
SHEET 9
SYNC128
TP19
WRITE_EN
TP17
ADDR_VALID
TP18
TRIGGER
TP20
RESET
LEVEL_IN
READ_ADDR(0:7)
ENC_RDY
TP13
CLOCK GENERATION SYSTEM SYNCHRONIZ
SHEET 6
TRIGGER FIFO
SHEET 8
FIFO NOT EMPTY
TP14
FIFO_WRITE
TP16
FIFO_READ
TP15
TOFFSET(0:7)
FDP(0:3)
PROPTM(0:2)
DXPHASE(0:5)
TRIG_WIND(0:7)
SEQ_READY
DE_SEQ_ERR
TRIGGER
CLOCK
SYNC
SYNC128
FB_RESET
FB_TRIGGER
TEST_MODE
ERROR_SUM
FIFO_NOT_EMPTY
FIFO_WRITE
FIFO_READ
PERMI NATIONAL ACCELERATOR CORPORATION
DETECTOR ELECTRONICS SYSTEMS GROUP
MTC BLOCK DIAGRAM
USED ON: MTC
FILE: MTC
D
REV D



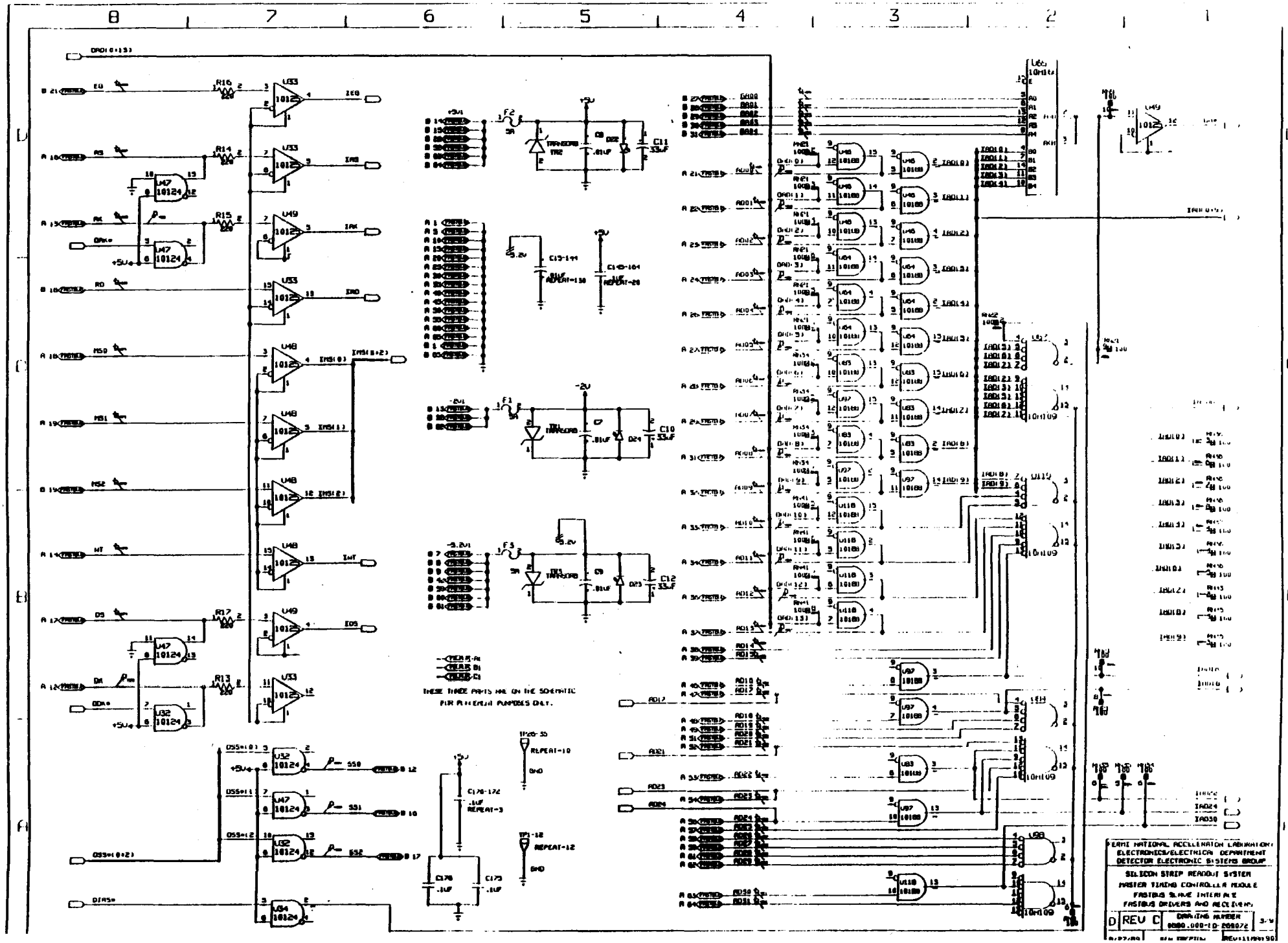
FERMI NATIONAL ACCELERATOR LABORATORY
ELECTRONICS/ELECTRICAL DEPARTMENT
DETECTOR ELECTRONIC SYSTEMS

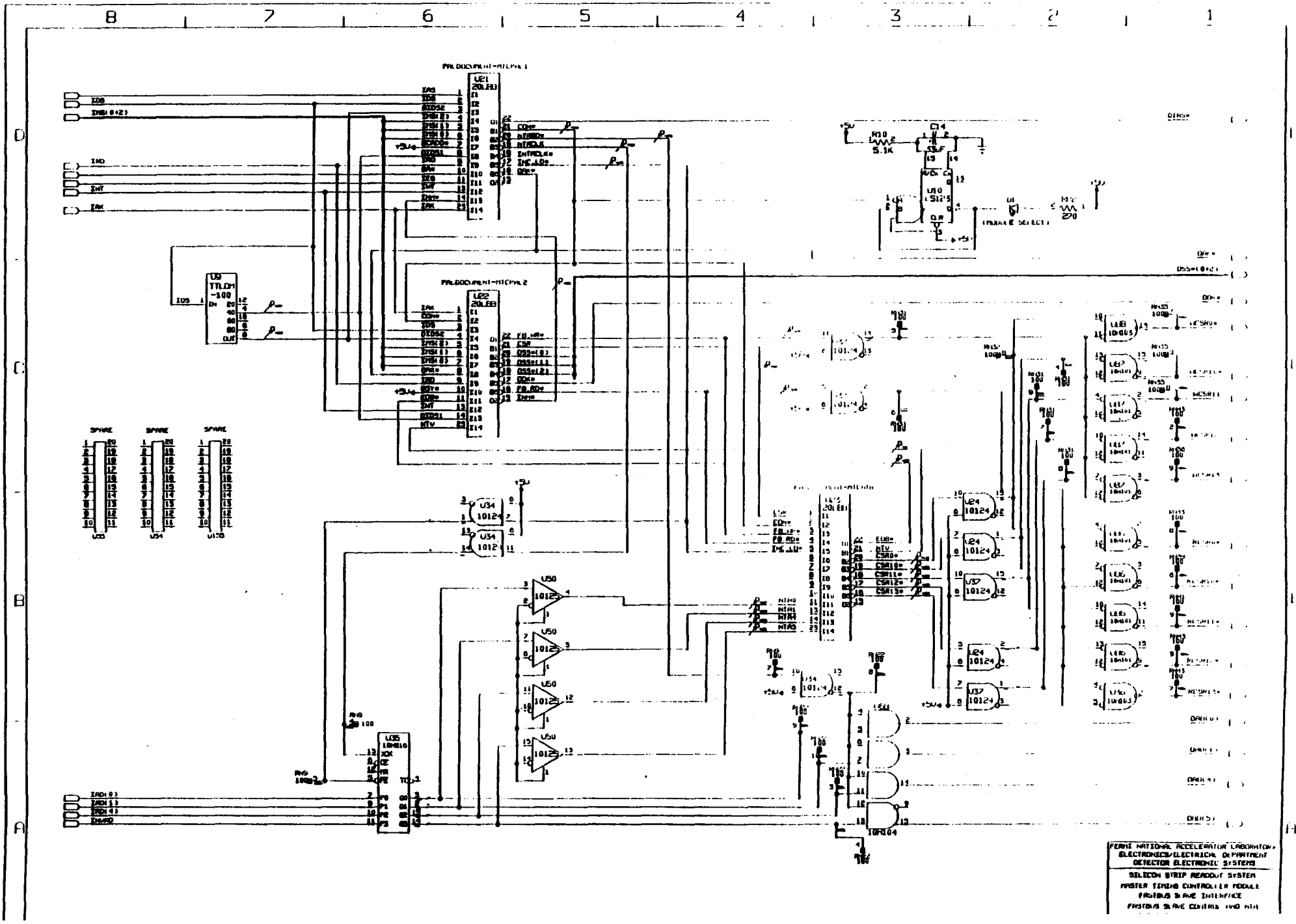
SILICON STRIP READOUT SYSTEM

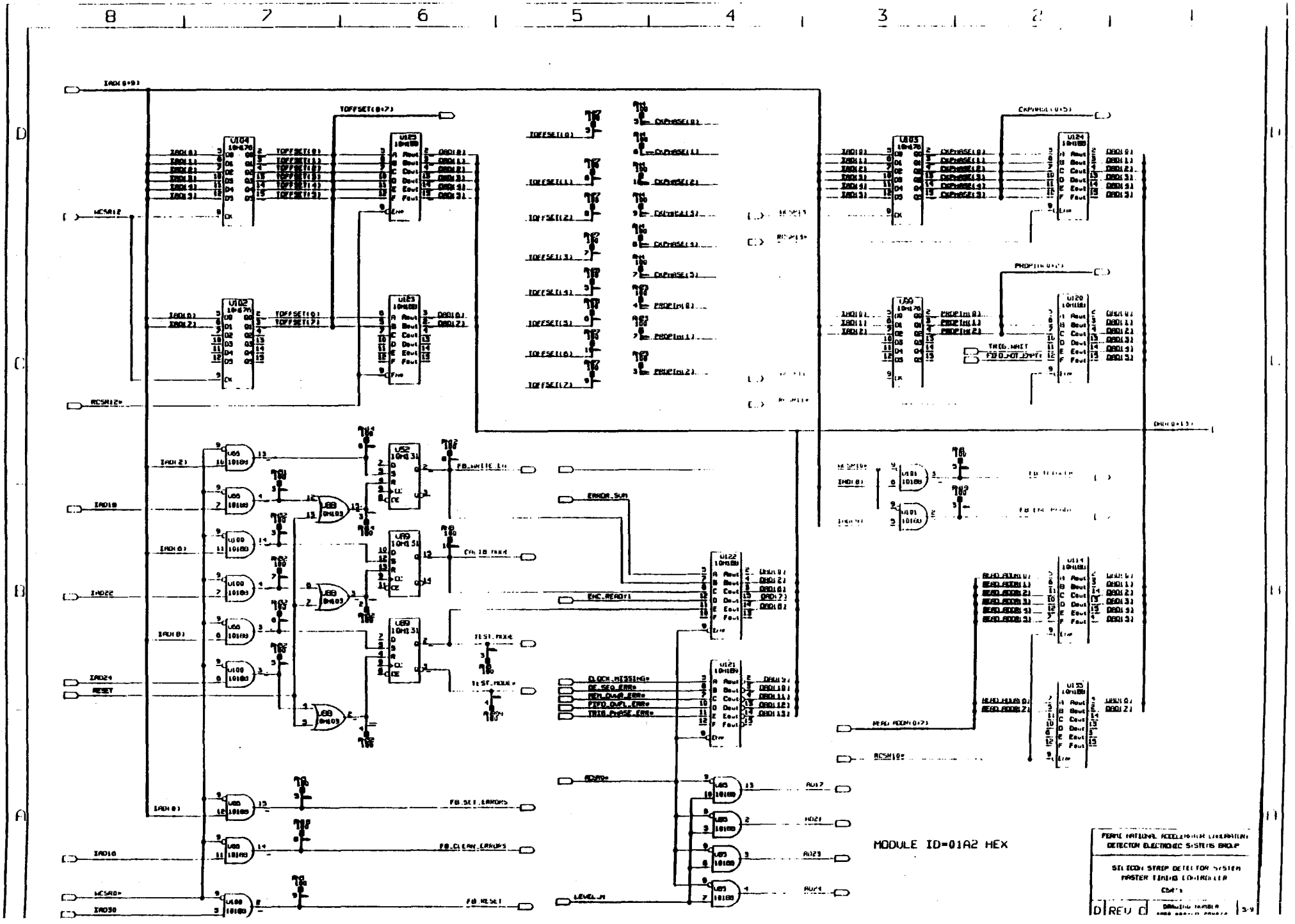
MASTER TIMING CONTROLLER

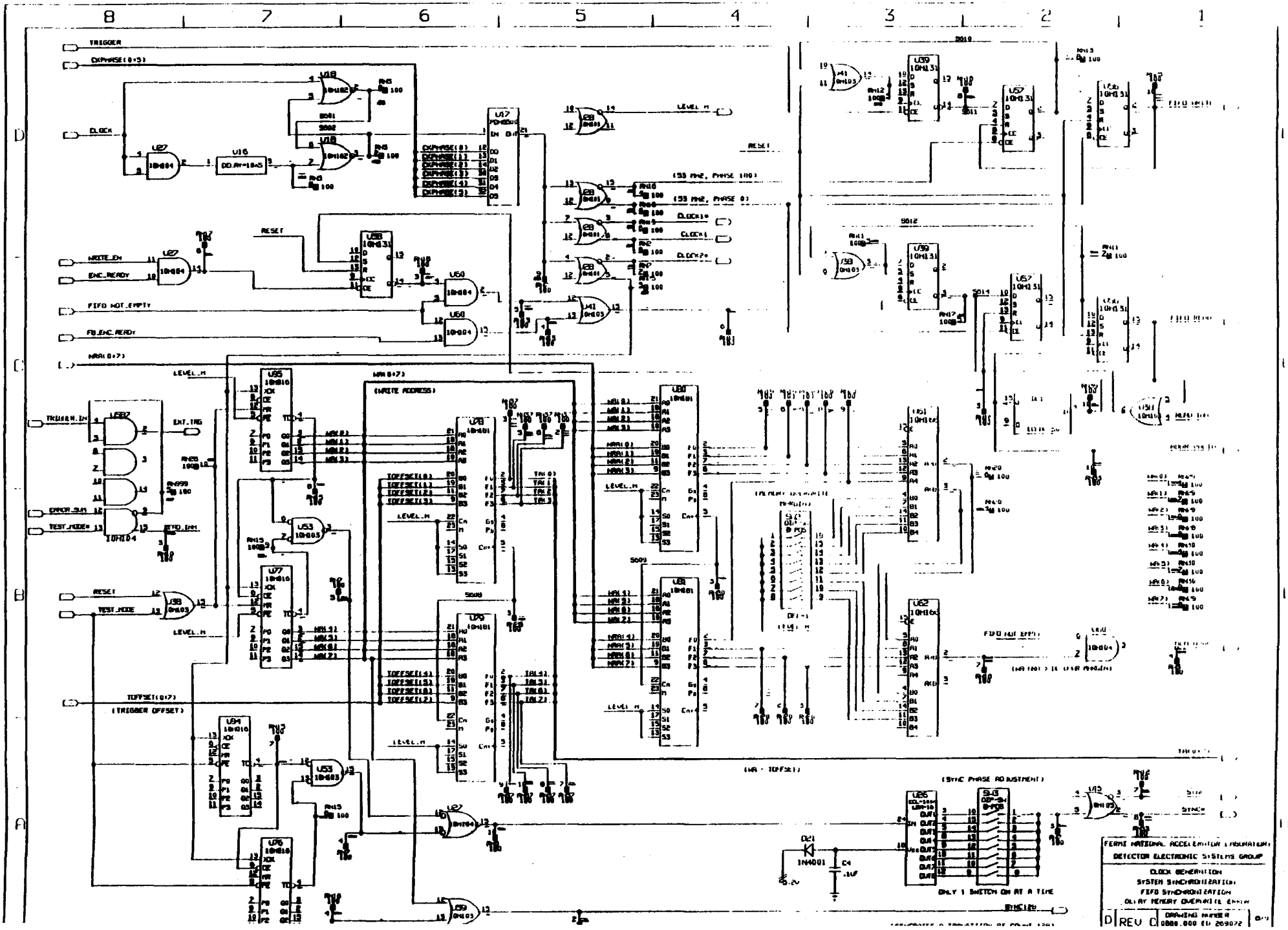
FASTBUS SLAVE INTERFACE

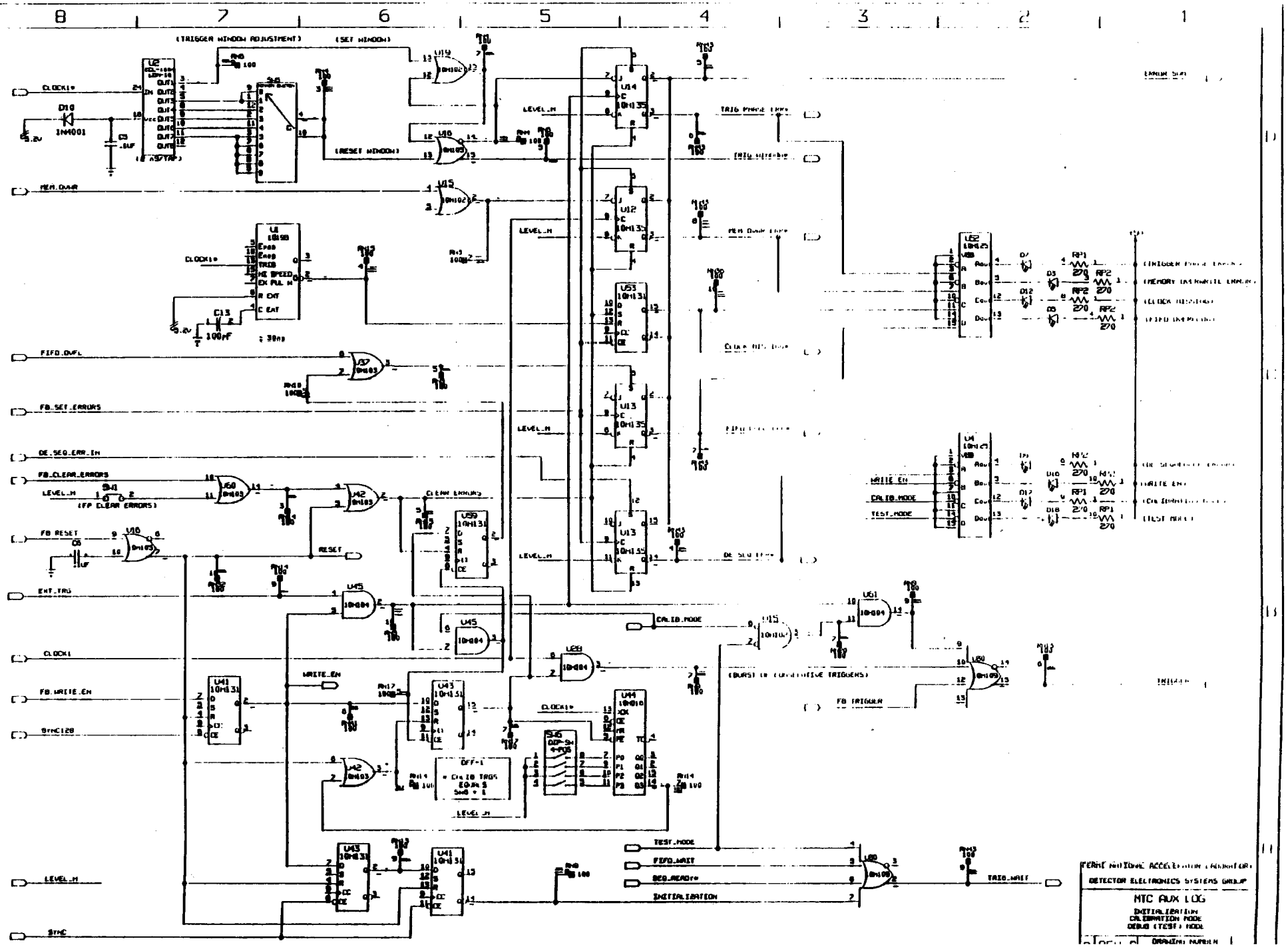
D | REV C | DRAWING NUMBER 8000.000-LD-20907C | 2 of 9

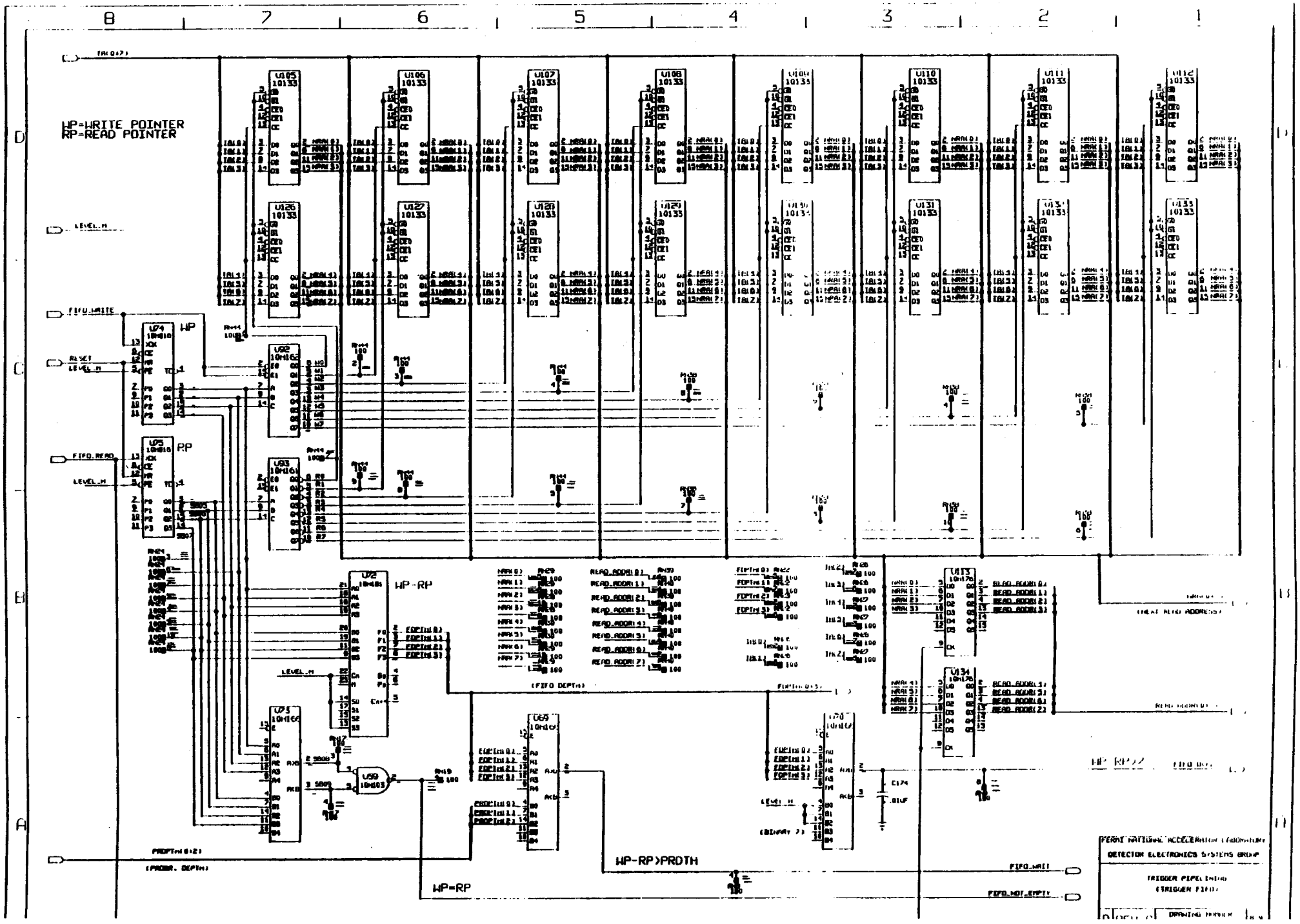

FERMI NATIONAL ACCELERATOR LABORATORY
ELECTRONICS/ELECTRICAL DEPARTMENT
DETECTOR ELECTRONIC SYSTEMS GROUP
SILICON STRIP READOUT SYSTEM
MASTER TIMING CONTROLLER MODULE
FASTBUS SLAVE INTERFACE
FASTBUS DRIVERS AND RECEIVERS
REV C
DRAWING NUMBER
9990.000-ED-268072
THESE THREE PARTS ARE ON THE SCHEMATIC FOR PLACEMENT PURPOSES ONLY.

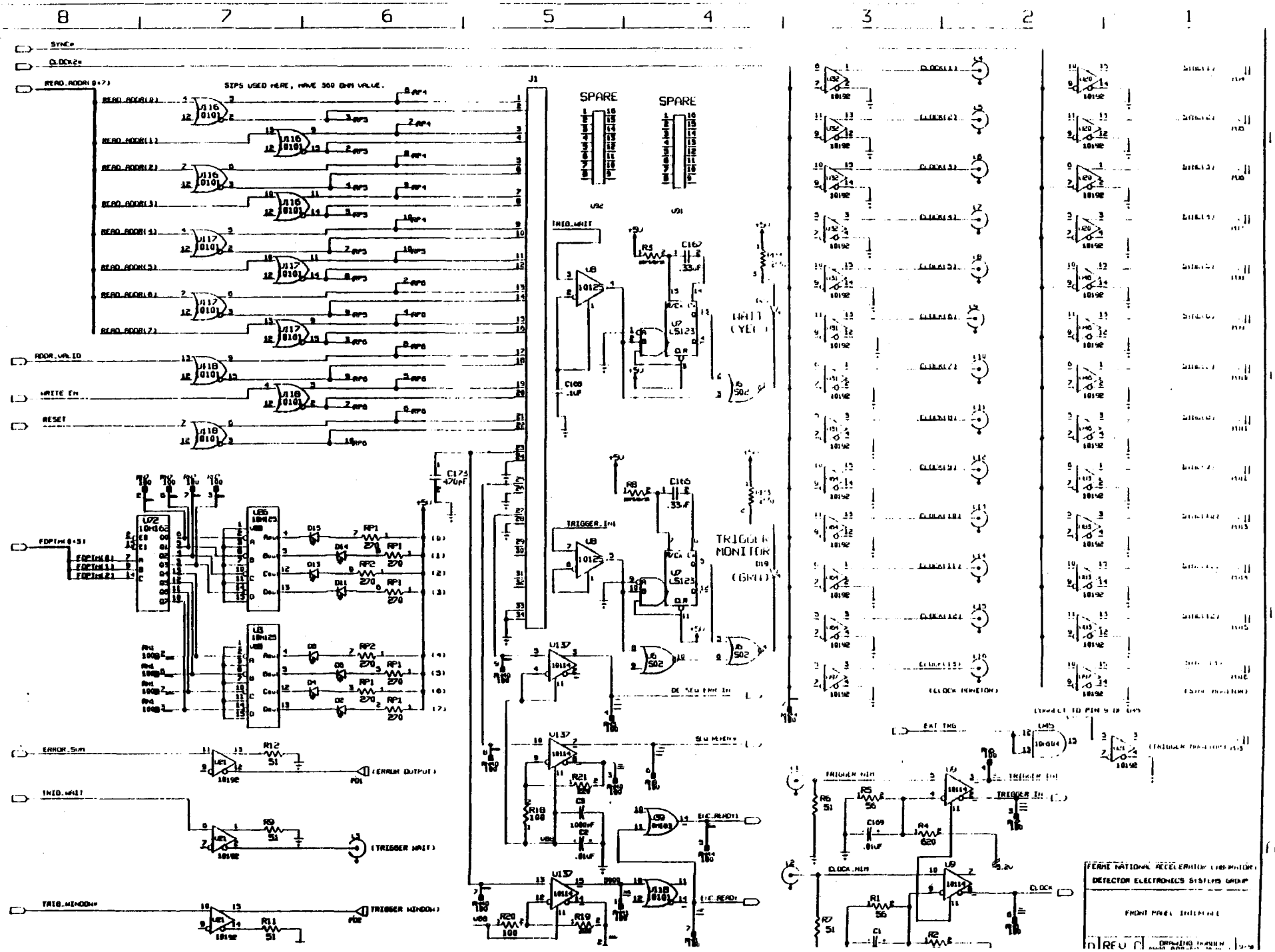

8
7
6
5
4
3
2
1
D
C
B
A
SPARE
SPARE
SPARE
FERMI NATIONAL ACCELERATOR LABORATORY
ELECTRONICS/ELECTRICAL DEPARTMENT
DETECTOR ELECTRONIC SYSTEMS
SILICON STRIP READOUT SYSTEM
MASTER TIMING CONTROLLER MODULE
FASTBUS SLAVE INTERFACE
FASTBUS SLAVE CONTROL AND MTM


MODULE ID=01A2 HEX
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONIC SYSTEMS GROUP
SILICON STRIP DETECTOR SYSTEM
FASTER TIMING CONTROLLER
CSFT
REV D


TRIGGER
DLOCK
RESET
WRITE_EN
ENC_READY
FIFO NOT_EMPTY
FB_ENC_READY
LEVEL_H
TRIGGER_IN
EXT_TRG
ERROR_OUT
TEST_MODE
TOFFSET(0:7)
(TRIGGER OFFSET)
(WRITE ADDRESS)
(SYNC PHASE ADJUSTMENT)
ONLY 1 SWITCH ON AT A TIME
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONIC SYSTEMS GROUP
CLOCK GENERATION
SYSTEM SYNCHRONIZATION
FIFO SYNCHRONIZATION
DELAY MEMORY OVERWRITE ERROR
DRAWING NUMBER
REV D


8
7
6
5
4
3
2
1
(TRIGGER WINDOW ADJUSTMENT)
(SET WINDOW)
(RESET WINDOW)
CLOCK1*
1N4001
REM.OVWR
FIFO.OVFL
FB.SET.ERRORS
DE.SEQ.ERR.IN
FB.CLEAR.ERRORS
(FP CLEAR ERRORS)
FB RESET
RESET
EXT.TRG
CLOCK1
FB.WRITE.EN
SYNC128
LEVEL.H
SYNC
WRITE.EN
CALIB.MODE
TEST.MODE
FIFO.WAIT
DEQ.READY*
INITIALIZATION
TRIG.WAIT
FB TRIGGER
(BURST OR CONTINUOUS TRIGGERS)
NTC AUX LOG
INITIALIZATION
CALIBRATION MODE
DEBUG (TEST) MODE
DETECTOR ELECTRONICS SYSTEMS GROUP


WP=WRITE POINTER
RP=READ POINTER
FIFO_WRITE
RESET
LEVEL_H
FIFO_READ
WP
RP
WP-RP
WP=RP
WP-RP>PRDTH
FIFO_WAIT
FIFO_NOT_EMPTY
(PRDTH-1:0)
(PROGR. DEPTH)
(FIFO DEPTH)
(NEXT READ ADDRESS)
(BINARY 7)
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP
TRIGGER PIPELINE
(TRIGGER FIFO)


SPARE
SPARE
TRIGGER MONITOR
(GMT)
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP
FRONT PANEL INTERFACE

APPENDIX B

PAL EQUATIONS USED IN THE FASTBUS INTERFACE

MODULE MTC_NTA

```
TITLE       'FASTBUS NTA decode PAL for SSD Master Timing Controller Module
             Ken Treptow -- FERMILAB
             Sep 25, 1989 -- Revised Sep 25, 1989 '

            MTCNTA   DEVICE          'P20V8C'; " normally P20L8

"Inputs:

            CSR,!CON,!FB_WR,!FB_RD,INC      PIN 1,2,3,4,5  ;
            nc6,nc7,nc8,nc9,nc10            PIN 6,7,8,9,10 ;
            NTA0,NTA1,NTA4,NTA5             PIN 11,13,14,23  ;

"Outputs:

            !EOB,NTV,!CSR0,!CSR10           PIN 22,21,20,19  ;
            !CSR11,!CSR12,!CSR13,nc15       PIN 18,17,16,15  ;

"Constant declarations:

            X = .X. ;

            ADDR = [NTA5,NTA4,X,X,NTA1,NTA0] ;


Equations

            NTV = CON & CSR & !((ADDR == 0)                 " Not CSR0
              # ((ADDR >= ^h10) & (ADDR <= ^h13))) ;         " Not CSR10-13

            CSR0 = CON & CSR & (ADDR == 0) ;

            CSR10 = CON & CSR & (ADDR == ^h10) ;

            CSR11 = CON & CSR & (ADDR == ^h11) ;

            CSR12 = CON & CSR & (ADDR == ^h12) ;

            CSR13 = CON & CSR & (ADDR == ^h13) ;

            EOB =  CON & CSR & INC & NTV ;

END MTC_NTA
```

MODULE MTC_PAL1

```
TITLE 'FASTBUS Slave PAL1 forr SSD Master Timing Controller Module
       This is a modified FASTBUS Slave PAL1 from E706s ICBM Module
       Ken Treptow -- FERMILAB
       Aug 25, 1989  -- Revised Aug 25, 1989 '

"Note this is a modification of FASTBUS Slave PALS done by L. PREGERNIG
" when he was at the UNIVERSITY OF ILLINOIS HIGH ENERGY PHYSICS GROUP
"DATE          1986 OCTOBER 20
"CHIP          FB009_SCL2    PAL20L8

       MTCPAL1      DEVICE       'P20V8C';  " normally P20L8

"Inputs:

       IAS,IDS,DIDS2,IMS2,IMS1,IMS0                PIN 1,2,3,4,5,6 ;
       !BCADD,DIDS1,IRD,!GA                        PIN 7,8,9,10 ;
       IEG,IWT,!INH,IAK                            PIN 11,13,14,23 ;


"Outputs:

       !DIAS,!CON,!NTARD,NTACLK                    PIN 22,21,20,19 ;
       !INTACLK,INC_LD,!OAK,nc                     PIN 18,17,16,15 ;


equations

 DIAS = IAS ; "Delayed Input AS

"CONnected (attached)

 CON = IAS & !IAK & !IRD & !IMS2 & !IMS1 & IMS0 & GA & IEG     "Geographical Add CS
       # IAS & !IAK & !IRD & !IMS2 & IMS1 & BCADD               "Broadcast Address
       # IAS & CON  ;                                          "latch while AS is up

"NTA ReaD

 NTARD = CON & IRD & !IMS2 & IMS1 & !IMS0 & IDS & !DIDS2       "set on DS up
         # NTARD & CON & IRD & !IMS2 & IMS1 & !IMS0 & IDS & DIDS2 ;
                                          "latch until new MS or Write and DS,DK up

"NTA CLocK

 NTACLK = INTACLK
          # INC_LD & !IAS                 " Inc NTA when terminating Blk xfer with AS dn
          # NTACLK & IDS & !DIDS2         "Latch while DS up
          # NTACLK & !IDS & DIDS2 ;       "Latch while DS down

"Internal NTA CLocK
```

(cont. MTC_PAL1)...

```
INTACLK = CON & !INC_LD & !IRD & !IMS2 & IMS1 & !IMS0 & DIDS1 & !DIDS2 & !NTACLK
            "NTA write cycle
          # CON & INC_LD & !INH & !IMS2 & IMS0 & IDS & !DIDS2 & !NTACLK
            "Block transfer DS up
          # CON & INC_LD & !INH & !IMS2 & IMS0 & !IDS & DIDS2 & !NTACLK
            "Block transfer DS dn
          # CON & INC_LD & !INH & IRD & !IMS2 & IMS1 & !IMS0
            & IDS & !DIDS2 & !NTACLK                    "NTA read cycle
          # CON & INC_LD & !INH & !IMS2 & !IMS1 & !IMS0
            & IDS & !DIDS2 & !NTACLK;                   "Single R/W cycle

"INCrement or LoaD the NTA

INC_LD = !(!(CON & !INH & IMS0 & IDS & DIDS2) & !INC_LD     "Set if blk
         # !IRD & !IMS2 & IMS1 & !IMS0 & IDS & !DIDS2       "Reset on NTA write
         # !IMS2 & !IMS0 & NTACLK                  "Reset on NTA or Single & after NTACLK
         # !CON ) ;                                "Reset when disconnected

"Output AK

OAK = CON & !IAK & !IWT & !BCADD                    "Send AK if not broadcast address
      # OAK & CON                                   "Latch until CON goes away
      # OAK & IWT                                   "Hold if WaiT is asserted
      # OAK & DIDS2 ;                               "Stretch until DK is off


END MTC_PAL1
```

MODULE MTC_PAL2

```
TITLE   FASTBUS Slave PAL2 for SSD Sequencer Module
        This is a modified FASTBUS Slave PAL2 from E706s ICBM Module
        Ken Treptow -- FERMILAB
        Aug 28, 1989 -- Revised Aug 28, 1989 '

"Note this is a modification of FASTBUS Slave PALS done by L. PREGERNIG
" when he was at the UNIVERSITY OF ILLINOIS HIGH ENERGY PHYSICS GROUP
"DATE            1986 OCTOBER 20
"CHIP            FB010_SCL2     PAL20L8


        MTCPAL2         DEVICE          'P20V8C';  " normally P20L8

"Inputs:

        IAK,!CON,IDS,DIDS2,IMS2,IMS1,IMS0               PIN  1,2,3,4,5,6,7 ;
        !OAK,IRD,!BSY,!EOB,IWT,DIDS1,NTV                PIN  8,9,10,11,13,14,23 ;


"Outputs:

        !FB_WR,CSR,!OSS0,!OSS1,!OSS2                    PIN  22,21,20,19,18 ;
        !ODK,!FB_RD,!OINH                               PIN  17,16,15 ;

"Constant declarations

"INHibit data transfers

 INH = CON & EOB & !IMS2 & IMS0          "If End Of Block is reached (block or pipeline)
       # CON & BSY                        "when BuSY
       # CON & NTV                        "when NoT Valid address is in the NTA
       # IMS2  ;                          "when bad MS code


Equations

"FastBus WRite strobe

 FB_WR = CON & !INH & !IRD & !IMS2 & !IMS1 & !IMS0 & DIDS1 & !DIDS2
              "MS=0 random data write, DS up only
       # CON & !INH & !IRD & !IMS2 & IMS0 & DIDS1 & !DIDS2
              "MS=1 block transfer write, DS up
              "MS=3 pipeline transfer write, DS up
       # CON & !INH & !IRD & !IMS2 & IMS0 & !DIDS1 & DIDS2  ;
              "MS=1 block transfer write, DS down
              "MS=3 pipeline transfer write, DS down

"Addressed in CSR

 CSR = CON & IMS0 & !IAK                 "set if CSR at primary address time
       # CSR & CON  ;                     "latch until end of CONnected (attached)


"Output Slave Status bit 0
```

(cont MTC_PAL2)

```
OSS0 = CON & BSY & !IMS2 & !IMS1 & !IMS0 & DIDS1 & !DIDS2      "BuSY & single xfer
     # CON & BSY & !IMS2 & IMS0 & DIDS1 & !DIDS2       "BuSY and MS=1 or 3 on DS up
     # CON & BSY & !IMS2 & IMS0 & !DIDS1 & DIDS2       "BuSY and MS=1 or 3 on DS dn
     # CON & NTV & !IMS2 & IMS1 & !IMS0 & DIDS1 & !DIDS2      "NTV & 2nd Address
     # OSS0 & IDS & DIDS2 & CON                        "Latch while DS=DK=1
     # OSS0 & !IMS2 & IMS0 & !IDS & !DIDS2 & CON ;     "Latch if DS=DK=0 & MS=1or3

"Output Slave Status bit 1
OSS1 = CON & IMS2   "any MS=4-7 on DS up or down
     # CON & !IMS2 & NTV & !BSY & DIDS1 & !DIDS2
     "any Not Valid address if not BuSY on DS up
     # CON & !IMS2 & IMS0 & NTV & !EOB & !BSY & !DIDS1 & DIDS2
     "any Not Valid address, not BuSY, not End Of Block, & MS=1or3 on DS down
     # CON & !IMS2 & IMS0 & EOB & !BSY & DIDS1 & !DIDS2
     "End Of Block if not BuSY on DS up
     # CON & !IMS2 & IMS0 & EOB & !BSY & !DIDS1 & DIDS2
     "End Of Block if not BuSY on DS down
     # OSS1 & IDS & DIDS2 & CON                        "Latch while DS=DK=1
     # OSS1 & IMS0 & !IDS & !DIDS2 & CON ;             "Latch if DS=DK=0 & MS=1,3,5,or 7

"Slave Status bit 2
OSS2 = CON & IMS2                                      "any MS=4-7 on DS up or down
     # CON & !IMS2 & NTV & !EOB & !BSY & DIDS1 & !DIDS2
     "any Not Valid address if not BuSY and not End Of Block on DS up
     # CON & !IMS2 & IMS0 & NTV & !EOB & !BSY & !DIDS1 & DIDS2
     "any Not Valid address, not BuSY, not End Of Block, & MS=1or3 on DS down
     # OSS2 & IDS & DIDS2 & CON                        "Latch while DS=DK=1
     # OSS2 & IMS0 & !IDS & !DIDS2 & CON ;             "Latch if DS=DK=0 & MS=1,3,5,or 7

"Output Data acKnowiege generates DK

ODK = !IWT & CON & OAK & DIDS2              "set if DS (delayed) and attached and not WaiT
     # IMS1 & IMS0 & CON & OAK & DIDS2
          "set if DS (delayed) and attached and MS=3 (pipeline) even if WaiT
     # ODK & CON & OAK & DIDS2               "transition hold while DS (delayed)
     # IWT & !IMS1 & ODK                     "hold if WaiT and not MS1
     # IWT & !IMS0 & ODK ;                   "hold if WaiT and not MS0
          "i.e. hold if not MS=3 (pipeline) AND WaiT, release it otherwise

"FastBus ReaD

FB_RD = CON & !INH & IRD & !IMS2 & !IMS1 & !IMS0 & DIDS1 & !DIDS2
          "set on DS up, MS=0 random data read
     # CON & !INH & IRD & !IMS2 & IMS0 & DIDS1 & !DIDS2
          "set on DS up, MS=1 block read
          "set on DS up, MS=3 pipeline read
     # CON & !INH & IRD & !IMS2 & IMS0 & !DIDS1 & DIDS2
          "set on DS dn, MS=1 block read
          "set on DS dn, MS=3 pipeline read
     # CON & IRD & !IMS2 & !IMS1 & !IMS0 &IDS & DIDS2 & FB_RD
          "latch while MS=0 read and DS,DK up
     # CON & IRD & !IMS2 & IMS0 & FB_RD
          "latch while MS=1,3 read
     # CON & IDS & DIDS2 & FB_RD                  "latch while DS,DK up
     # CON & !IDS & !DIDS2 & FB_RD ;              "latch while DS,DK down
                    "i.e. latch until new MS or WR cycle
```

(cont MTC_PAL2)

"Output INHibit data transfers

```
OINH = INH
        # CON & !CSR ;        "This stops NTA Incrementing for the FIFO in Data Space

END MTC_PAL2
```

APPENDIX C

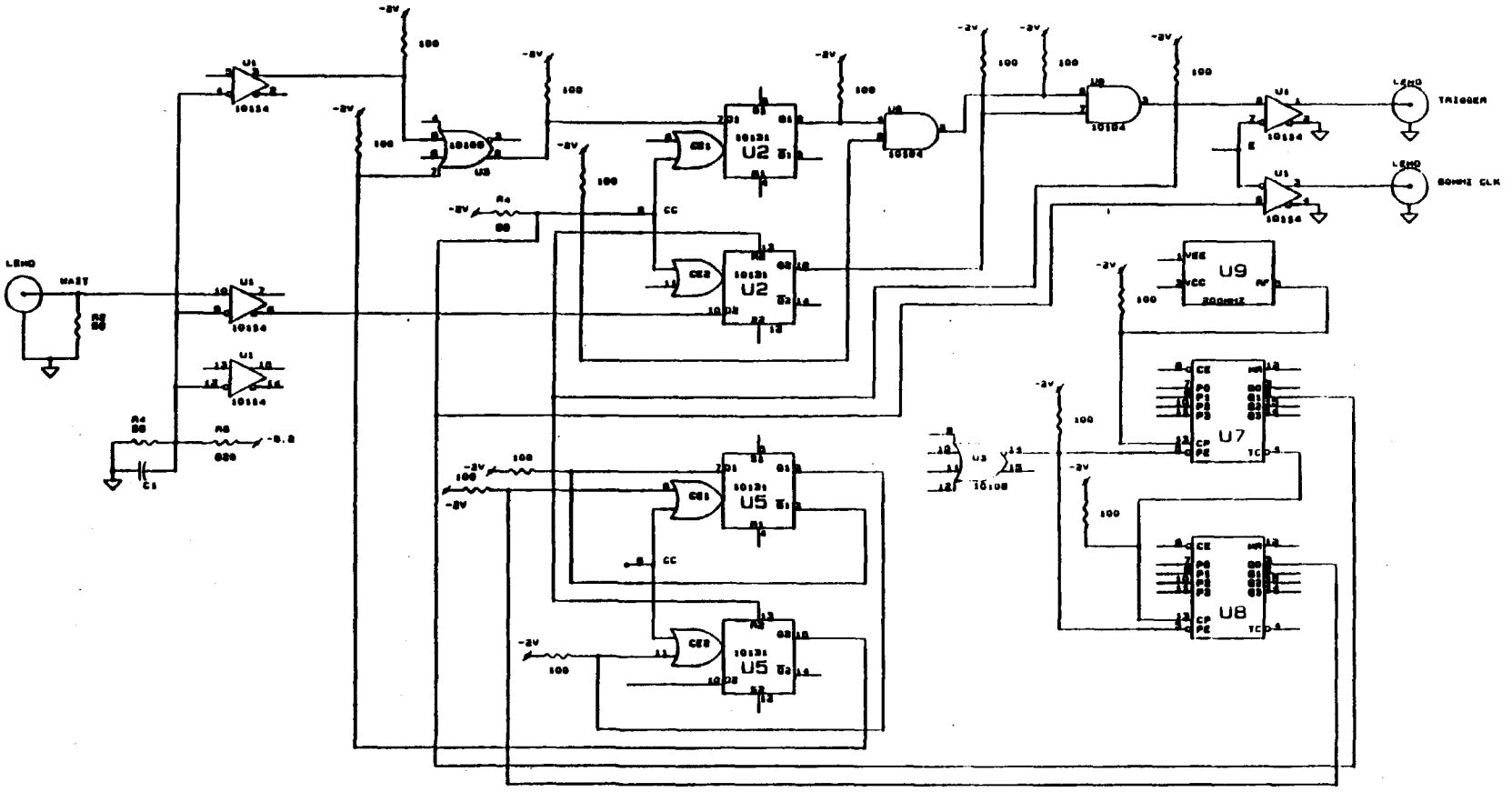
PARTS LIST

No.	MTC Part No.	Manuf	Manuf PART No.	FNAL Stock No.	Description	Qty	Cost	Cost/brd
			Resistor					
1	R3,8	AB		1487-0385	10K 1/8W Res	1	0.13	0.13
2	R6,7,9,11,12			1478-0247	51 ohm 1/8w Res	5	0.16	0.80
3	R13-R17,19,21			1487-0285	220 ohm 1/8w Res	7	0.16	1.12
4	R18,20			1487-0620	100 ohm 1/4w Res	2	0.06	0.12
5	R1,R5			1487-0590	56 ohm 1/4w Res	2	0.06	0.12
6	R10			1487-0830	5.1K ohm 1/4w Res	1	0.06	0.06
7	R2,4			1487-0720	620 ohm 1/4w Res	2	0.04	0.08
8	RN1-RN44	AB	4610X-101		100 ohm 10-pin SIP	44	0.32	14.08
9	RP1-RP3	AB	4610X-101		270 ohm 10-pin SIP	3	0.32	0.96
10	RP4-RP6	AB	4610X-101		330 ohm 10-pin SIP	3	0.32	0.96
			Integrated Circuits					
11	U116-U118	Motorola	10101P	1455-5801	Quad OR/NOR Gate	1	0.31	0.31
12	U9,137	Motorola	10114P		Triple Line Receiver	1	0.76	0.76
13	U25,33,35,38,48	Motorola	10124P	1455-5824	Quad TTL to ECL Translator	5	1.52	7.60
14	U8,34,49,50,51	Motorola	10125P	1455-5825	Quad ECL to TTL Translator	5	1.22	6.10
15	U106-U113 U127-U134	Motorola	10133p	1455-5833	Quad latch	16		
16	U47,65,67,84,86 U98,101,102 U119	Motorola	10188P		Hex Buffer w/enable	9	2.20	19.80
17	U20,21,31,32,46 U64,83,97	Motorola	10192P		Quad Bus Driver (ECL to Nim)	8	3.90	31.20
18	U1	Motorola	10198P		Monostable Multivibrator	1	8.85	8.85
19	U29,87,88	Motorola	10H101P		Quad OR/NOR Gate	3	0.64	1.92
20	U15,19	Motorola	10H102P		Quad 2 Input NOR Gate	2	0.64	1.28
21	U37,39,42,54,60,89	Motorola	10H103P		Quad 2 Input OR Gate	6	0.64	3.84
22	U28,45,61	Motorola	10H104P		Quad 2 Input AND	4	0.64	2.56
23	U16	Motorola	10H105P		Triple 2-3-2 Input OR/NOR	1	0.64	0.64
24	U30,68,85,99,120	Motorola	10H109P		Dual 5-4 Input OR/NOR	5	1.49	7.45
25	U75-U78 U36,44,95,96	Motorola	10H016P		4-Bit Binary counter	8	7.46	59.68
26	U3,4,26,52	Motorola	10H125P		Quad ECL to TTL Translator	4	2.00	8.00
27	U40,41,43,53,57	Motorola	10H131P		Dual D type F/F	8	1.90	15.20
28	58,59,90, U12-U14	Motorola	10H135P		Dual JK MS Flip Flop	3	2.30	6.90
29	U94	Motorola	10H161P		Binary TO 1-8 Decoder (Low)	1	2.44	2.44
30	U72,93	Motorola	10H162P		Binary TO 1-8 Decoder (High)	2	2.44	4.88
31	U62,63,66,70,71,74	Motorola	10h166P		5-Bit magnitude comparator	6	3.24	19.44
32	U100,114,135 U103-U105	Motorola	10H176P		Master-slave flip-flop	6	3.54	21.24
33	U79-U82,73	Motorola	10H181P		4-Bit alu/function generator	5	9.95	49.75
34	U123-U126 U115,121,136	Motorola	10H188P		Hex Buffer w/enable	1	1.45	1.45
35	U122	Motorola	10H189P		Hex Inverter w/enable	1	2.20	2.20
36	U7,U11	T.I.	74LS123N	1455-8123	Dual Retriggerable Monostable	2	0.59	1.18
	U6	T.I.	74LS02	1455-8002	Nor gate	1	0.23	0.23
			Switches					
37	SW2,SW3	CTS Corp.	P/N 206-8s	1455-9708	8 section dip switch	2	0.89	1.78
	SW5	C&K	3M120		10 Pos. thumbwheel switch CW811	1	4.65	4.65
38	SW6	CTS Corp.	P/N 206-4s	1455-9704	4 section dip switch	1	0.75	0.75
39	SW1	C&K	MODEL TP11		tiny pushbutton switch	1	4.05	4.05
			Connectors					
40	L1-L16	KINGS	K-LOCK P/N1077-3	1435-4400	Lemo PC mont	32	4.89	156.48
41	J1	3M	P/N 3431-5302	1435-7105	34pin right angle header	1	1.61	1.61
42	FBSEG A,FBSEG B	AMP	1-102585-3		FASTBUS 130 SOCKET CONNE	1	7.87	7.87
			Diodes					
43	D2,4,6,8,11,13,14 D15,22,23,24	H.P.	P/N HLMP-1503	1445-0470	green LED	11	0.24	2.64
44	D3,5,7,9,12	H.P.	P/N HLMP-1301	1445-0475	red LED	5	0.24	1.20
45	D16-D20,D1	H.P.	P/N HLMP-1401	1445-0495	yellow LED	6	0.24	1.44
	D10,D21		1N4001	1445-1550	Signal diode	2	0.03	0.06
			Fuse					
46	F1-F3	Littlefuse	type 251005	1120-0250	picofuse 5A	3	0.48	1.44
			Capacitors					
47	C165,C167	ERIE	8131-100-651-334M	1415-3170	.33ufd ceramic cap	2	0.18	0.36

No.	MTC Part No.	Manuf	Manuf PART No.	FNAL Stock No.	Description	Qty	Cost	Cost/brd
48	C13	SPRAGUE	P/N 10TS-T10	1415-2110	100pfd	1	0.14	0.14
49	C173	SPRAGUE	P/N 10TS-T47	1415-2150	470pfd cap	1	0.16	0.16
50	C3	SPRAGUE	P/N 10TS-D10	1415-2170	1000pfd cap	2	0.18	0.36
51	C174	SPRAGUE		1415-2160	680pfd	1	0.06	0.06
52	C10-C12,14	MALLORY	CSR13-C336K	1425-1180	33ufd cap	4	0.45	1.80
53	C145-164,C170-172	SPRAGUE	923CZ5U104M050B		.1ufd dip cap	29	0.42	12.18
54	C4,61,68,175,176							
55	C7-C9,C15-C144	SPRAGUE	923cx7r103k050b		.01ufd dip cap	137	0.36	49.32
56	C1,2,169							
57			Delay Line					
58	U8	EC2	TTLDM-100T		DELAY LINE	1	15.20	15.20
59	DL1	EC2	ECLDL-80		DELAY LINE	1	13.20	13.20
60	U2,U25	EC2	ECL-100K-LDM-16		DELAY LINE	2	48.70	97.40
61	U17	ELMEC	PDH6500		DELAY LINE	1	85.00	85.00
62	U16	ELMEC	FDD9010		DELAY LINE	1	9.00	9.00
63								
64			Hardware					
65					MTC front panel 800.000-MD-269	1	40.00	40.00
66	HW				NYLON SPACER .625LONGX4-4	7	0.10	0.70
								812

APPENDIX D

MTC TEST MODULE DIAGRAM



ORIGINATOR	NEEDLES	7/30/90	CHECKED
DRAWN	CUNY	7/30/90	APPROVED
FILE NAME	NEEDLES1		

FERMI NAT'L ACCELERATOR LAB
UNITED STATES DEPT OF ENERGY

RESEARCH DIVISION
MTC TEST MODULE

APPENDIX E

CORRECTIONS TO THE PRINTED CIRCUIT BOARD

Two design changes are responsible for two modifications on the MTC printed circuit board.

The first one is the addition of a NOR gate to invert the signal to the pin 6 of U6 (74S02). This NOR gate is in the same U6 package. The corrections to the board are shown in Fig. E-1 (the "X" represents a cut to a trace, and the hand drawn lines are the added wires).

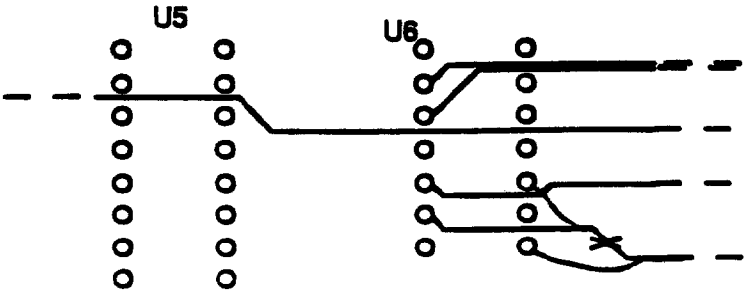


Fig E-1 - Modification number 1 of the MTC'S PCB

In the case of a new PCB design, there is a better way of implementing this logic. A suggestion is given in Fig. E-2, for the trigger rate LED and for the WAIT rate LED in schematic 9/9.

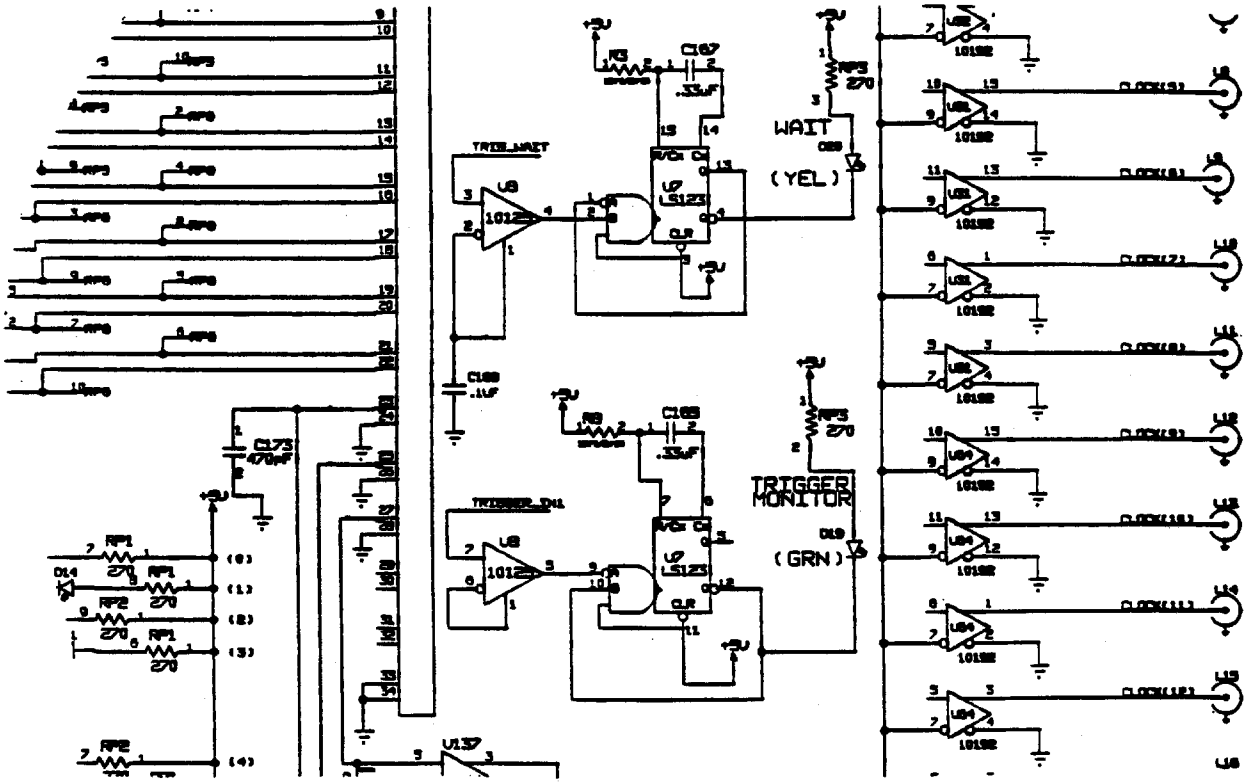


Fig E - 2 - Suggested changes to the TRG IN and TRG WAIT LED driver logic in schematic 9/9

The 2nd change was required to make the front panel TRG MON and TRG WIN signals have the right timing, as observed with a scope, for adjusting the external trigger delay to the module. In order to accomplish this, the trigger signal to the NIM driver was taken from a different point in the circuit (also an extra NAND gate was added, mainly for not messing with the ECL terminations and not to run the signal too long distances; the available gate was found in U45, inputs 12 & 13 of a 10H104, and the input signal to the gate was taken from the same U45, pin 4). The changes are shown in Fig. E-3.

The above changes in the schematics are updated in the respective Cadnetix files. The 2nd modification, however, is not handled by the Cadnetix system, since it makes use a heterogeneous gate in the 10H104 package. A note is posted in the schematic such that new PCB designs will have to find a way to bypass this problem.

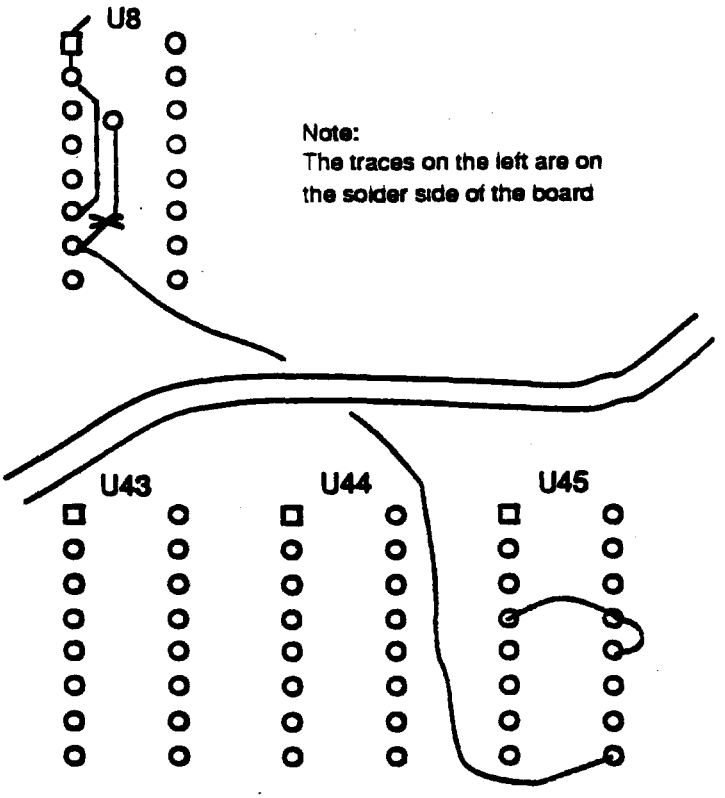


Fig E-3 - Changes to retime the TRG MON signal on the front panel coax connector

Fermilab

April 26, 1990

TO: See Distribution

FROM: Carl Swoboda

SUBJECT: New Version

This is an updated version of Main timing Controllers Specification.

Please replace old version in binders with attached new version of specification.

MASTER TIMING CONTROLLER

for use in the

Silicon Strip Detector Readout System

SPECIFICATION

M. Fachin. C. Rotolo
April 5, 1990

Table of Contents

1. GENERAL

The Master Timing Controller (MTC) module is part of the Silicon Strip Detector System, intended for use in experiments E771 and E789. In addition to generating the system clock coherent to the beam, the MTC provides the mechanism to maintain system synchronization, generates hit addresses, and acts as an interface between the readout system and the 1st level trigger system. The MTC provides a pipeline for triggers and is responsible for system reset (initialization). Figure 1 shows the MTC connections to the Sequencer modules and to the trigger system.

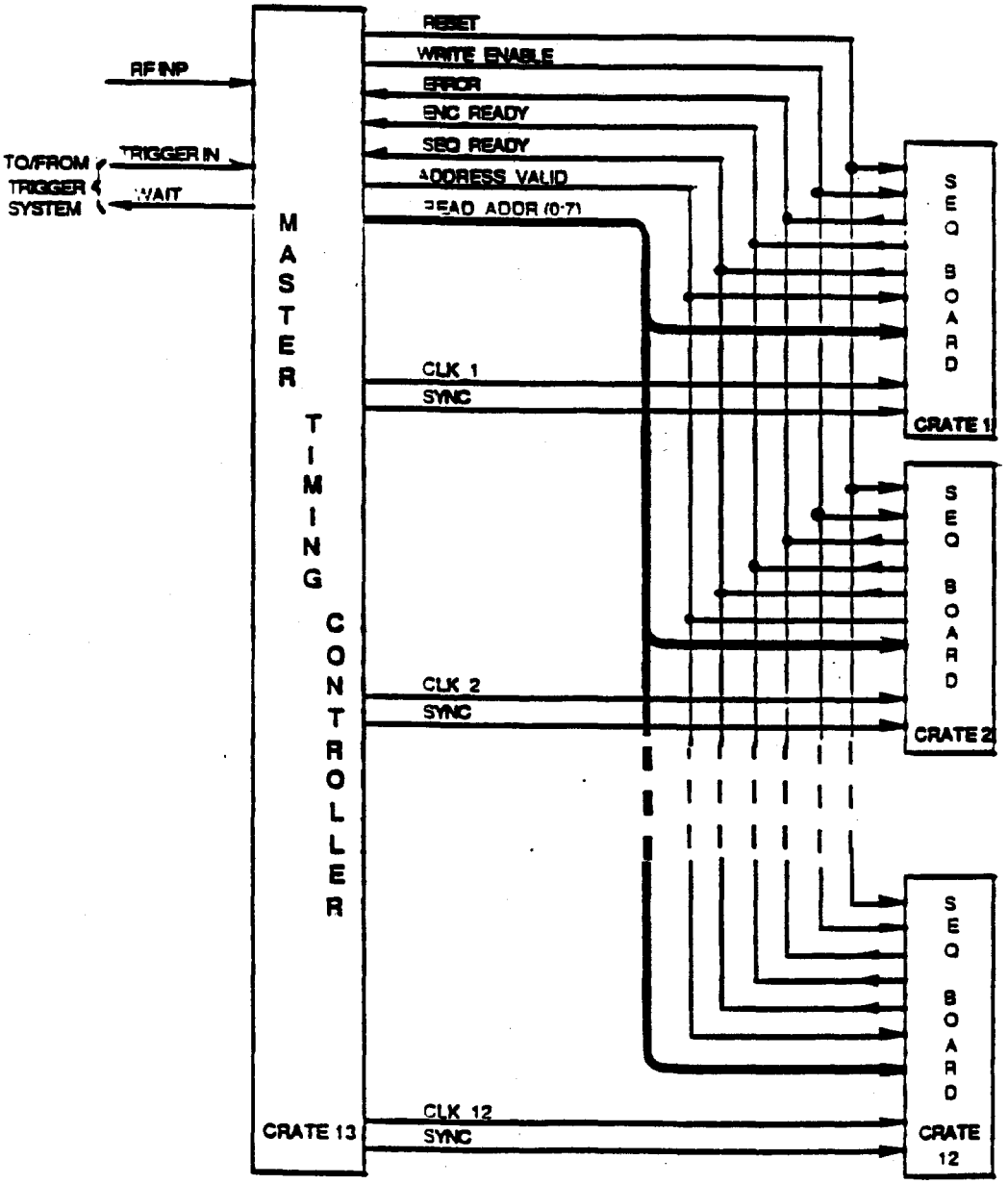


Fig. 1 - MTC connections to the Sequencers and 1st level trigger system.

2. SYSTEM DESCRIPTION

The MTC plays a central role in the control of the SSD readout system. It throttles the trigger rate based on maximum system throughput capability and broadcasts hit addresses in response to trigger requests.

Upon receiving a trigger pulse from the 1st level trigger system, the MTC sends the corresponding hit address to the Sequencers, which relay it to the D/Es. Trigger requests pending in the MTC due to trigger pipelining are forwarded only after all D/Es in the system have encoded the previous trigger. The MTC also incorporates feaures for system calibration and debugging. Figure 2 shows the MTC's internal functions.

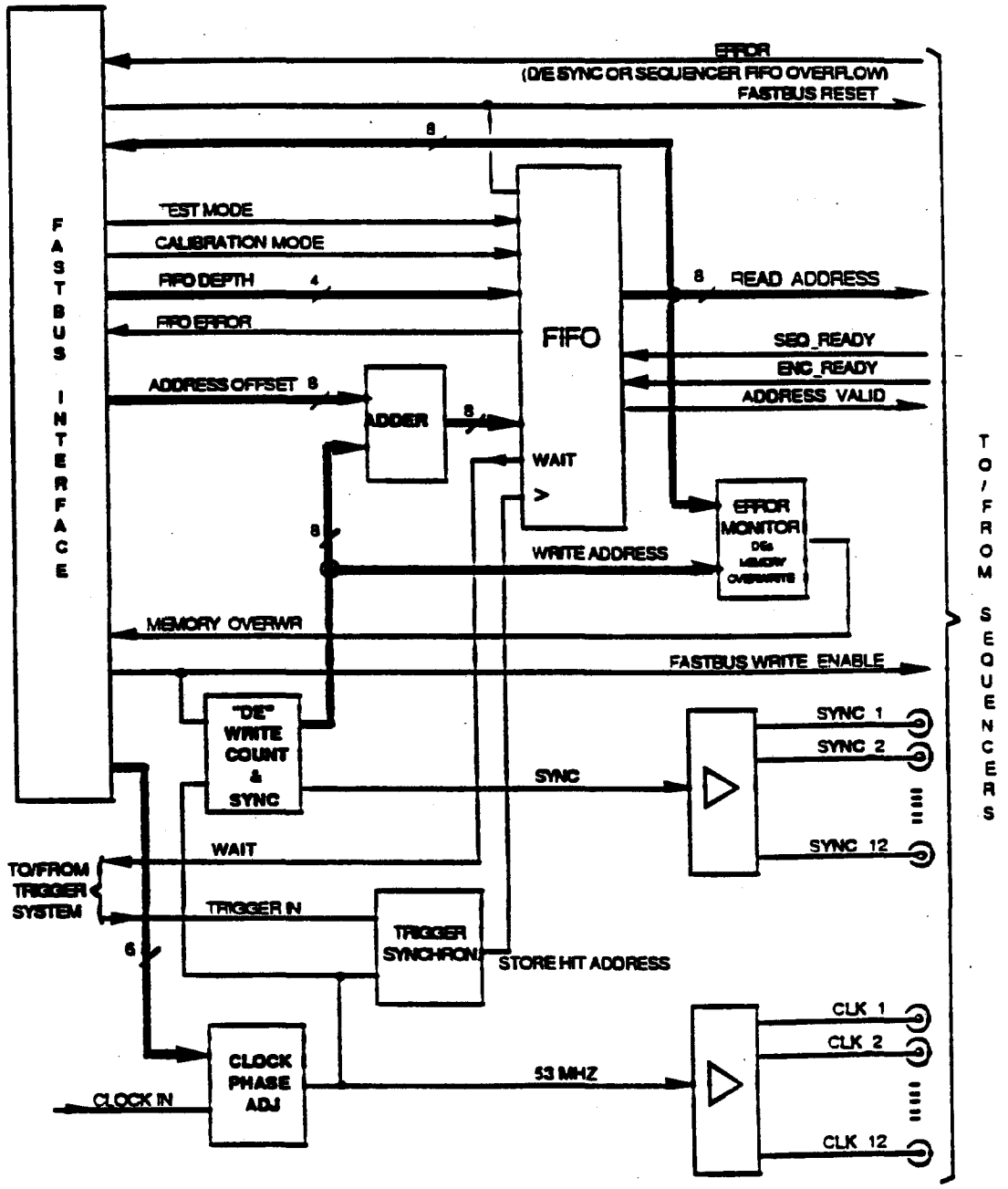


Fig. 2 - MTC internal block diagram

2.1. Clock Generation and System Synchronization

Figure 3 shows the Clock and the Sync generation block diagram.

The MTC expects a continuous NIM 53 MHz RF from the accelerator, and generates a nearly 50% duty cycle clock signal that is fanned out individually to each crate. In order to maintain constant phase relationship to the RF, a EG&G Model 140/N Zero Crossing Discriminator is used. The 50% duty cycle is achieved by reshaping the 53 MHz pulses with a 8 ns delay line. To account for slow drifts of the RF with respect to the actual beam, the MTC provides a 6-bit programmable delay line with .5 ns resolution for clock phase adjustment. This delay line is programmed through Fastbus.

The MTC is capable of delivering a fixed phase 53 MHz clock relative to the beam. This clock is the absolute timing signal for the system and can be used as a reference throughout the system. If individual modules need a different phase relationship, they have to adjust the phase locally.

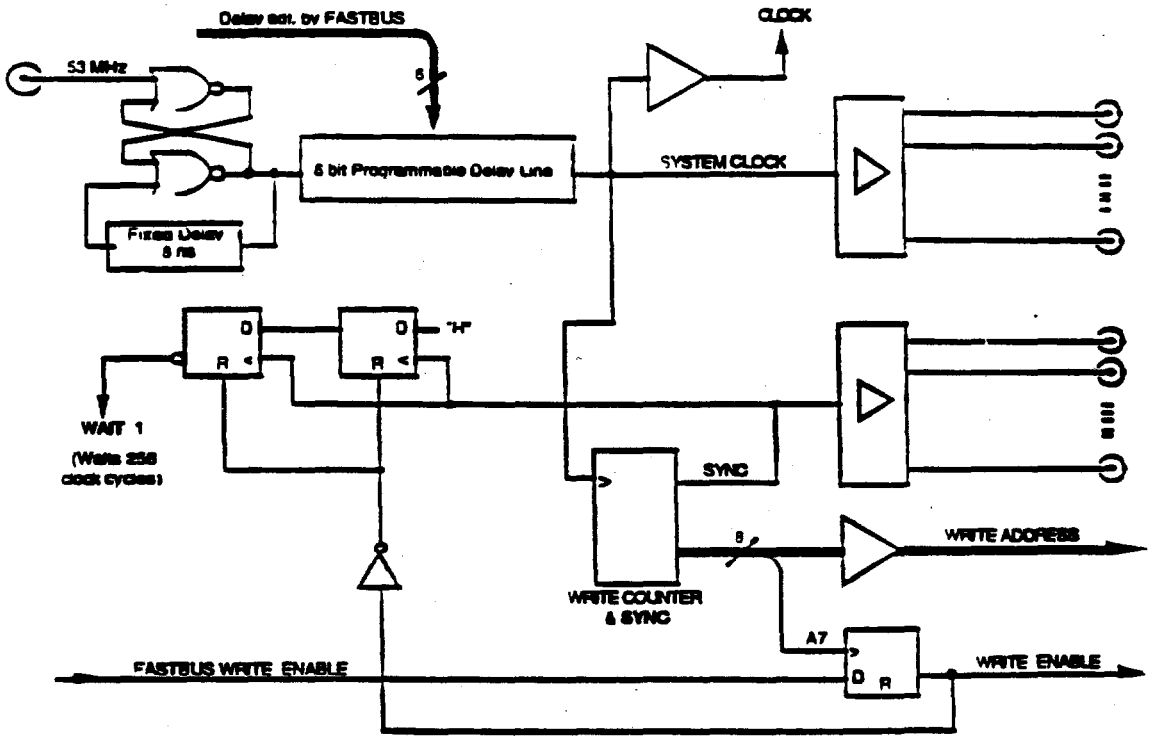
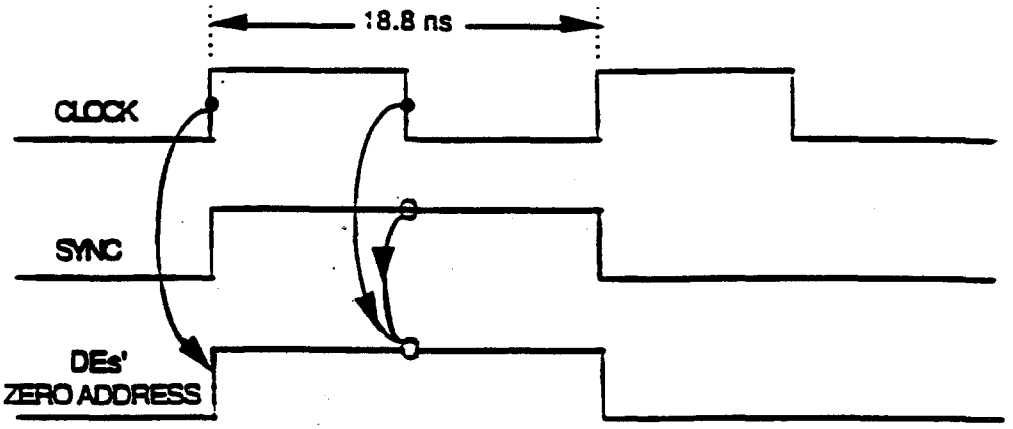
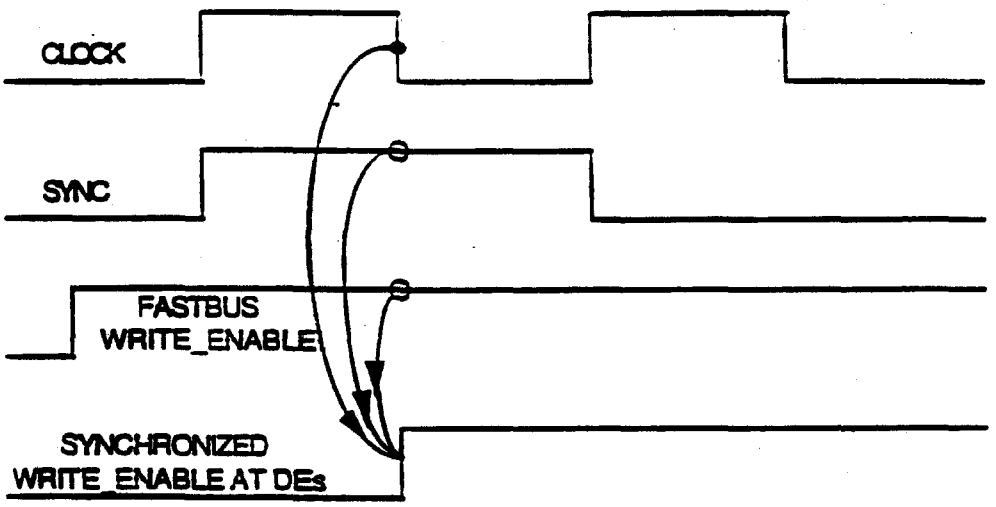


Fig. 3 - Clock Generation and System Synchronization

The MTC generates a SYNC signal with fixed phase relative to the clock on every 256 clock cycles. The MTC keeps a copy of the D/Es write counters for use as a reference to evaluate the read address (hit address). The MTC thus knows when the write counters in every D/E are expected to be at address zero. At this precise time the MTC broadcasts a 18.8 ns SYNC pulse to the Sequencers in the system. The Sequencer will then bus this signal to all the D/Es in the crate, with each D/E checking its synchronization to the SYNC pulse. If a loss of sync is detected, the faulty D/E will inform the Sequencer in the crate by asserting the SYNC_ERROR line (wired-or of all D/Es in a crate). Each faulty D/E will identify itself by latching the error and displaying it on the front panel. The SYNC signal has a close phase match to the system clock, although it is not used as a timing signal. The SYNC signal is sampled in the D/Es on the trailing edge of the system clock, as shown in Figure 4a (note that the SYNC signal allows some phase skew without affecting D/Es synchronization). The Sequencers, which have control over the clock phase in each crate, have to adjust the SYNC signal phase every time they change the clock phase.



(a) Synchronism checking in the D/Es



(b) WRITE_ENABLE Synchronization

Figure 4 - D/E and WRITE_ENABLE timing diagrams

The WRITE_ENABLE signal programmed through FASTBUS is intended to start the D/Es to acquire hit data. The WRITE_ENABLE is sent to the Sequencers/DEs 128 clock cycles before the SYNC pulse. The D/Es retime this signal using the CLOCK and SYNC signals for synchronizing the start of event acquisition (see Figure 4b). The D/Es are expected to remain synchronized to the MTC generated SYNC signal and flag loss of sync errors.

In system start up, the WAIT signal to the trigger system is deasserted only after the D/Es' memories have been writing data for 256 clock cycles; this allows de D/Es to initialize their memories with new data and assures that the data stored in the D/Es' memories is valid when the trigger pulse arrives.

A flow chart of the system initialization is shown in Figure 5.

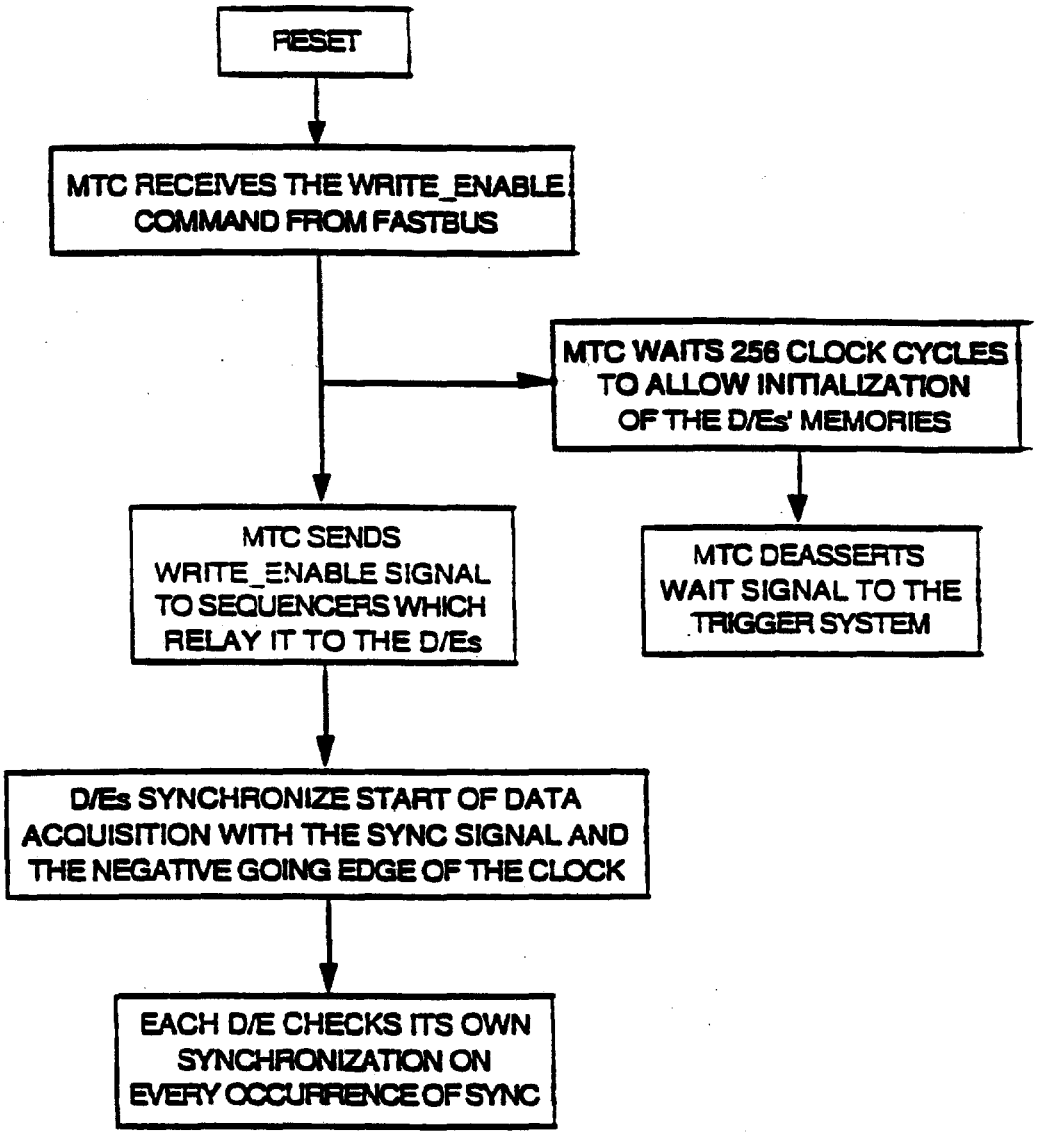


Figure 5 - System initialization

2.2. Read Address

Each trigger pulse received by the MTC generates a read address -- address where the hit data corresponding to the trigger is stored in de D/Es memories --, obtained by adding an offset to the reference D/E write counter of the MTC. This offset is software programmable and its value is determined by a calibration procedure. The offset accounts for the trigger decision time, which is expected to be close to 1 μs.The read addresses are sent asynchronously to the Sequencers (and D/Es) along with the address valid signal. If the D/Es are busy, The MTC stores the trigger addresses in a FIFO for later delivery. If a read address is close by some amount to the current write address in the D/Es, such that the memory could be overwritten, an error is issued. This amount is programmed through dip switches.

2.3. 1st Level Trigger Communication

The MTC accepts pulses from the 1st level trigger system and transmits the address of the data corresponding to that trigger to the Sequencers/DEs. The MTC is capable of pipelining trigger requests if the D/Es are busy encoding the previous trigger (see section 2.4 below). The pipeline depth, programmable through FASTBUS, depends on the trigger delay, and on the expected average number of hits in the detector, which in turn causes different encoding times in the D/Es. A wise selection of the number of stages in the pipeline prevents the D/Es from wrapping around their memories and consequently overwriting data: the MTC tests for this type of error as said in the previous section. When the number of events waiting to be serviced nears the maximum programmed number of stages in the pipeline, the MTC sends a WAIT signal to the trigger system. The MTC, however, doesn't block out new trigger requests, even though it has sent the WAIT to the trigger system. This feature is important in the case of trigger requests that are in the process of being delivered when the WAIT signal is asserted. The trigger request FIFO can store up to 8 trigger requests.

The MTC will also assert the WAIT signal to trigger system when the Sequencers' FIFOS used to store the encoded data from the D/Es become more than half full with encoded data from the D/Es. The Sequencers notify the MTC of this condition by deasserting the SEQ_READY line.

2.4. Trigger pipelining

When the D/Es are busy (not READY), the MTC has to store the incoming triggers in a FIFO, so that they don't get lost. Commercial FIFOs available today have limitations in speed and functionality. To meet the requirements of the SSD trigger pipelining, an ECL discrete FIFO is built in the MTC, with the following characteristics:

- 53 MHz input frequency
- depth (number of stages) programmable through FASTBUS
- generation of a WAIT signal if the current number of stages in the FIFO is equal or greater than the FASTBUS programmed depth
- generation of status such as *empty, almost full, and error*

The number of stages in the FIFO was limited to 8, a number that satisfies experiment requirements and is easy to implement. Figure 6 shows the FIFO connections to other MTC blocks.

The FIFO is implemented as 8 registers, 2 pointers (write and read pointers) and a status generation logic.

The write pointer is always pointing to the next free location in the FIFO. Upon receiving of a trigger pulse, the FIFO writes the hit address to the current free location and increments the pointer to the next one. The write operation is synchronized internally to the falling edge of the clock (the hit addresses are generated after the rising edge of the clock).

The read pointer is always pointing to the register where the data in the pipe is to be read. In order to allow minimum time for the error and status circuitries, the reads are synchronized internally to the rising edge of the clock. After retrieving a trigger address from the FIFO, the read pointer is incremented. The FIFO depth is simply the write pointer minus the read pointer.

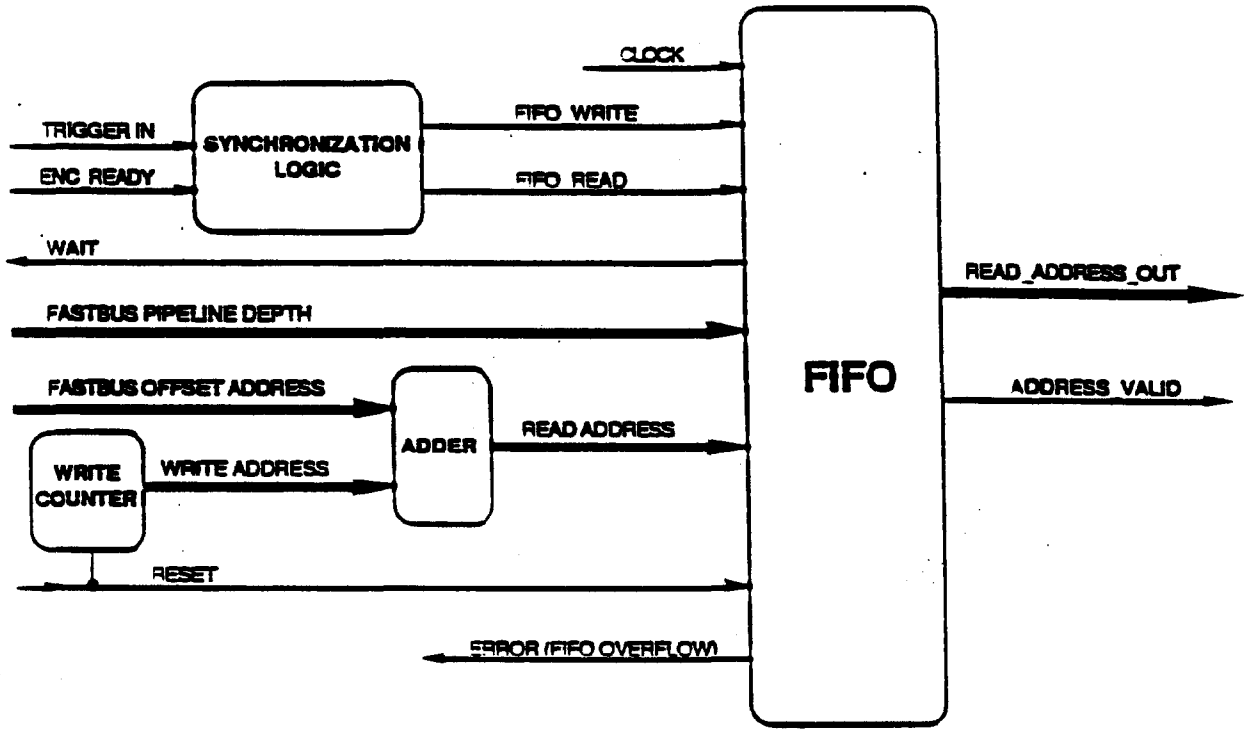


Fig.6 FIFO signals

The pipelining functional characteristics are:

- *fifo empty*: do nothing; wait for trigger.
- *fifo not empty*: if the D/Es are READY, retrieve a read address stored in the pipe; point to the next read address in the pipe.
- *fifo depth > programmed number of stages*: send WAIT to the trigger system. This is a very important feature that makes the FIFO more flexible, allowing the trigger pipelining to be tuned to the detector hit rate.
- *fifo depth > 6*: if the fifo depth is 7 or greater, send WAIT to the trigger system. This makes use of the *almost full* status condition.
- *fifo depth > 8*: error.

A more detailed block diagram for the FIFO is depicted in Figure 7.

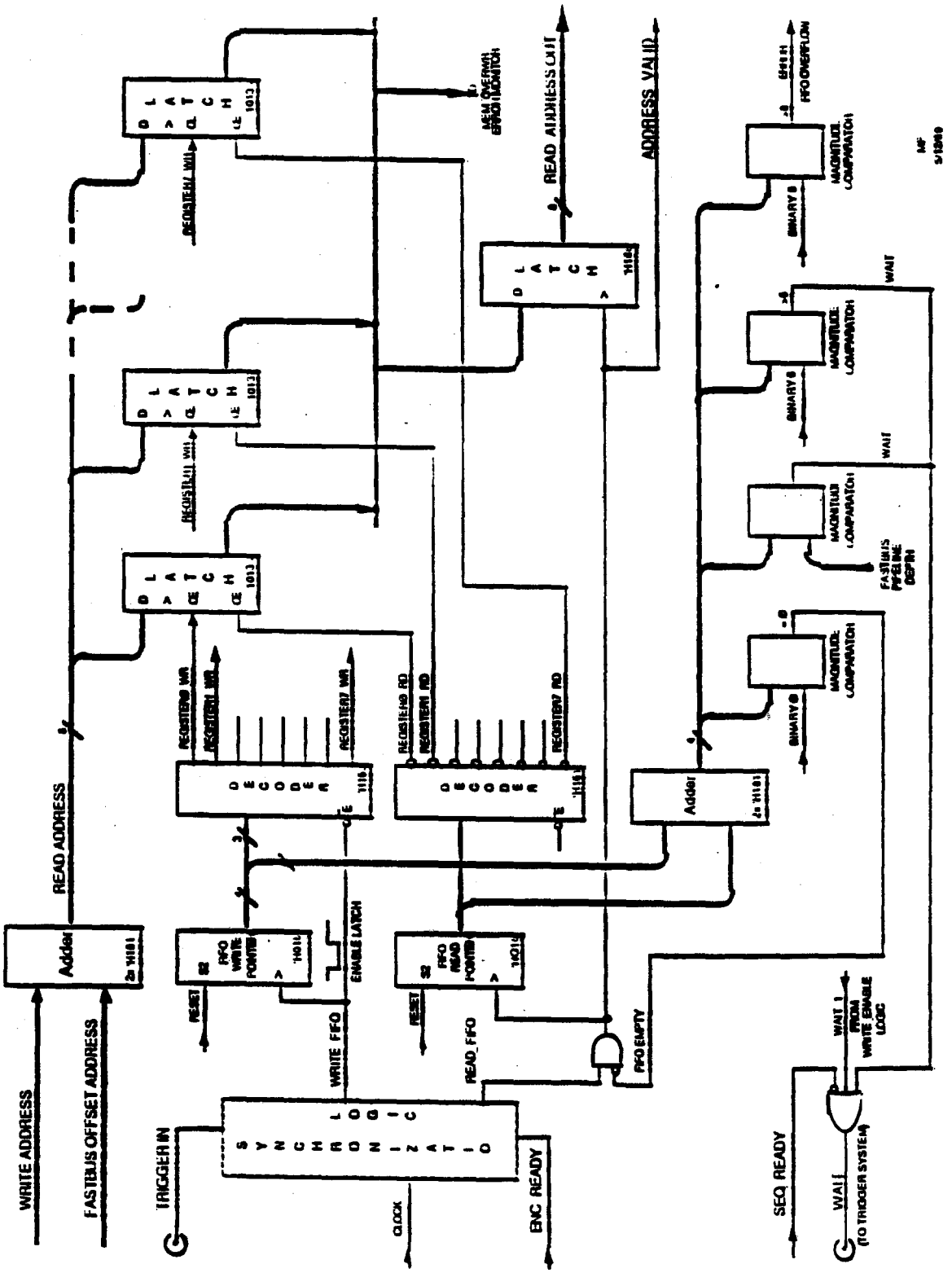


Fig. 7 - Fifo internal logic

2.5. Calibration Mode

This mode is used for evaluating the correct offset (number of clock cycles) necessary to accomplish for trigger decision time and other intrinsic delays.

First, the CALIB_MODE signal is asserted, causing the MTC to empty its FIFO by ignoring trigger requests. When a software generated START_CALIB command is issued, the MTC then waits for the first external trigger pulse to arrive and generates 5 consecutive read addresses (using an estimated offset), which are delivered on demand to the D/Es. Other trigger pulses are ignored, unless a new START_CALIB command is issued, causing the MTC to generate a new burst of trigger addresses synchronized to the arrive of the following trigger pulse input. In other words, each START_CALIB pulse will generate one burst of 5 consecutive trigger addresses, the first trigger address being synchronized to a trigger input.

Performing this calibration procedure with different clock phases (adjusted in the Sequencers) and different offsets, the system is able to determine the correct offset by looking into the data collected in the D/Es. This offset then becomes a constant to the system and will be used by the MTC in evaluating the read addresses that are broadcasted to all Sequencers/D/Es in the system.

2.6. Test (Debugging) Mode

The test mode feature is intended for checking the D/Es proper operation. It also tests the D/E-Sequencer communication, since all data from the D/Es pass through the Sequencers.

In TEST_MODE, the D/Es acquire data in the usual fashion, the data now being a known pattern generated by the Post-Amp/Comparator board. The TEST_MODE capability allows one to read the contents of the D/Es' memories and compare them to the pattern being written into the memories.

In this mode, the write counter in the MTC is shut down (SYNC pulses are still sent out), and the read address is controlled by the trigger offset setting alone. The actual read address that is broadcast is achieved by subtracting an offset from the current write counter contents (which is zero in this case). In this way, one has control over which addresses to read from the D/Es. A software program can scan over the entire D/E memory and check for correct data.

Trigger requests are disregarded in this mode, and triggers are generated by software (FB_TRIGGER). The generation of software triggers has to adhere to the same conventions used by the external trigger, i. e., has to wait if the WAIT signal is asserted, caused by either the FIFO wait or by the Sequencer not being ready.

The procedures to execute the test are:

. Assert FB_WRITE_EN. This will initiate the D/Es accepting hit data.

. Assert the TEST_MODE signal.

. Load an offset that will give the desired read address.

. Generate the FB_TRIGGER pulse by software.

3. ERRORS

The MTC informs the system of the occurrence of errors. The errors can be read through FASTBUS and include:

. DE_SEQ_ERROR - means that a D/E module has lost synchronization or that a Sequencer FIFO has overflown. To identify the exact cause of the error, one needs to read the Sequencers' error registers.

. MEM_OVWR_ERROR - indicates that a trigger address cannot be broadcast due to data being overwritten in the D/E memories. This monitoring is accomplished in the MTC.

. FIFO_OVFL_ERROR - indicates that the trigger pipelining FIFO in the MTC has overflown. This error may occur if the trigger system disregards the WAIT signal the MTC sends to it informing the pipe is almost full.

. TRIG_PHASE_ERROR - signals that the trigger input from the first level trigger system has drifted by an amount that causes it to fall outside a pre-established time window.

4. INPUT/OUTPUT

4.1. FASTBUS Interface

The MTC communicates to the external world through FASTBUS. Functions executed through FASTBUS include system resetting, system calibration, system debugging, clock phase adjustment, pipeline depth setting, read address offset programming and error reporting. Below are the functions provided in FASTBUS.

4.1.1. Reset

The system reset is accomplished in bit 30 of CSR0. Reset is asserted in system power up or in response to a fatal error.

4.1.2. Write_enable

Command signal to start data acquisition. The MTC transmits this signal to all Sequencers/DEs. It is set by CSR0(02) and reset by CSR0(18).

4.1.3. Calibration mode

The calibration mode is set by CSR0(06) and reset by CSR0(22). Associated with the calibration mode is the start_calib pulse in CSR0(07).

4.1.4. Test mode

Set by CSR0(08) and reset by CSR0(24).

4.1.5. Clock Phase

The clock phase adjustment requires a 6 bit register for programming a delay line with .5 ns resolution (31.5 ns total delay time). The delay line used is the ELMEC PDH 6500. The MTC receives a quantity that reflects the clock phase deviation and compensates it by reprogramming the delay line. CSR13 is used for this purpose.

4.1.6. Trigger pipeline depth

CSR11 is used to program the pipeline depth from 0 to 7.

4.1.7. Read address offset

The read address is obtained by adding a programmable offset to the reference write counter in the MTC. The offset 8 bit number is stored in CSR12.

4.1.8. Trigger Wait Status

The status of WAIT output on the front panel can be read from CSR11(4) which reads 1 whenever WAIT is asserted. This is intended for module diagnostic purposes.

4.1.9. FIFO NOT EMPTY

The NOT EMPTY/EMPTY status of the FIFO can be read from CSR11(5) which reads 1 when the FIFO is NOT Empty and 0 when the FIFO is EMPTY. This bit is is intended for module diagnostic purposes.

4.2. Front Panel Signals

Coaxial connectors:

. CLOCK INP - a NIM 53 MHz clock from the Zero Crossing Discriminator Module. The only concern here is to have a signal that has a fixed phase relationship to the RF. This signal may experience slow timing drifts over long periods of time.

. CLOCK OUT - a NIM 50% duty cycle clock output reference that has a constant phase to the beam. This signal is fanned out individually to each crate.

. SYNC OUT - NIM output pulse, synchronous to CLOCK, to test write counters synchronization at each zero count. This signal is delivered individually to each Sequencer.

. ERROR_OUT - NIM output intended for immediate signalization of an error condition.

. TRIGGER INP - NIM trigger pulse input from the 1st level trigger system.

. WAIT - NIM output signal informing the 1st level trigger that the system is busy and cannot accept new trigger requests.

Ribbon cable connector:

. RESET - differential ECL output signal, asynchronous to CLOCK, for system resetting and initialization.

. WRITE_ENABLE - output signal, differential ECL, asynchronous to CLOCK, for enabling the D/E modules to start data acquisition, i. e., accept hit data and increment write counters. This signal is bussed to all crates.

. ENC_READY - wire OR'd ECL input signal from the Sequencers informing the D/Es status. This signal is used in determining when to deliver another read address to the Sequencers/DEs.

. SEQ_READY - input from the Sequencers, wire OR'd ECL, it signals that the Sequencers have room in its FIFOS for encoded events. If the Sequencers are not READY, the MTC immediately sends a WAIT to the trigger system.

. ERROR - wire OR'd ECL input from the Sequencers informing that the FIFOS have been overfilled by the encoders or that a D/E has lost synchronization.

. READ_ADDRESS - 8 bit differential ECL output, asynchronous to CLOCK, bussed to all crates.

. ADDRESS_VALID - differential ECL output, asynchronous to CLOCK, strobes read addresses in the Sequencers. This signal is bussed to all crates.

5. APPENDIX -Fastbus interface memory map

The FASTBUS interface for the MTC has the module ID 01A2. The signals and addresses are listed below.

CSR0 Register		
bit	read	write
CSR0(00)	Error Flag	Set Error Flag
CSR0(16)	ID	Clear Error Flag
CSR0(02)	WRITE_ENABLE	WRITE_ENABLE Set
CSR0(18)	ID	WRITE_ENABLE Clear
CSR0(06)	Calibration Mode status	Calibration Mode set
CSR0(07)	ENC_READY	START_CALIB
CSR0(22)	ID	Calibration mode clear
CSR0(08)	Test Mode status	Test Mode set
CSR0(09)	CLOCK_MISSING*	not used
CSR0(24)	ID	Test Mode clear
CSR0(10)	DE_SEQ_ERR*	not used
CSR0(11)	MEM_OVWR_ERR*	not used
CSR0(12)	FIFO_OVFL_ERR*	not used
CSR0(13)	TRIG_PHASE_ERR*	not used
CSR0(30)	ID	Reset
CSR0(16:31)	MODULE ID (01A2)	

CSR10(0:7)	FIFO_RA(0:7)	don't care
CSR10(08)	not used	FB_TRIGGER
CSR10(09)	not used	FB_ENC_READY

CSR11	Trigger Pipeline depth	
CSR11(0:2)	PRDPTH(0:2)	same
CSR11(4)	TRIG_WAIT	don't care
CSR11(5)	FIFO _NOT_EMPTY	dont't care

CSR12	Trigger Address offset TOFFSET(0:7)	same

CSR13	Clock Phase adjustment CKPHASE(0:5)	same

Fermi National Accelerator Laboratory

MASTER TIMING CONTROLLER

HARDWARE DESCRIPTION

M.Fachin, C. Rotolo, C. Needles, P. Spentzouris

October 19, 1990
Rev. January 23, 1991

Table of Contents

1. INTRODUCTION

The Master Timing Controller (MTC) module is part of the Silicon Strip Detector System, intended for use in experiments E771 and E789. In addition to generating the system clock coherent to the beam, the MTC provides the mechanism to maintain system synchronization, and acts as an interface between the readout system and the 1st level trigger system, generating hit addresses in response to trigger requests. The MTC provides a pipeline for triggers and is responsible for system reset (initialization). Figure 1 shows the MTC connections to the Sequencer modules and to the trigger system.

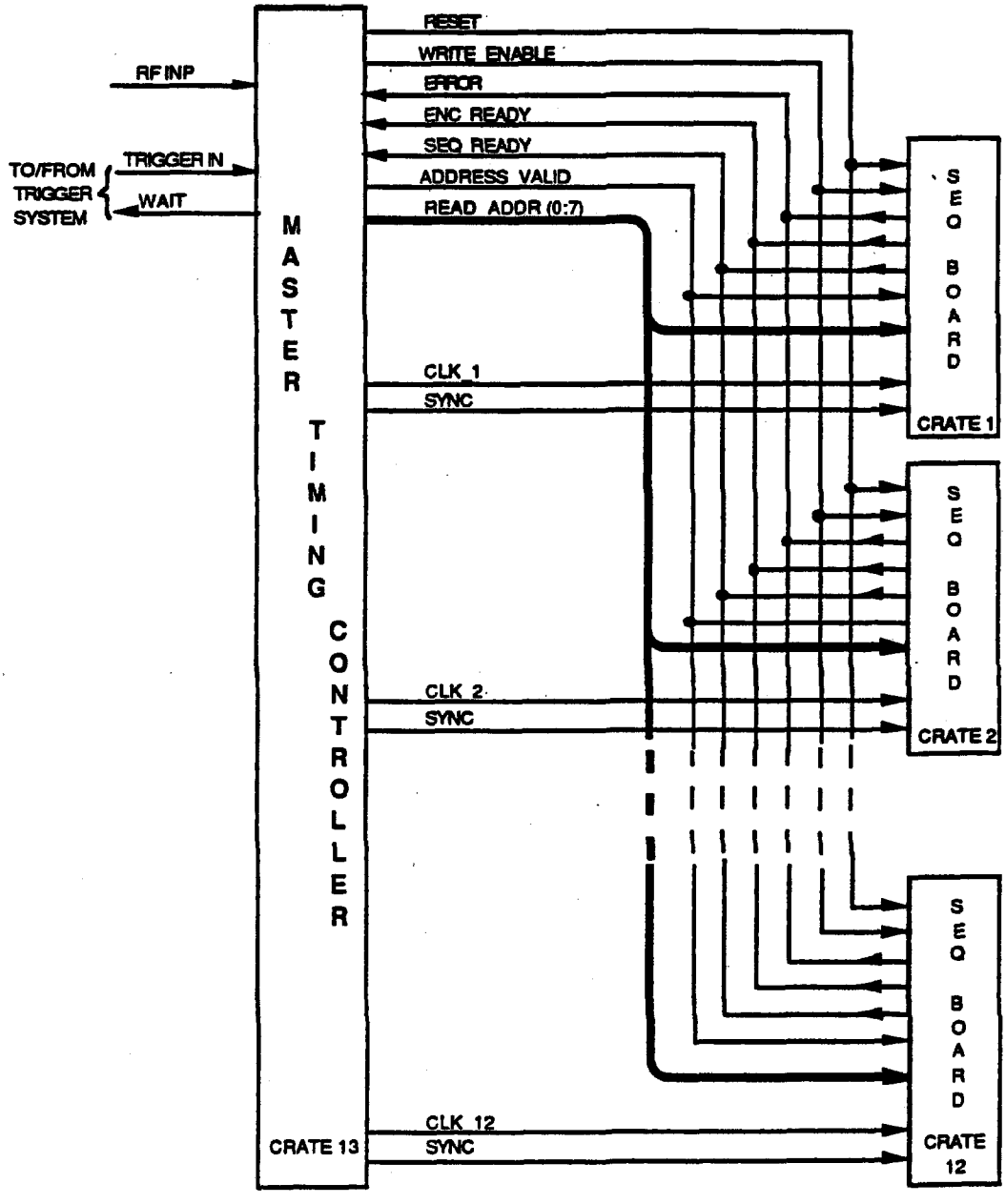


Fig. 1 - MTC connections to the Sequencers and 1st level trigger system.

2. GENERAL DESCRIPTION

The Master Timing Controller generates the system clock, controls system synchronization, and generates the hit addresses upon the receipt of Level 1 triggers from the experiments Trigger system. A block diagram of the MTC is shown in Figure (2). The MTC receives the 53 MHz Tevatron RF and establishes a near 50% duty cycle clock whose phase is adjustable relative to the incoming RF. This CLOCK along with a SYNC pulse (occurring every 256 clock cycles) are distributed to each Sequencer and eventually to all Delay/Encoders (D/Es). D/Es use this clock and sync to determine write addresses for incoming data. Being synchronized, the MTC knows the current D/E write address and generates a read or "hit" address when a trigger is received. The hit address generated is offset from the write address based upon a calibration of the Trigger decision time.

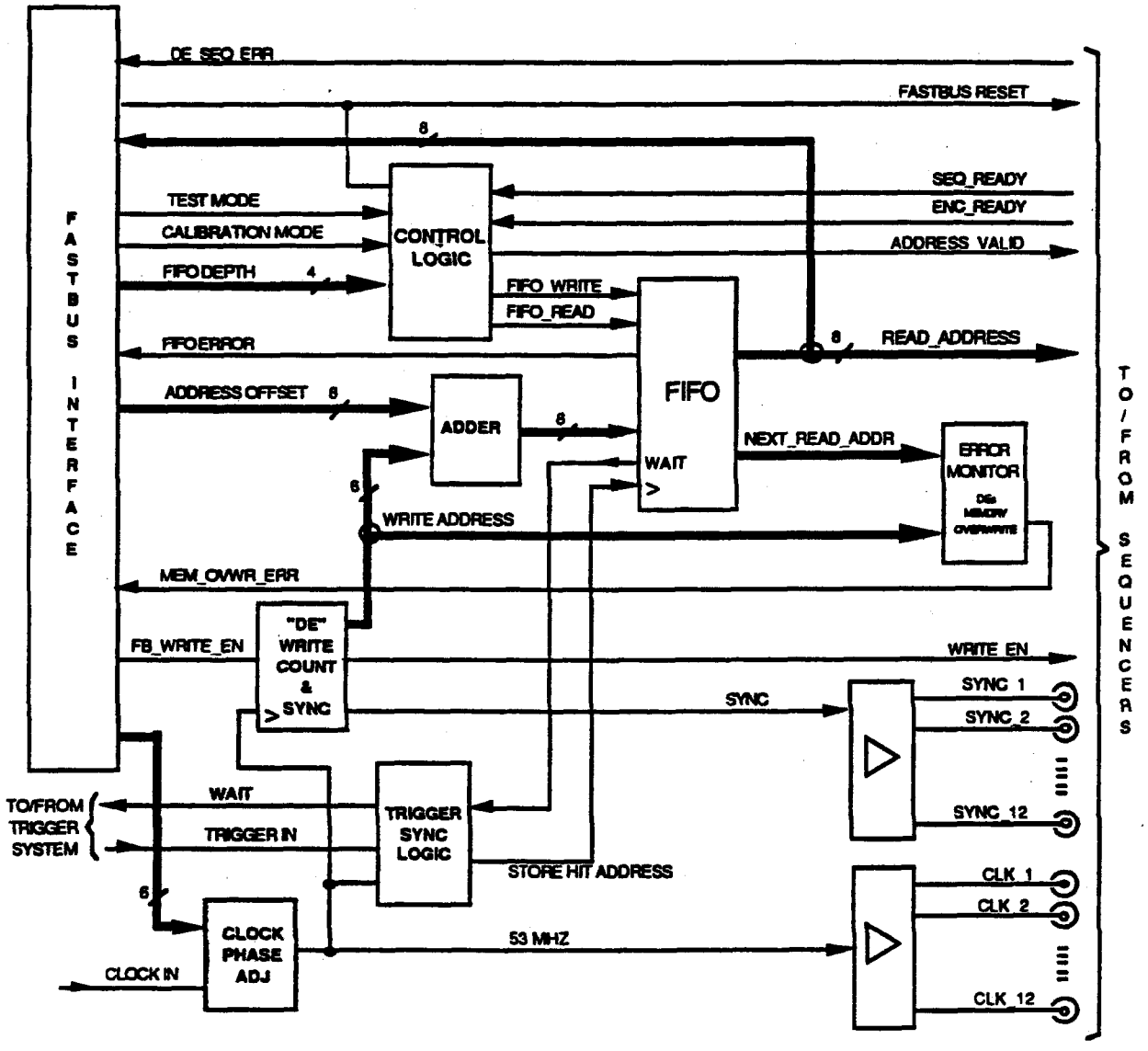


Fig. 2 - MTC internal block diagram

In the case of D/Es being busy, hit addresses are placed into a high speed FIFO queueing up to seven trigger requests which can occur as close together as successive RF buckets. The read or "hit" address output from the FIFO is broadcast to all D/E modules at a rate determined by the ENC_READY signal summed from all D/E modules.

The ability to pipeline triggers makes the system truly deadtimeless at trigger rates up to the readout bandwidth. With knowledge of the read address and the D/E's current write address, the MTC detects fatal D/E memory overwrite errors (256 clock cycles, or 4.8 μs, is how much time it takes to overwrite the memory). However, to prevent such a condition, the system is throttled by sending a WAIT signal to the Trigger system — the MTC trigger pipeline depth can be set by Fastbus and will generate a trigger WAIT at this depth.

The MTC provides trigger synchronization monitoring and incorporates features for system calibration and debugging.

3. CLOCK GENERATION AND SYSTEM SYNCHRONIZATION

Figure 3 shows the Clock and the Sync generation block diagram.

The MTC expects a continuous NIM 53 MHz RF from the accelerator, and generates a fixed duty cycle near 50% that is fanned out individually to each crate. In order to maintain constant phase relationship to the RF, it is suggested that the sinusoidal RF signal be processed by EG&G Model 140/N Zero Crossing Discriminator, producing NIM level pulses, prior to being fed to the MTC, which in turn reshapes the 53 MHz pulses with a 8 ns delay line to achieve the desired duty cycle. To account for slow drifts of the RF with respect to the actual beam, as well as equipment changes, the MTC provides a 6-bit programmable delay line with 0.5 ns resolution for clock phase adjustment. This delay line is programmed through FASTBUS.

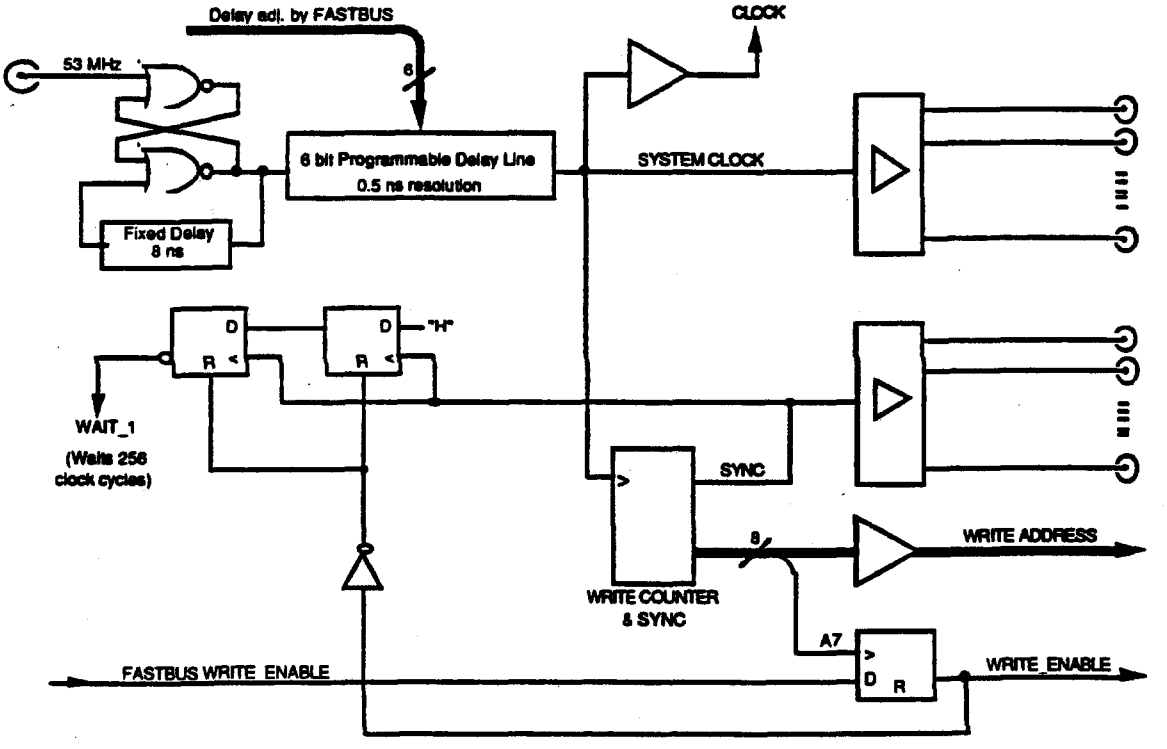
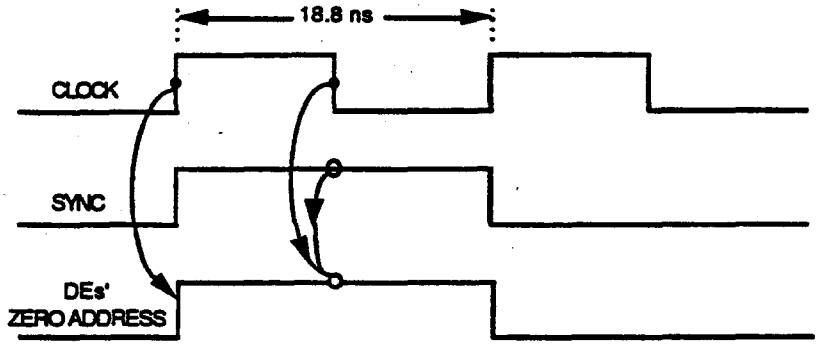


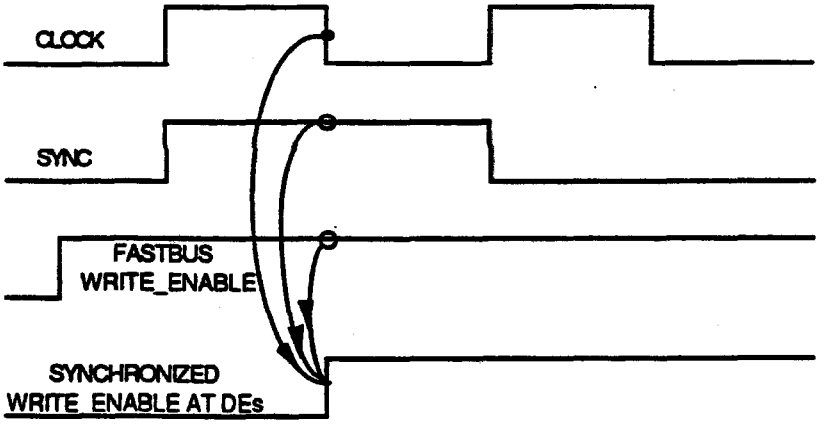
Fig. 3 - Clock Generation and System Synchronization

The MTC is thus capable of delivering a fixed phase 53 MHz clock relative to the beam. This clock is the absolute timing signal for the system and can be used as a reference throughout the system. If individual modules of the Data Acquisition System need a different phase relationship, they have to adjust the phase locally.

The MTC generates a SYNC signal with fixed phase relative to the clock on every 256 clock cycles. The MTC keeps a copy of the D/Es write counters for use as a reference to evaluate the read address (hit address). The MTC thus knows when the write counters in every D/E are expected to be at address zero. At this precise time the MTC broadcasts a 18.8 ns SYNC pulse to all Sequencers in the system. The Sequencer will then bus this signal to all the D/Es in the crate, with each D/E checking its synchronization to the SYNC pulse. If a loss of sync is detected, the faulty D/E will inform the Sequencer in the crate by asserting the SYNC_ERROR line (wired-or of all D/Es in a crate). Each faulty D/E will identify itself by latching the error and displaying it on the front panel. Although the SYNC signal has a close phase match to the system clock, it is not used as a timing signal. The SYNC signal is sampled in the D/Es on the trailing edge of the system clock, as shown in Figure 4a, and is allowed some phase skew without affecting D/Es synchronization. The Sequencers, which have control over the clock phase in each crate, have to adjust the SYNC signal phase if they change the clock phase.



(a) Synchronism checking in the D/Es



(b) WRITE_EN Synchronization

Fig. 4 - D/E and WRITE_ENABLE timing diagrams

The SYNC signal has to be calibrated internally in the MTC to phase match the clock signal. A tapped delay line is provided for this purpose, with dip switch SW3 being used to select the tap which gives the right phase. This calibration is done only once and remains fixed thereafter.

The WRITE_EN signal programmed through FASTBUS is intended to start the D/Es to acquire data, as well as allowing the MTC to accept trigger requests. The WRITE_EN is sent to the Sequencers/D/E 128 clock cycles before the SYNC pulse, so that all D/Es can see it before the next SYNC pulse.

On system start up, the WAIT signal to the trigger system is deasserted only after WRITE_EN is asserted, and 256 clock cycles after the D/Es have started taking data.

A flow chart of the system initialization is shown in Figure 5.

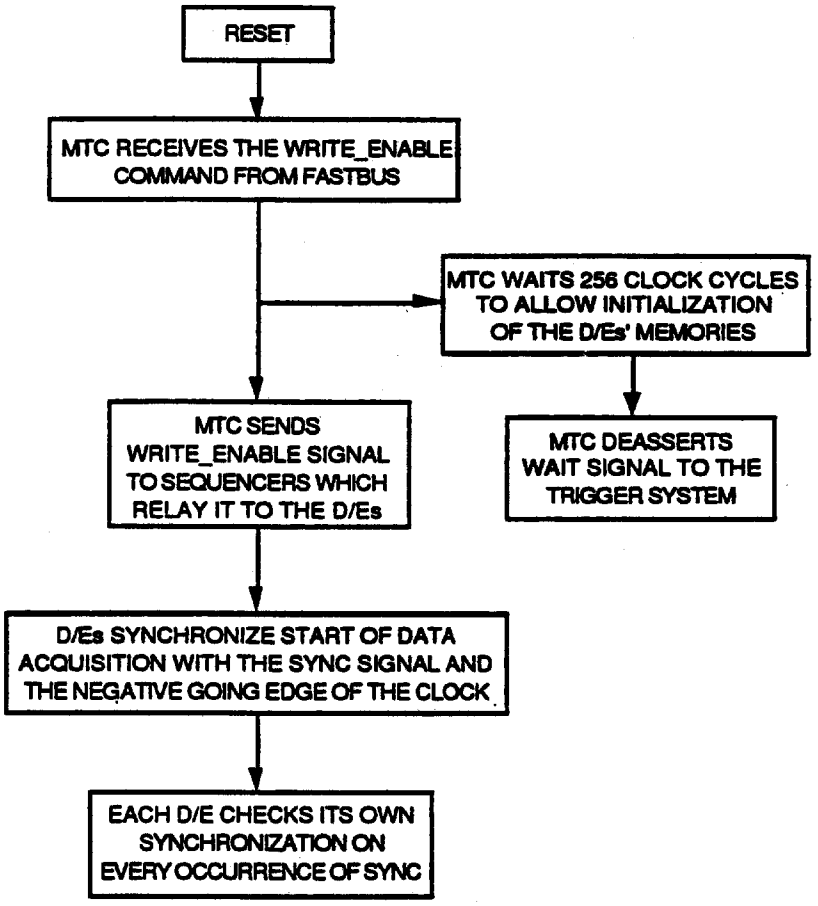


Fig. 5 - System initialization

4. READ ADDRESS

Each trigger pulse received by the MTC generates a READ_ADDRESS — address where the hit data corresponding to the trigger is stored in the D/Es memories —, obtained by subtracting an offset to the reference D/E write counter of the MTC. This offset is software programmable and its value is determined by a calibration procedure. The offset accounts for the trigger decision time, which is expected to be close to 1 μs (for example, for a trigger decision

time of 1 μs, the offset would be 1000 ns/18.8 ns ≈ 53 clock cycles). The READ_ADDRESSes are sent asynchronously to the Sequencers (and D/Es) along with the ADDRESS_VALID signal. If the D/Es are busy, the MTC stores the trigger addresses in a FIFO for later delivery. An error is issued if the NEXT_READ_ADDR to be output from the FIFO is too close (dip switch programmable in SW2) to the current write address in the D/Es, such that a MEM_OVWR error could occur. An opened switch on SW2 means a logic one, and the quantity programmed in SW2 should be equivalent to the time (number of clock cycles) needed for the READ_ADDRESS to reach the D/Es and be processed before that memory location is overwritten with new data. The memory overwrite error monitor is inhibited when the FIFO is empty, a situation in which the hardware would flag a non-existing error.

5. TRIGGERS

Triggers can be external or internal. External triggers come from the 1st level trigger system and signal an event (external triggers are inhibited in Test mode and under error conditions in other modes). Internal triggers are software generated triggers and are intended for test purposes only. There are no hardware restrictions as to when an internal trigger is allowed. Triggers produce READ_ADDRESSes that are loaded into the trigger FIFO.

5.1. Trigger Phase

The incoming trigger pulse must hold a certain phase relationship to the system clock for the MTC to work properly. Specifically, a trigger must not arrive close to the rising edge of the clock, which would cause the trigger synchronization logic to malfunction, by not honoring the setup time of the logic. If the trigger phase is not set right, the MTC will flag an error named TRIG_PHASE_ERR, which will halt the system by discarding new trigger requests and by not sending out READ_ADDRESSes.The MTC produces the signals TRG WIN (trigger window) and TRG MON (trigger monitor) to help adjust the trigger timing. By observing these signals with a scope, one should externally delay the trigger pulse input such that the TRIGGER MONITOR signal lies inside the range presented by the TRIGGER WINDOW signal. It is recommended that the TRIGGER MONITOR signal be positioned in the center of this time window, to allow for eventual timing drifts.

An alternative way of setting the trigger timing is to monitor the TRIG_PHASE_ERR LED while making the delay adjustments. By noting the range for which no errors are flagged, one can set the trigger delay to be centered to the window.

The width of the window described above can be trimmed to allow a narrower or looser phase error check for the incoming triggers. A rotary switch (SW5) is provided for setting the window width.

In test mode, triggers are generated by software and have a random phase. The trigger phase errors are disregarded in this case, and do not halt the system.

5.2. Trigger Interface

The MTC accepts pulses from the 1st level trigger system and broadcasts the addresses of the data corresponding to those triggers (READ_ADDRESSes) to the Sequencers/D/E. The MTC is capable of pipelining trigger requests if the D/Es are busy encoding the previous trigger. The pipeline depth, programmable through FASTBUS, depends on the trigger delay, and on the expected average number of hits in the detector, which in turn causes different encoding times in the D/Es. A wise selection of the number of stages in the pipeline prevents the D/Es from wrapping around their memories and consequently overwriting data; the MTC tests for this type of error as said in the previous section. When the number of events waiting to be serviced exceeds the maximum programmed number of stages in the pipeline, the MTC sends a WAIT signal to the trigger system. The MTC, however, doesn't block out new trigger requests, even though it has sent the WAIT to the trigger system. This feature is important in the case of trigger

requests that are in the process of being delivered when the WAIT signal is asserted. The trigger request FIFO can store up to 7 trigger requests.

For test purposes, triggers can be generated by software (FB_TRIGGER) and are OR'ed to external triggers.

5.3. Trigger WAIT

The way the MTC throttles the trigger system is by sending a WAIT signal. This signal is asserted under four conditions: whenever the system is in stand-by, is running in Test mode, the FIFO depth exceeds the programmed FIFO depth, or the Sequencers are not ready (the Sequencers' FIFOs used to store the encoded data from the D/Es become more than half full with encoded data from the D/Es).

5.4. Trigger FIFO

When the D/Es are busy encoding the previous trigger, the MTC stores the READ_ADDRESSes corresponding to the incoming triggers in a FIFO. Commercial FIFOs available today have limitations in speed and functionality. To meet the requirements of the SSD trigger pipelining, a discrete ECL FIFO is built in the MTC, with the following characteristics:

- 53 MHz input frequency
- depth (number of stages) programmable through FASTBUS
- generation of a WAIT signal if the current number of stages in the FIFO is greater than the FASTBUS programmed depth
- generation of status such as *empty, full, and error*
- generation of the NEXT_READ_ADDRESS

The number of stages in the FIFO was limited to 7. Figure 6 shows the FIFO connections to other MTC blocks.

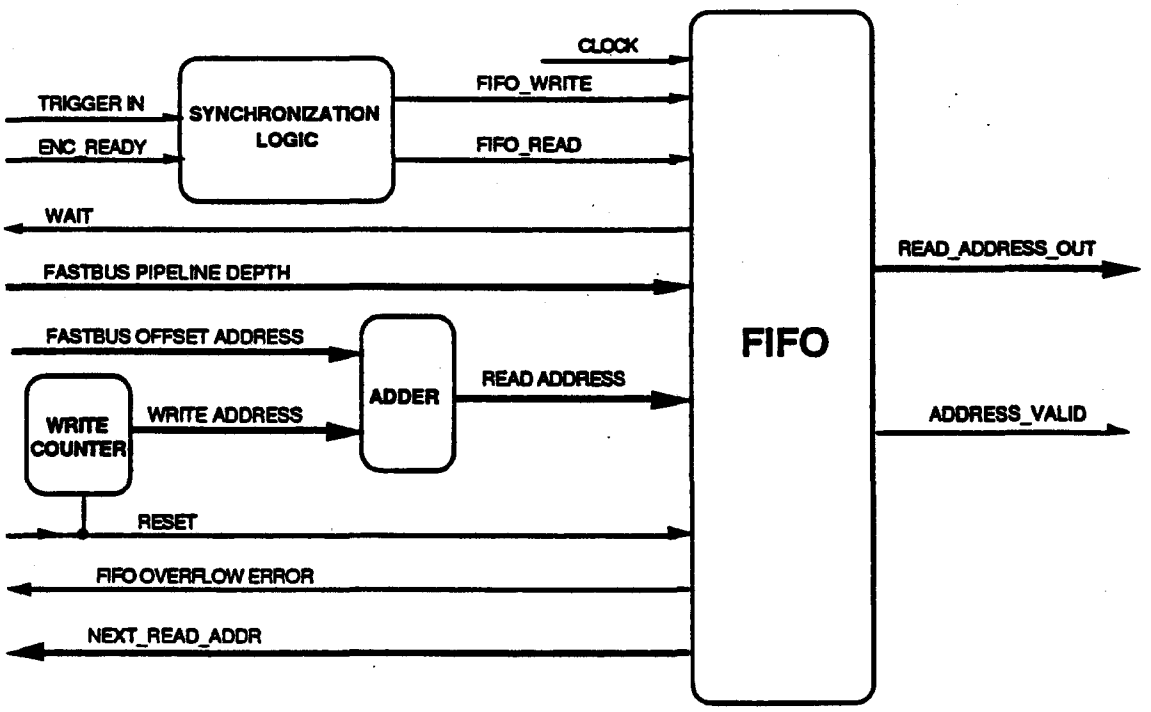


Fig. 6 - FIFO signals

The pipelining functional characteristics are:

- *fifo empty*: do nothing; wait for trigger.
- *fifo not empty*: if the D/Es are READY, retrieve a read address stored in the pipe; point to the next read address in the pipe.
- *fifo depth > programmed number of stages*: send WAIT to the trigger system. This is a very important feature that makes the FIFO more flexible, allowing the trigger pipelining to be tuned to the detector hit rate.
- *fifo depth > 7*: error.

A more detailed block diagram for the FIFO is depicted in Figure 7.

The FIFO is implemented as 8 registers, 2 pointers (write and read pointers) and a status generation logic.

The write pointer is always pointing to the next free location in the FIFO. Upon receiving a trigger pulse, the FIFO writes the hit address to the current free location and increments the pointer to the next one. The write operation is synchronized internally to the falling edge of the clock; the READ_ADDRESSes are generated after the rising edge of the clock.

The read pointer is always pointing to the register where the data in the pipe is to be read. The contents of this register is available and is called NEXT_READ_ADDR, used in the MEM_OVWR error monitor. In response to a FIFO_READ, this data is latched and becomes the READ_ADDRESS, which is sent out to the Sequencers/D/Es. The read pointer is then incremented, pointing to the NEXT_READ_ADDR. In order to allow minimum time for the error and status circuits, the FIFO_READ is synchronized internally to the rising edge of the clock. This provides one half clock cycle for the error checking circuit, since the writes occur on the falling edge of the clock. The difference between the write pointer and the read pointer is the FIFO depth.

6. MODES OF OPERATION

The MTC has 3 modes of operation: Run (Acquisition) mode, Test mode, and Calibration mode.

6.1. Run Mode

The Run mode is the normal mode for data acquisition. The MTC basically broadcasts READ_ADDRESSes in response to trigger requests, and is capable of pipelining triggers (read addresses) when the system is busy. The MTC checks for system integrity, halting the system on the occurrence of an error. A FB_WRITE_EN causes the MTC to enter Run mode.

6.2. Test (Debugging) Mode

The Test mode feature serves many purposes, the most important one being the system test. In TEST_MODE, the D/Es acquire data in the usual fashion, the data now being a known pattern generated by the Post-Amp/Comparator board. The TEST_MODE capability allows one to read the contents of the D/Es' memories and compare them to the known pattern being written into the memories.

In this mode, the write counter in the MTC is shut down (SYNC pulses are generated by an alternate counter used solely for this purpose), and the read address is controlled by the trigger offset (TOFFSET) setting alone. The actual read address that is broadcast is achieved by subtracting an offset from the current write counter contents (which is zero in this case).

"Test" READ_ADDRESS = 256 - TOFFSET

In this way, one has control over which addresses to read from the D/Es. A software program can scan over the entire D/E memory and check for correct data.

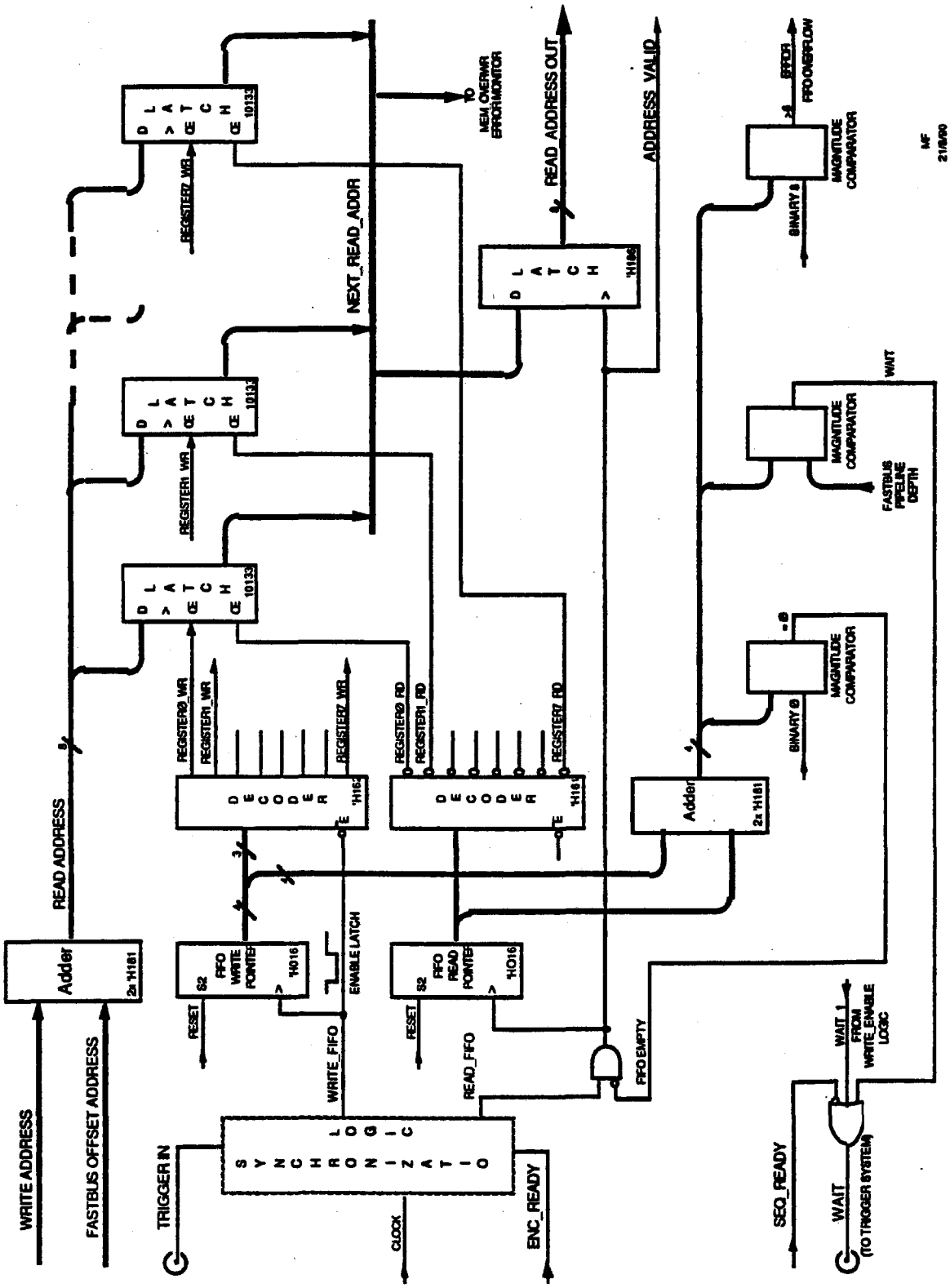


Fig. 7 - Fifo internal logic

External trigger requests are disregarded in this mode, and triggers are generated by software (FB_TRIGGER). The software generated triggers are not phased to the clock, and their role is to load test READ_ADDRESS into the trigger FIFO. A WAIT condition can be read back through Fastbus, caused by either a FIFO_WAIT or by the Sequencers not being ready, and has to be observed by the software routine that generates triggers.

The procedures to execute the test are:

. Assert the TEST_MODE signal.

. Assert FB_WRITE_EN. This will initiate the D/Es accepting hit data.

. Load an offset that will give the desired READ_ADDRES:

$$READ\_ADDRESS = 256 - TOFFSET$$

. Generate the FB_TRIGGER pulse by software.

. Change offset and wait for ENC_READY to issue another FB_TRIGGER

The Test mode can be also used to test the MTC alone. A great amount of the MTC's hardware can be checked under Test mode. The generation of a FB_ENC_READY closes the loop on the board, by allowing triggers stored in the FIFO to be read back through Fastbus. This checks the circuitry that evaluates the read address, the FIFO itself, and the Fastbus interface. The routine can be extended to further check the FIFO depth (generates Fastbus readable WAIT signals if the number of stages in the FIFO exceeds the programmed depth), the memory overwrite error monitor, and the hardware used in the calibration feature of the MTC.

6.3. Calibration Mode

This mode is used for evaluating the correct TOFFSET (number of clock cycles) necessary to accomplish for trigger decision time and other intrinsic delays. When set to Calibration mode, the MTC waits for the first external trigger pulse to arrive and generates N (switch setable) consecutive read addresses, using an estimated offset. The number of triggers produced is equal do the switch setting plus 1.
Fig. 8 shows a diagram of this switch, which is set to the number 4 in the figure (5 trigger pulses produced)

These READ_ADDRESSes are stored in the MTC's trigger FIFO, and are delivered on demand to the D/Es. Additional trigger pulses cause the MTC to generate a new burst of N trigger addresses synchronized to the arrival of the trigger pulse input. In other words, each trigger pulse will generate one burst of N consecutive trigger addresses. However, if a new trigger arrives before the N calibration triggers corresponding to the previous trigger have been stored in FIFO, the FIFO_OVFL_ERR is generated.

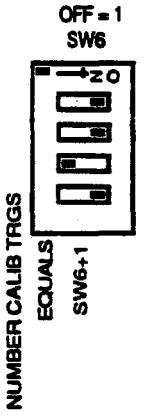


Fig. 8 - Diagram for the number of calibration triggers switch

The procedure to perform the calibration cycle is:

. RESET the module to clear the error latches and reset the FIFO.

. Set trigger pipeline depth (FIFO_DEPTH) to 1, so that the FIFO_WAIT will inhibit external triggers if the first burst of N read addresses is not serviced yet.

. Assert CALIB_MODE

. Set WRITE_EN

If any error occurs, external triggers are blocked out and ADDRESS_VALIDs are not produced.

Performing this calibration procedure with different clock phases (adjusted in the Sequencers) and different offsets, the system is able to determine the correct offset by looking into the data collected in the D/Es. This offset then becomes a constant to the system and will be used by the MTC to determine the READ_ADDRESSes that are broadcasted to all Sequencers/ D/Es in the system.

The Calibration mode internal hardware of the MTC can be checked by operating the MTC with both the Calibration mode and Test mode asserted. This allows one to have control of the READ_ADDRESS to be stored in the trigger FIFO (see Test mode) when a a calibration cycle is initiated by a software generated trigger. The expected contents of the FIFO are N (switch setable) consecutive numbers, the first one being equal to the loaded TOFFSET subtracted from 256.

7. ERROR MONITORS

The MTC monitors errors in the system. Any error is fatal, and the mechanism by which the MTC informs the control system of an error in the DAS or itself is to halt the system, i.e., external triggers are disregarded and ADDRESS_VALIDs are not sent out, except when the MTC is in Test mode. In addition, the MTC provides a NIM output indicating the presence of an error, and front panel LEDs for visualization of the errors. All errors are latched, and can be cleared by software or by pressing the front panel switch CLEAR ERRORS (provided the error conditions has been corrected).

The errors can be read through FASTBUS and include:

DE_SEQ_ERROR : means that a D/E module has lost synchronization or that a Sequencer FIFO has overflowed. To identify the exact cause of the error, one needs to read the Sequencers' error registers.

MEM_OVWR_ERROR : indicates that a trigger address cannot be broadcast due to data being overwritten in the D/E memories. The MTC compares the D/E' write address against the NEXT_READ_ADDR; if they are too close to one another, as programmed in dip switch SW2 (memory overwrite margin), the MEM_OVWR_ERR is flagged.

FIFO_OVFL_ERR : indicates that the trigger pipelining FIFO in the MTC has overflowed. This error may occur if the trigger system disregards the WAIT signal the MTC sends to it informing the pipe is almost full. Normally, the WAIT signal is sent out before the FIFO becomes full, whenever the programmed depth is exceeded.

TRIG_PHASE_ERR : signals that the trigger input from the first level trigger system has drifted by an amount that causes it to fall outside a pre-established time window.

CLOCK_MISSING : detects that at least one clock cycle was not received from the accelerator.

ERROR_SUM : it's the OR logic of all errors.

8. FRONT PANEL I/O AND DISPLAYS

Figure 8 shows a section of the MTC's front panel.

8.1. Front panel LEDs

MOD SEL : it is a yellow LED that flashes when module address is selected by Fastbus.

FIFO DEPTH : Is composed of 8 green LEDs numbered 0-7. They monitor the difference between the read and write pointers of the FIFO. If at 0, the read and write pointers are

pointing to the same register in the FIFO, which means the FIFO is *empty*. The maximum depth is 7, and an additional write to the FIFO will cause it to overflow, flagging an error.

- Errors -

MEM OVWR : indicates a MEM_OVWR_ERR in the D/Es.

FIFO OVF : indicates that the FIFO has overflowed.

TRG PHZ : monitors the external trigger phase to be within a pre-established time window.

DE/SEQ : signals that at least one D/Es is out of sync or that some Sequencer FIFO has overflowed.

CLK MIS : indicates the 53 MHz system clock is missing or was not present for at least one clock cycle.

- Status -

WRITE EN : status LED indicating the MTC is ready to accept triggers. The WRITE_EN is required also when the MTC is being operated in Test or Calibration modes.

CALIB : informs the MTC is in Calibration mode..

TEST : Test mode indication.

TRG IN : alongside TRG IN connector, shows when the MTC is receiving external triggers.

TRG WAIT : alongside the WAIT output, shows WAIT signals being sent to the trigger system. A persistent WAIT condition will leave the LED constantly lighted.

- At the front panel bottom -

See Fig 10.

+5 : monitors the 5 volts power in the MTC board.

-5.2 : monitors the -5.2 volts power for the ECL logic.

-2 : monitors the -2 volts power supply for the ECL terminations.

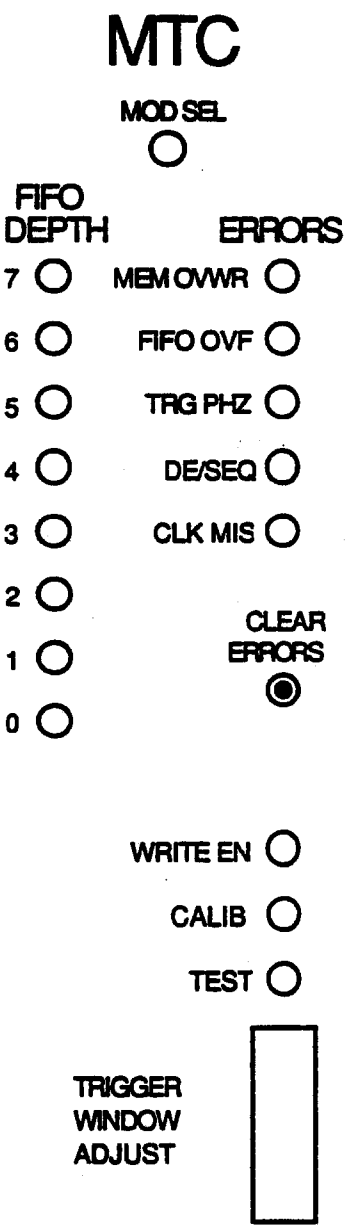


Fig 8 - MTC's front panel partial view

8.2. Coaxial connectors:

Fig 9 depicts the MTC's front panel coaxial connectors.

CLOCK IN : a NIM 53 MHz clock input which is derived from the accelerator RF via the CATV system. The only concern here is to have a signal that has a fixed phase relationship to the RF. This signal may experience slow timing drifts over long periods of time, which can be compensated by reprogramming (Fast-bus operation) the delay line internal to the MTC.

CLOCK OUT : a NIM 50% duty cycle clock output reference that has a constant phase to the beam. This signal is fanned out individually to each of the 12 crates. The 13th output is for monitoring.

SYNC OUT : NIM output pulse, synchronous to CLOCK, to test write counters sync at each zero count. This signal is delivered individually to each Sequencer. The 13th output is for monitoring.

ERROR : NIM output intended for immediate signalization of an error condition.

TRG IN : NIM trigger pulse input from the 1st level trigger system.

TRG MON : output used in conjunction with the TRG WIN signal to adjust the phase of the trigger signal.TRG WIN : presents the time window for phasing the trigger.

TRG WAIT : NIM output signal informing the 1st level trigger that the system is busy and cannot accept new trigger requests

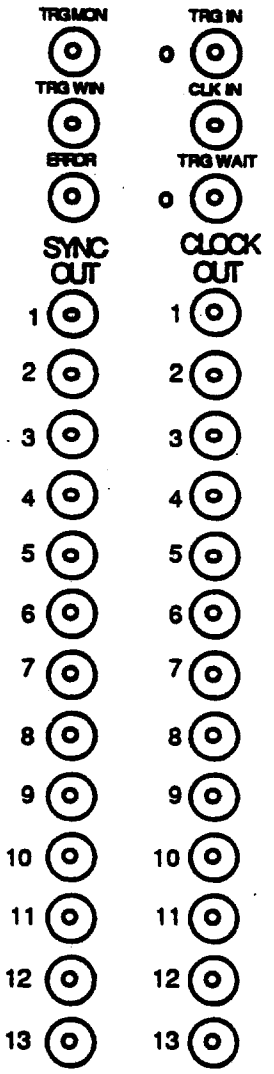


Fig 9 - Front panel Coaxial Connectors

8.3. Ribbon cable connector:

All signal in the ribbon cable connector are bussed to all crates. It is shown in Fig. 10.

RESET : initiated from Fastbus, it accomplishes system resetting and initialization. It is a differential ECL output signal, asynchronous to CLOCK.

WRITE_EN : output signal, differential ECL, asynchronous to CLOCK, for enabling the D/E modules to start data acquisition, i. e., accept hit data and increment write counters.

ENC_READY : single-ended wire OR'd ECL input signal from the Sequencers informing the D/Es status. This signal is used in determining when to deliver another

READ_ADDRESS to the Sequencers/ D/Es. It is the OR logic of the encoders not being ready.

SEQ_READY : input from the Sequencers, single-ended wire OR'd ECL, it signals that the Sequencers have room in its FIFOs for encoded events. If the Sequencers are not READY, the MTC immediately sends a WAIT to the trigger system. SEQ_READY performs the OR logic of the Sequencers not being ready.

DE_SEQ_ERR : single-ended wire OR'd ECL input from the Sequencers informing that the event FIFOs in the Sequencers have been overfilled by the encoders or that a D/E has lost synchronization.

READ_ADDRESS : 8 bit differential ECL output, asynchronous to CLOCK.

ADDRESS_VALID : it is a 80 ns wide differential ECL output pulse, asynchronous to CLOCK, occurring 80 ns after the READ_ADDRESS is asserted. It signals the Sequencers that a new READ_ADDRESS is available.

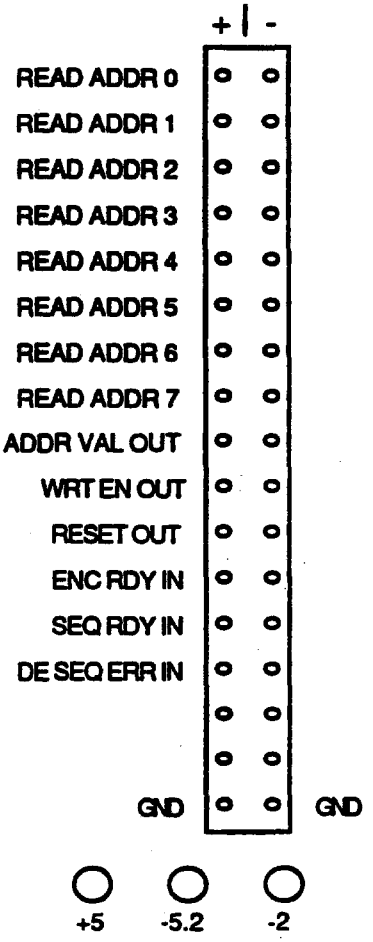


Fig 10 - Ribbon Cable connector

9. FASTBUS INTERFACE

The MTC communicates to the external world through FASTBUS. Functions executed through FASTBUS include system resetting, system calibration, system debugging, clock phase adjustment, pipeline depth setting, read address offset programming and error reporting.

The MTC is a slave device that responds to geographical address in the CSR space. The module ID is 01A2. The signals and addresses are listed below.

9.1. CSR0

CSR0 is used to control the MTC. Its signals are listed below:

bit	write	read
CSR0(00)	Set Error Flag	Error Flag
CSR0(16)	Clear Error Flag	ID
CSR0(02)	WRITE_EN Set	WRITE_EN
CSR0(18)	WRITE_EN Clear	ID
CSR0(06)	Calibration Mode set	Calibration Mode status
CSR0(07)	START_CALIB	ENC_READY

CSR0(22)	Calibration mode clear	ID
CSR0(08)	Test Mode set	Test Mode status
CSR0(09)	not used	CLOCK_MISSING
CSR0(24)	Test Mode clear	ID
CSR0(10)	not used	DE_SEQ_ERR
CSR0(11)	not used	MEM_OVWR_ERR
CSR0(12)	not used	FIFO_OVFL_ERR
CSR0(13)	not used	TRIG_PHASE_ERR
CSR0(30)	Reset	ID
CSR0(16:31)		MODULE ID (01A2)

RESET is not latched internally, and the RESET pulse is the result of writing a one to bit 30 of CSR0. A hardware reset is performed on system power up, and a software reset is issued in system initialization or after a fatal error had halted the system. RESET causes the MTC to go to the following state:

. WRITE_EN is reset
. Calibration mode is reset.
. Test mode is reset.
. All errors are cleared.
. TOFFSET, CKPHASE, and PRDPTH (programmable FIFO depth) are left unchanged.

The FB_WRITE_EN command signal to start data acquisition is set by CSR0(02) and reset by CSR0(18).

The Calibration mode is set by CSR0(06) and reset by CSR0(22).

The Test mode is set by CSR0(08) and reset by CSR0(24).

9.2. CSR10 : READ_ADDRESS

CSR10 is used in Test mode to read back the READ_ADDRESSes and to generate FB_TRIGGERs and FB_ENC_READYs.

bit	write	read
CSR10(0:7)	don't care	FIFO_RA(0:7)
CSR10(08)	FB_TRIGGER	not used
CSR10(09)	FB_ENC_READY	not used

FB_TRIGGERs and FB_ENC_READYs are not latched internally. Pulses on these lines are produced by writing a one to bit 8 and 9 of CSR10, respectively.

9.3. CSR11 : PRDPTH

CSR11 programs the trigger pipeline depth (PRDPTH). Legitimate values are 1 to 7. CSR11 is also used to monitor status used in Test mode.

bit	write	read
CSR11(0:2)	PRDPTH(0:2)	same
CSR11(4)	don't care	TRIG_WAIT
CSR11(5)	don't care	FIFO_NOT_EMPTY

The status of WAIT output on the front panel can be read through CSR11(4) which reads 1 whenever WAIT is asserted. This is intended for module diagnostic purposes.

The NOT EMPTY/EMPTY status of the FIFO can be read in CSR11(5) which reads 1 when the FIFO is NOT empty (the FIFO has READ_ADDRESSes stored in it) and 0 when the FIFO is EMPTY. This bit is intended for module diagnostic purposes.

9.4. CSR12 : TOFFSET

CSR12 holds the 8-bit trigger address offset (TOFFSET).

bit	write	read
CSR12	TOFFSET(0:7)	same

9.5. CSR13 : CKPHASE

CSR13 is used to adjust the the internal clock delay. Six bits are used to program the ELMEC PDH 6500 delay line with a .5 ns resolution.

bit	write	read
CSR13	CKPHASE(0:5)	same

9.6. Fastbus Error Responses

SS=7 Bad NTA R/W

SS=6 R/W to invalid address

SS=2 End of Block (Although not normally used, the MTC is capable of Block Transfers)

10. MODULE CALIBRATION

As said in section 3, the MTC requires the phase of the SYNC signal to phase match the clock phase. This is accomplished in SW3 (only one switch should be closed at a time) and needs to be performed only once.

11. MODULE INTERNAL SETTINGS

In addition to SW3 (SYNC phase) adjustment which is fixed for a particular module, two other dip switch adjustments are required. They are dependant on the experiment, which means that they have to be tuned to a particular system. These dip switches are SW6 for adjusting the number of triggers for Calibration mode, and SW2, used for the memory over-write margin adjustment.

SW6 is shown in Fig. 8 and explained in section 6.3 above.

SW2 setting depends on the number of clock cycles required for a READ_ADDRESS to be recognized by the D/Es, after a ADDRESS_VALID signal was issued. This number compensates for all delays due to cables, Sequencer processing and D/E acknowledging. In other words, this number should be set to some safe number (margin) that would guarantee that the D/Es' memories will retrieve the stored data before being overwritten by new data. The MTC monitors this situation by knowing the current D/Es' write address and the NEXT_READ_ADDRESS that is to be sent out.

Another switch, SW5, is provided for adjusting a time window for incoming triggers. SW5 is a rotary switch and is located in the front panel. Positions 1 to 5 are used. SW5 setting depends on how tight the external trigger phase is to be monitored.

12. POWER REQUIREMENTS

+5 Volts	1A Typ.	Fuse F2: 5A
-5.2 Volts	6A Tyo.	Fuse F3: 10A
-2 Volts	1.5A Typ.	Fuse F1: 5A

13. MTC PRELIMINARY TESTING

After assembling the MTC module, there are several tests that are required before automated software tests can be performed.

A Test Module (Appendix D) was built to provide the clock and trigger inputs to the MTC, so that the board can be tested. By providing a 53 MHz clock input to the MTC, the signals CLOCK OUT, SYNC, and TRG WIN should be present at the front panel coaxial connectors. It is important to check that all the 13 CLOCK and SYNC outputs deliver a nice NIM output. The CLOCK outputs should have approximately 50% duty cycle; the TRG_WIN width can be adjusted by a front panel rotary switch. This is the right moment to calibrate the SYNC phase delay, as explained in section 10.

14. AUTOMATED TESTS

The hardware requirements for testing the MTC are a standard Fastbus crate, with a Fastbus Smart Crate Controller, and the MTC Test Module, which provides the 53 MHz clock and the external trigger input to the MTC. Fig. 11 shows the connections between the MTC and the Test Module. The circuit diagram of the MTC Test Module is found in Appendix E.

The trigger phase, as observed in the trigger monitor output, has to be adjusted to reside within the time window presented by the TRIG_WINDOW signal (the leading edge of the window is dependent on the leading edge of the clock, and the width is set by the SW5 front panel rotary switch). See section 5.1 for a more detailed description. If the trigger phase is not correct, an error is produced and some tests will not run, since the MTC was designed to stop at an error condition.

The normal situation is when the external trigger produced by the 1st level trigger system is delayed externally in order to have the right phase. For the tests, however, the absolute phase is not important and the phase adjustment can be done by adjusting the clock phase, which changes the clock delay and, consequently, the window delay with respect to the trigger input. The trigger could also be delayed externally, using a delay line module or cable.

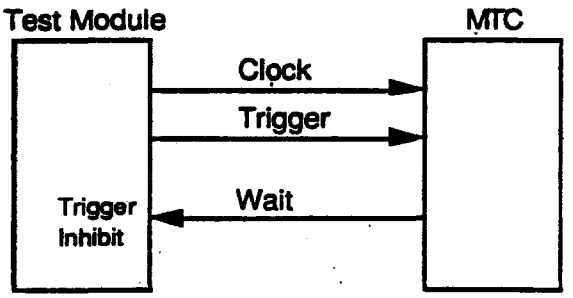


Fig. 11 - The MTC Test Module

The front panel TRIG PHZ LED goes on if the trigger phase is not correct. By providing the MTC with an external clock and an appropriate trigger input, the front panel error LEDs should go off by pressing the CLEAR ERRORS front panel push-button switch.

A program, called MTC Test Software, was developed to test the MTC. This program tests all MTC features, being able to perform system tests as well.

14.1. Test Software

The MTC Test Software is resident in the FSCC (Fastbus Smart Crate Controller), burned in EPROM. Below is presented a brief summary of what is expected from the hardware and software to accomplish the tests on the MTC.

Test Mode: External trigger requests are disregarded, triggers are generated by software (FB triggers).

Write counter is set to zero (256) and the READ_ADDRESS is controlled by the trigger offset (READ_ADDRESS = 256 - TOFFSET)

WAIT is always set in this mode.

FB_ENC_READY reads back the READ_ADDRESSes.

ADDRESS_VALID continues to be generated in the presence of errors.

Calibration Mode: Each external trigger pulse generates N (switch setable) consecutive READ_ADDRESSes. They are stored in the Trigger FIFO and send to the Sequencers/ D/Es on demand (upon receiving of a ENC_READY signal).

ADDRESS_VALID will not be generated in the presence of an error.

Calibration Mode may also be used in combination with Test Mode. This is useful primarily in checking if the MTC is generating the correct READ_ADDRESSes. The first READ_ADDRESS, in this way, can be determine by software.

Run Mode: Each external trigger generates a READ_ADDRESS; if the D/Es are not ready, the READ_ADDRESS is stored in the trigger FIFO.

The FIFO depth (number of stages in the FIFO) is programmable and WAIT is asserted if the number of FIFO stages is greater than the programmed depth. If the number of triggers exceed the maximum FIFO capacity (equal to 7), then the FIFO OVFL error will be generated.

If the read address (obtained by subtracting the offset — determined by the calibration procedure — to the reference D/E write counter of the MTC) is close by a switch setable amount to the current write address in the D/Es, a MEM_OVWR error will be issued.

Any of the above errors, or TRIGGER PHASE and D/E errors, will stop external triggers.

ADDRESS_VALID is inhibited until all errors are cleared.

MTC features that are tested

1) Fastbus interface

2) Fifo read address

3) Fifo overflow

4) Clock phase adjustment

5) Memory overwrite

6) Wait

7) Calibration

Description of tests

Pseudo code for each of the tests are:

Fastbus interface : Write and read to (from) CSR10 - CSR13;
set and reset flags (bits) and read status of CSR0.

Fifo read address :
Reset
Error flag reset
Test mode
Write enable
Trigger address offset to 256 - N
FB trigger
FB encoder ready
Read CSR10
Compare
Reset

The above test checks the arithmetic unit that evaluates the READ_ADDRESS, the trigger FIFO where this address is stored, and the control pulses FB_TRIGGER and FB_ENC_READY.

Fifo overflow :
Reset
Error flag reset
Test mode
Write enable
Set trigger pipeline depth to 7
While not overflow
  Set trigger address offset
  FB trigger
End While
Check number of triggers generated
Loop on trigger number
  Read back with Encoder Ready and
  check address
End Loop
Reset

The current FIFO depth can be found at any time by writing to the programmable depth register and observing the WAIT signal. The WAIT signal is asserted whenever the FIFO depth exceeds the programmed depth.

Clock phase adjustment : Change the clock phase over a range of values, checking for trigger phase error. The change in the clock phase causes the trigger window to move, throwing the trigger input outside the window, producing the error. For automatic tests make sure that trigger remains in phase (inside window).

Memory Overwrite :
Reset
Error flag Reset
Test mode
Write Enable
Loop on the 255 possible read addresses
  Error flag Reset
  Set trigger address offset
  Assert FB trigger
  Check Memory Overflow
  If not when address agrees with the Dip Switch, report it
  Assert FB trigger
  Read and compare address
End Loop
Reset

In the Memory Overwrite test, the memory overwrite margin switch SW2 has to be hardcoded to the value 8, or one can change the default during the initialization time to agree with the switch setting.

```
Wait :  Reset
        Error flag reset
        Loop over fifo depth
          Set fifo depth
          Set write enable
          Loop until wait or error or timeout
          When wait or error, external triggers are inhibited
          Set test mode
          Assert Fb encoder ready until fifo is empty
          Compare with depth or if error report
        End Loop
        Reset

Calibrate :  Reset
             Error flag reset
             Set trigger pipeline to 1 so wait will inhibit all  but the first external trigger
             Set calibration
             Set Write enable
             Loop until Wait
                Set Test mode
                Loop
                   Assert Fb encoder ready
                   Fill array
                   Break if fifo empty
             End Loop
             Compare and check array, report errors      ; the FIFO contents should
                                                         ; read consecutive numbers
                                                         ; representing the read addresses
             Reset
```

Notes:

Test mode is required for reading back READ_ADDRESSes in order to prevent the blocking of ADDRESS_VALIDs due to MEM_OVWR error. For every FB write to a register, the corresponding read is performed in order to check the FB interface.

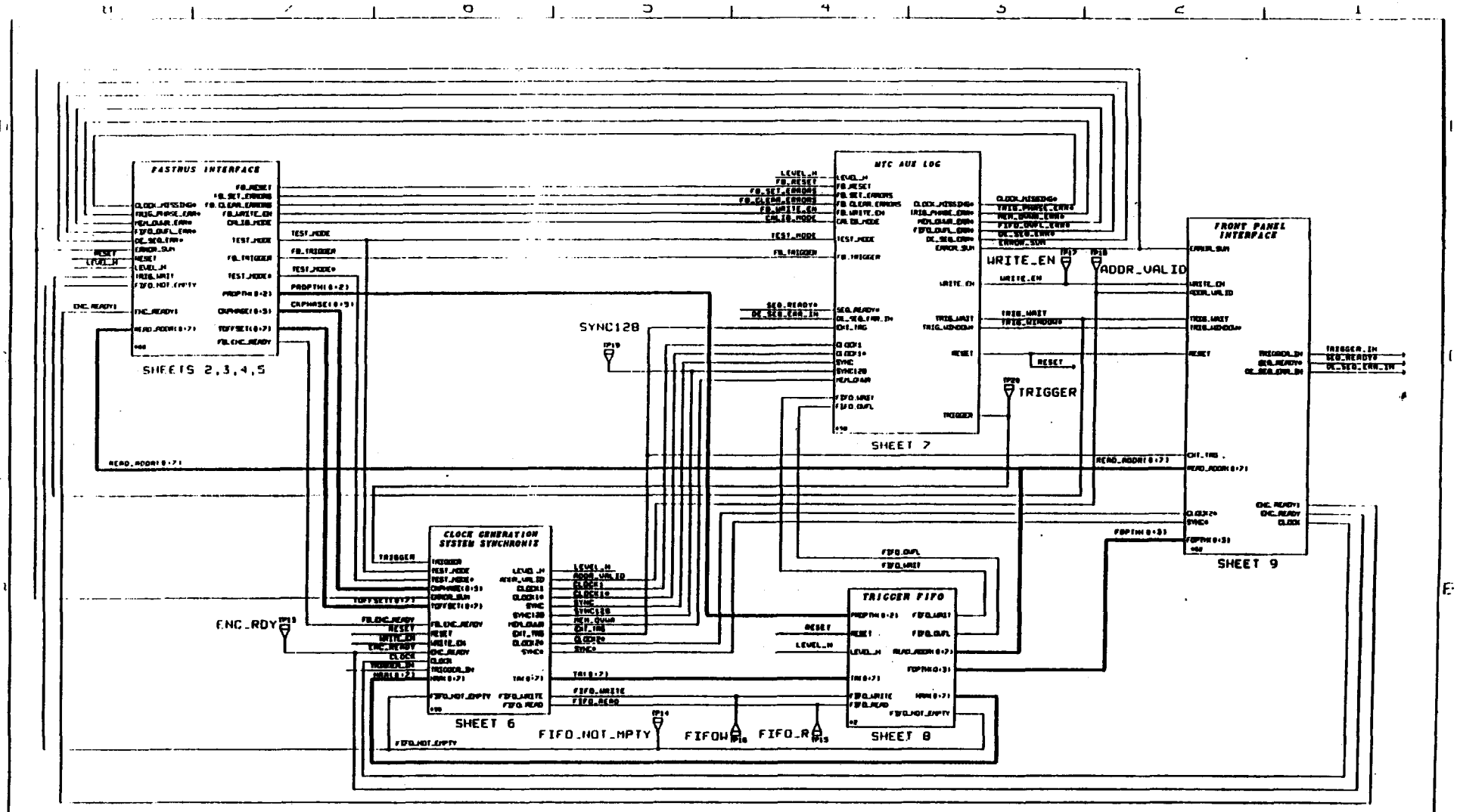
During the initialization, the default values that correspond to dip switches can be changed (main menu), or the hardcoded values can be tested to set the defaults.

The default values (i.e trigger offset, pipeline depth,...) can be modified for each test if the program is being runned in the interactive mode.

Refer to document _____ for a complete description of the MTC Test Software.

APPENDIX A

CIRCUIT DIAGRAMS

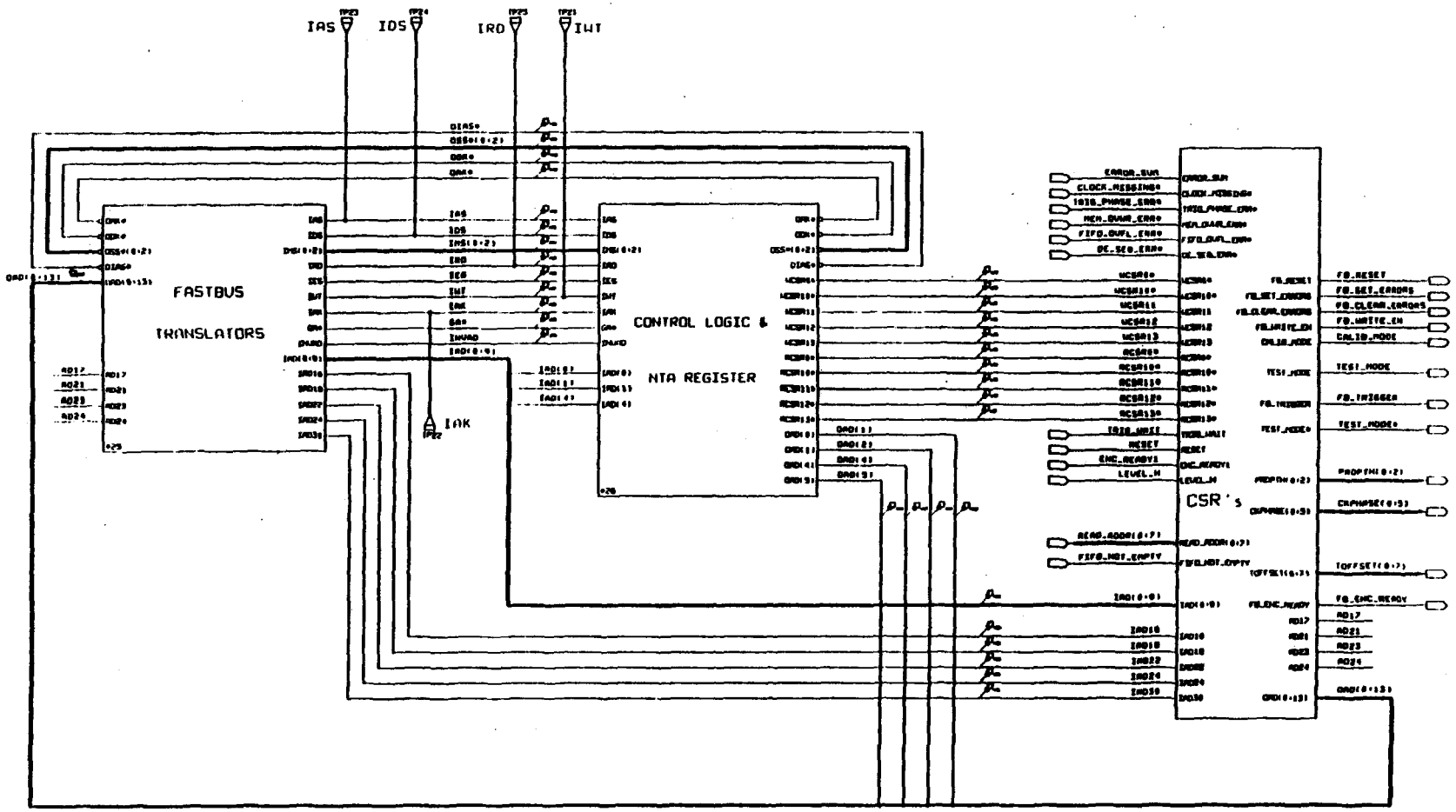


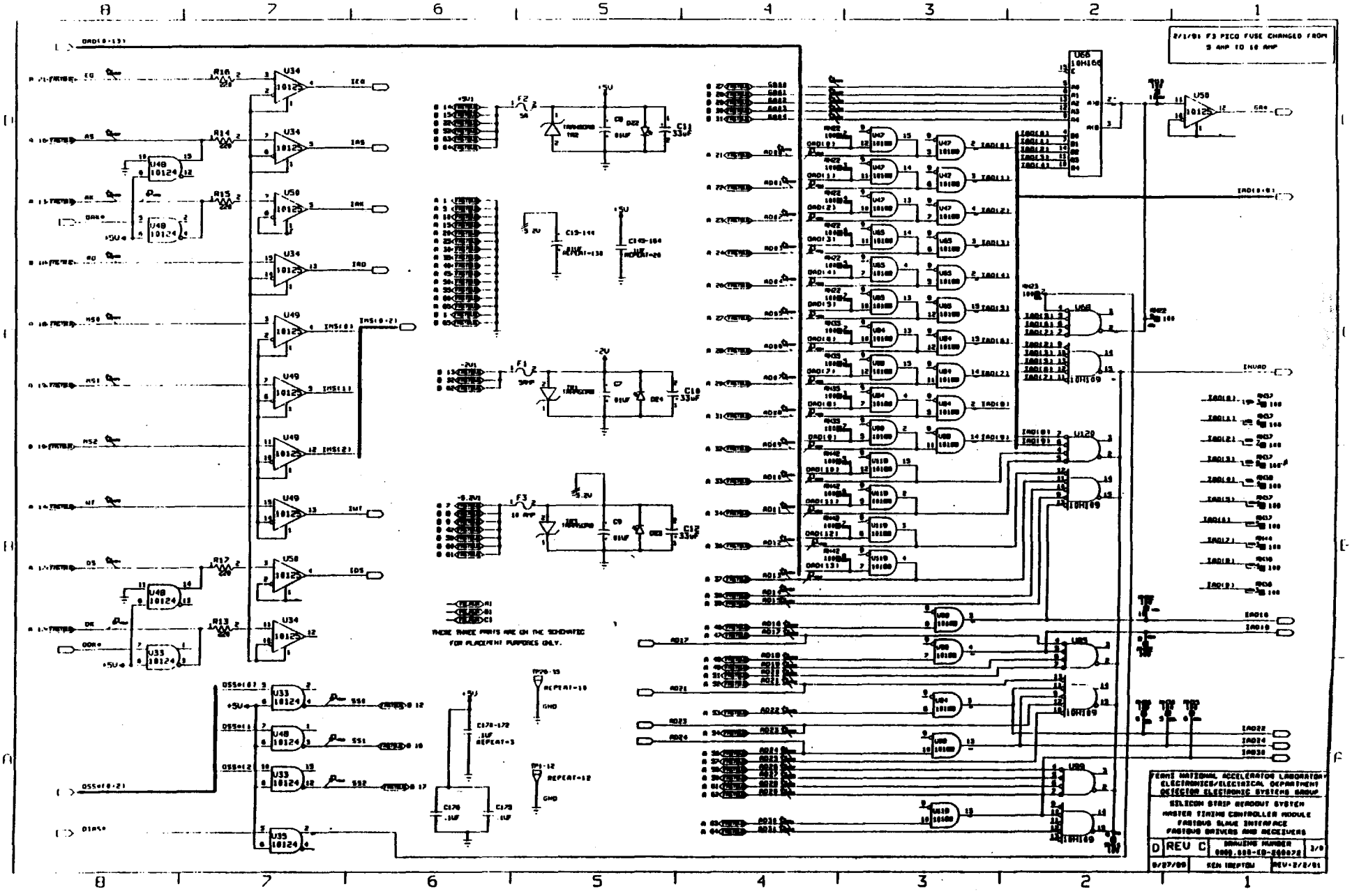
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP

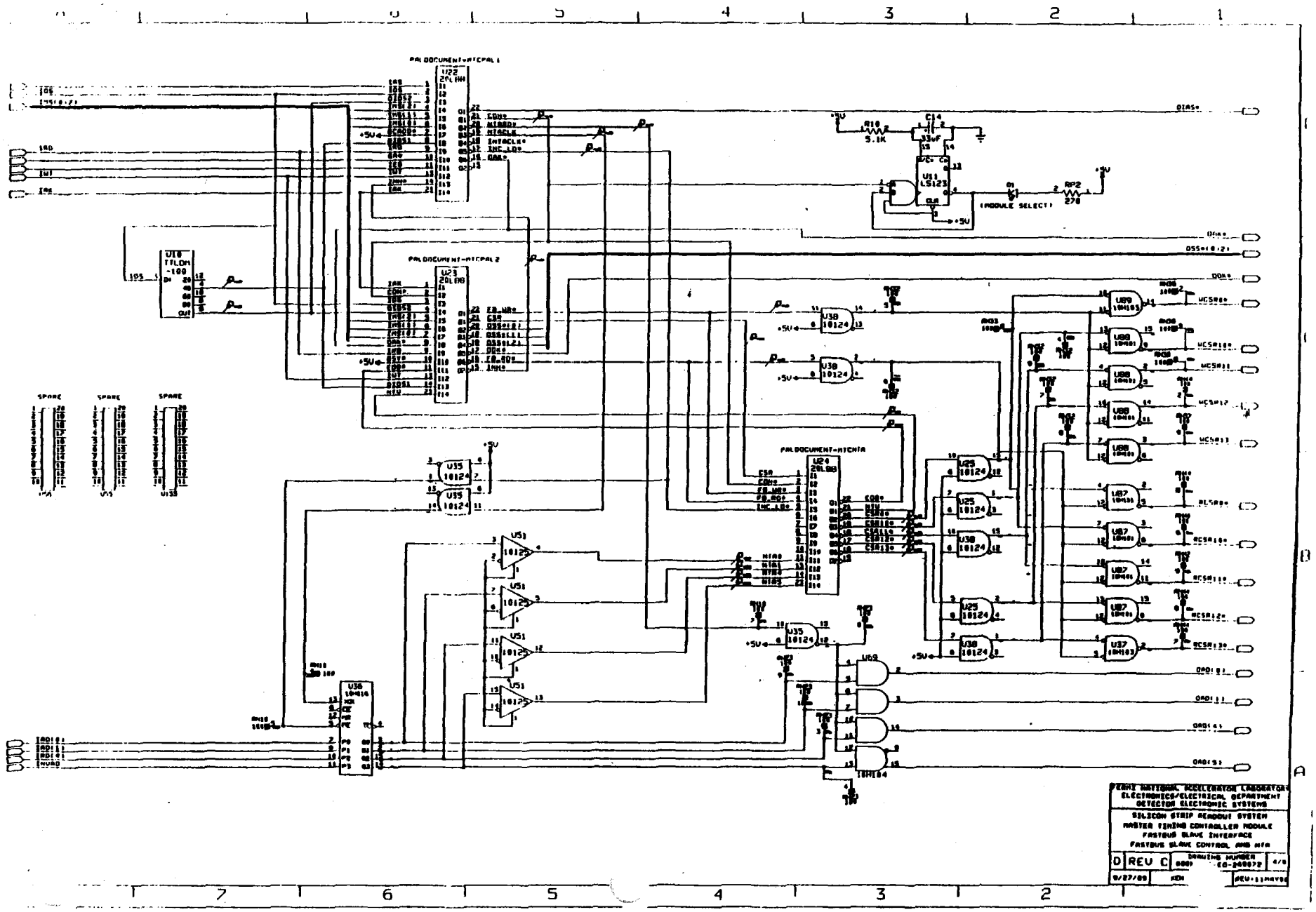
MTC BLOCK DIAGRAM

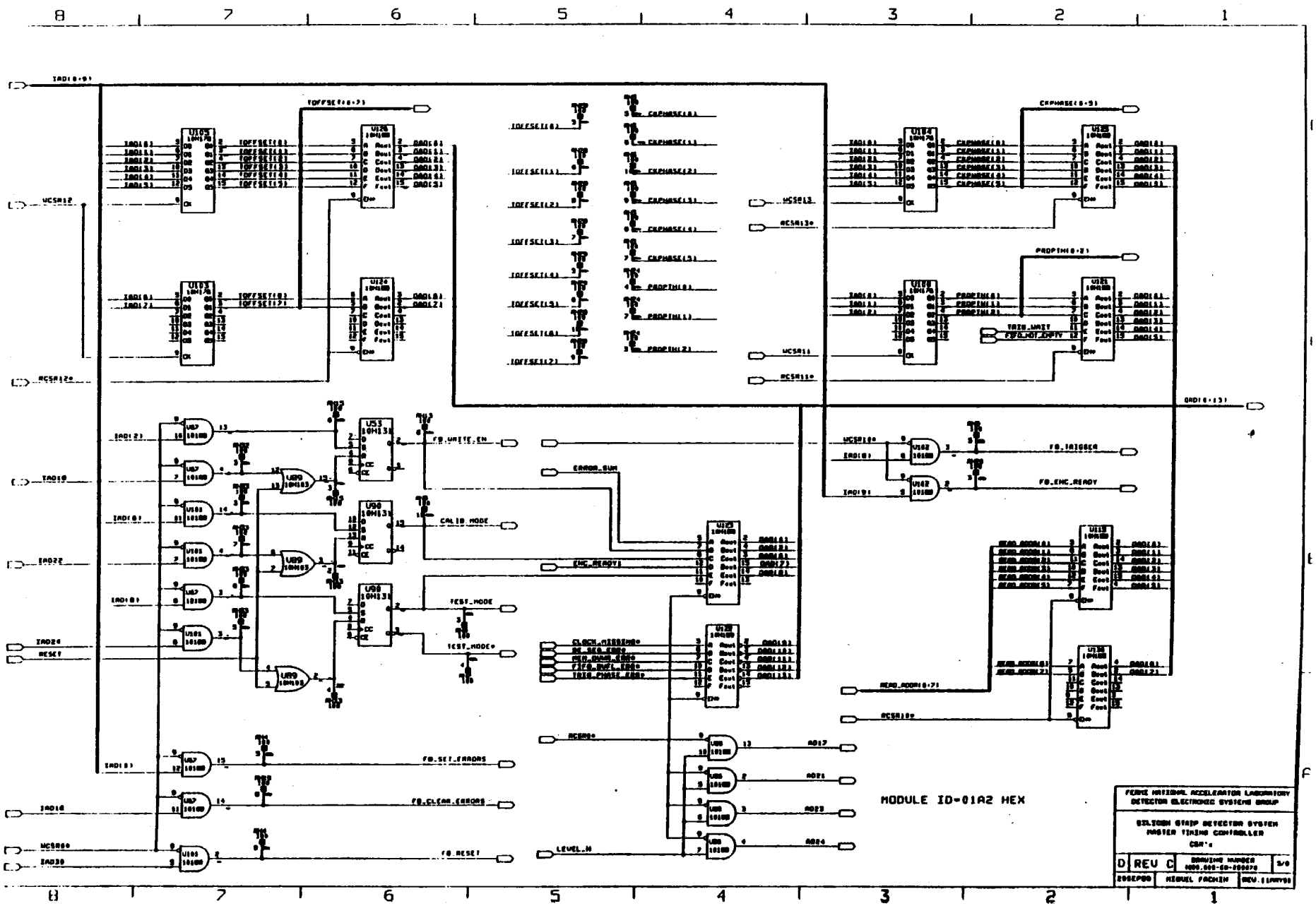
USED ON: MTC
FILE: MTC

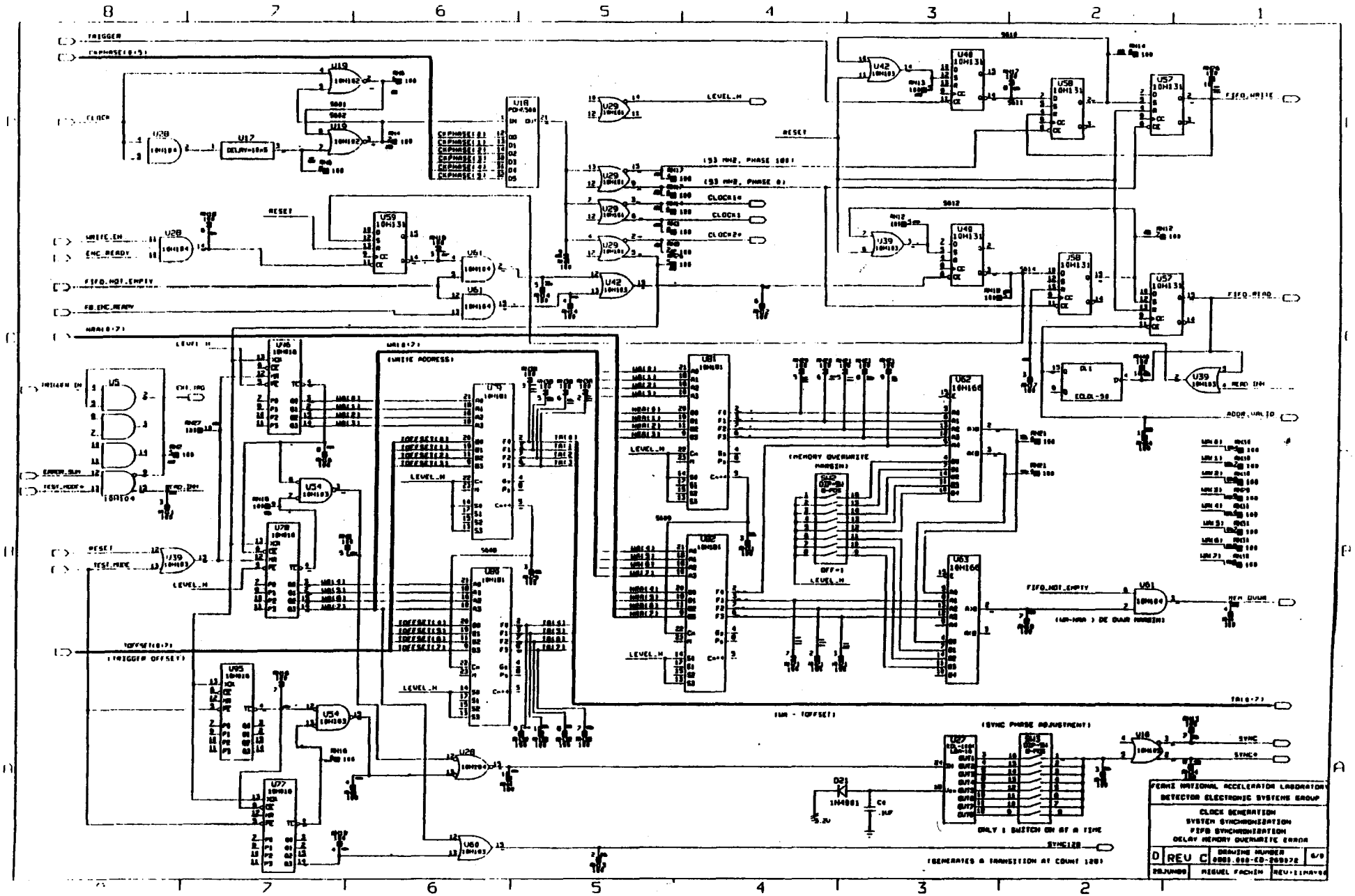
D	REV C		

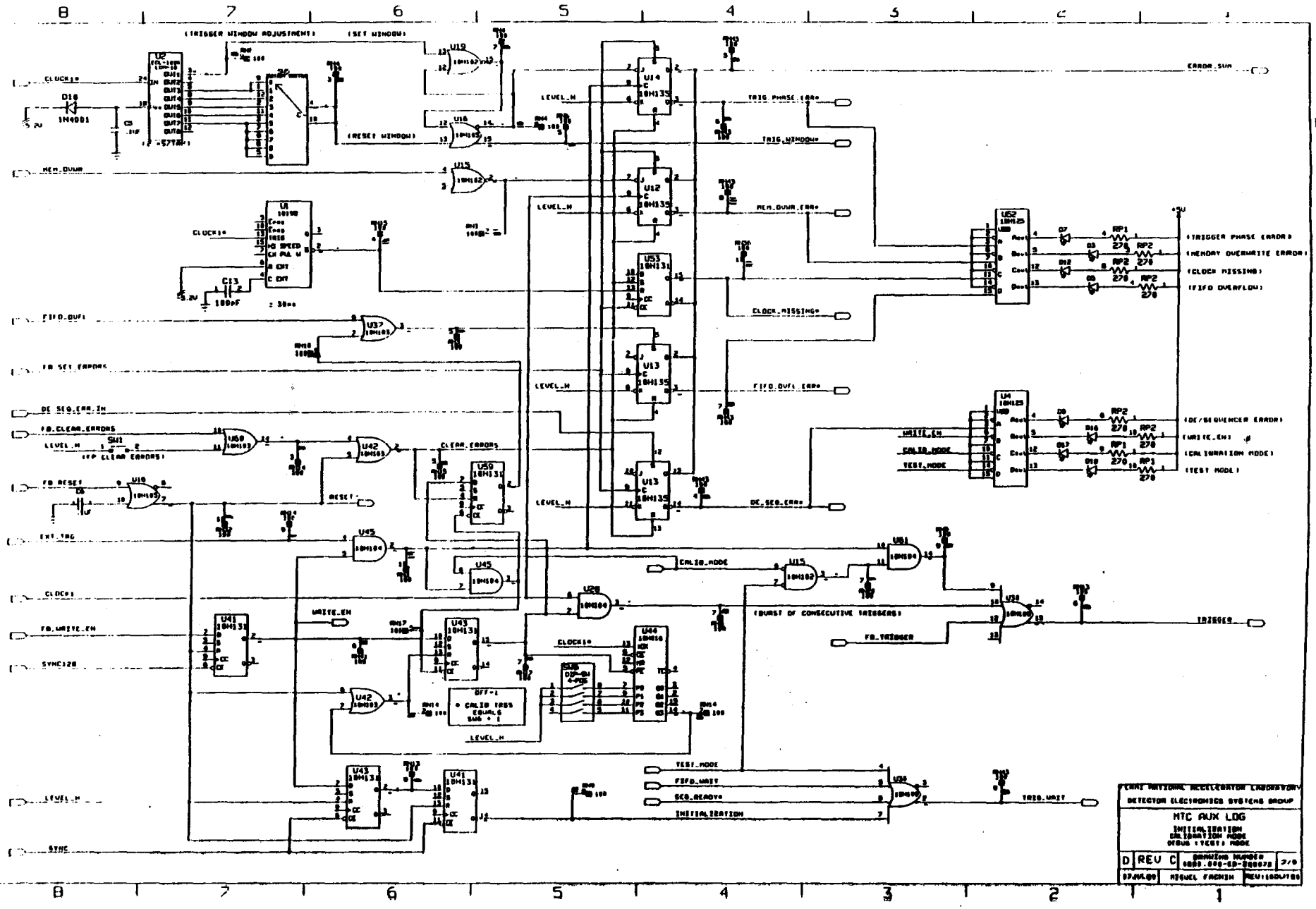

IAS
IDS
IRD
IWT
FASTBUS
TRANSLATORS
CONTROL LOGIC &
NTA REGISTER
IAK
CSR's

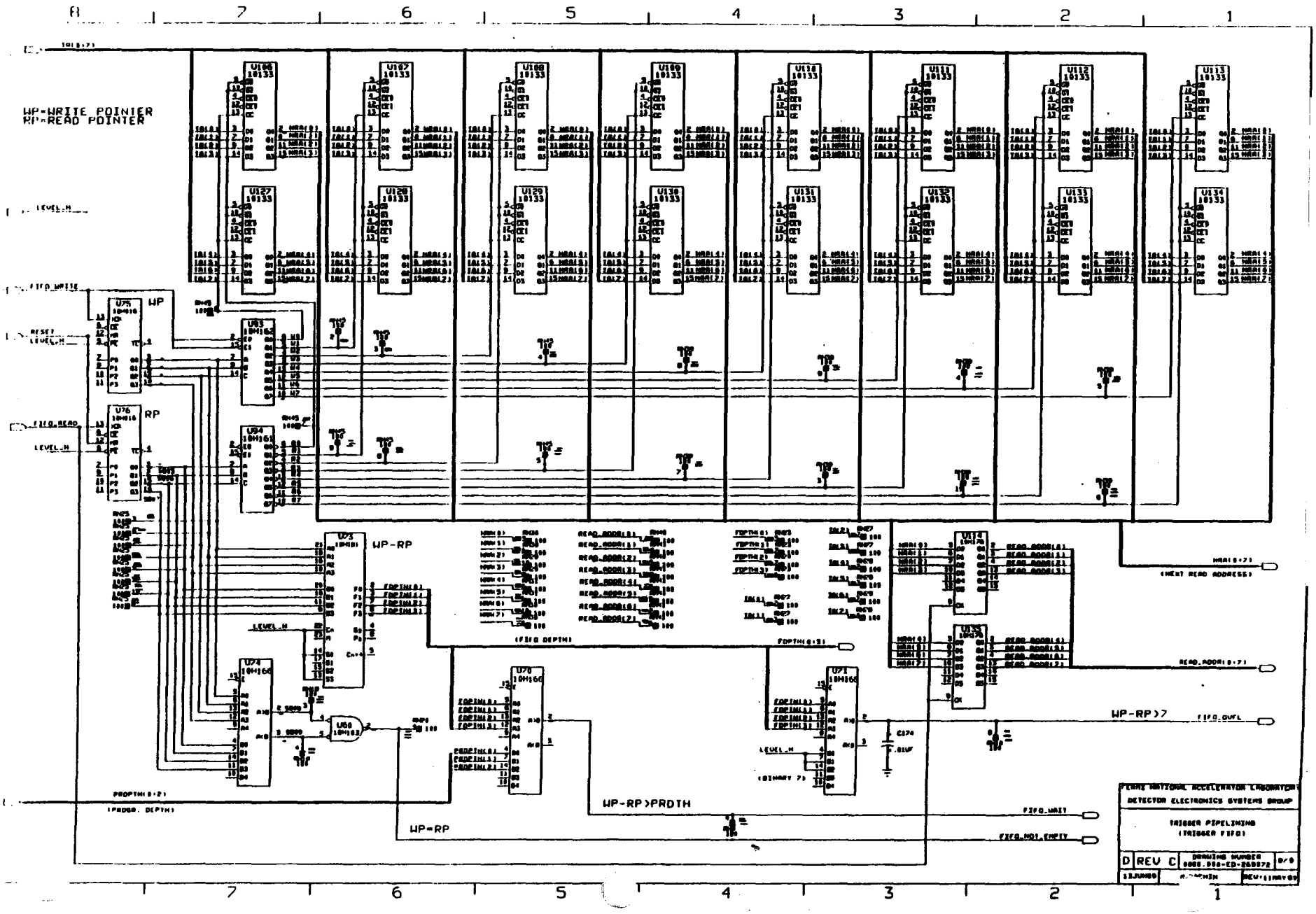

2/1/91 F3 PICO FUSE CHANGED FROM
5 AMP TO 10 AMP
FERMI NATIONAL ACCELERATOR LABORATORY
ELECTRONICS/ELECTRICAL DEPARTMENT
DETECTOR ELECTRONIC SYSTEMS GROUP
SILICON STRIP READOUT SYSTEM
MASTER TIMING CONTROLLER MODULE
FASTBUS SLAVE INTERFACE
FASTBUS DRIVERS AND RECEIVERS
D
REV C
DRAWING NUMBER
0000.000-ED-000078
3/9
8/27/90
KEN KEETON
REV-2/2/91
THESE THREE PARTS ARE ON THE SCHEMATIC
FOR PLACEMENT PURPOSES ONLY.

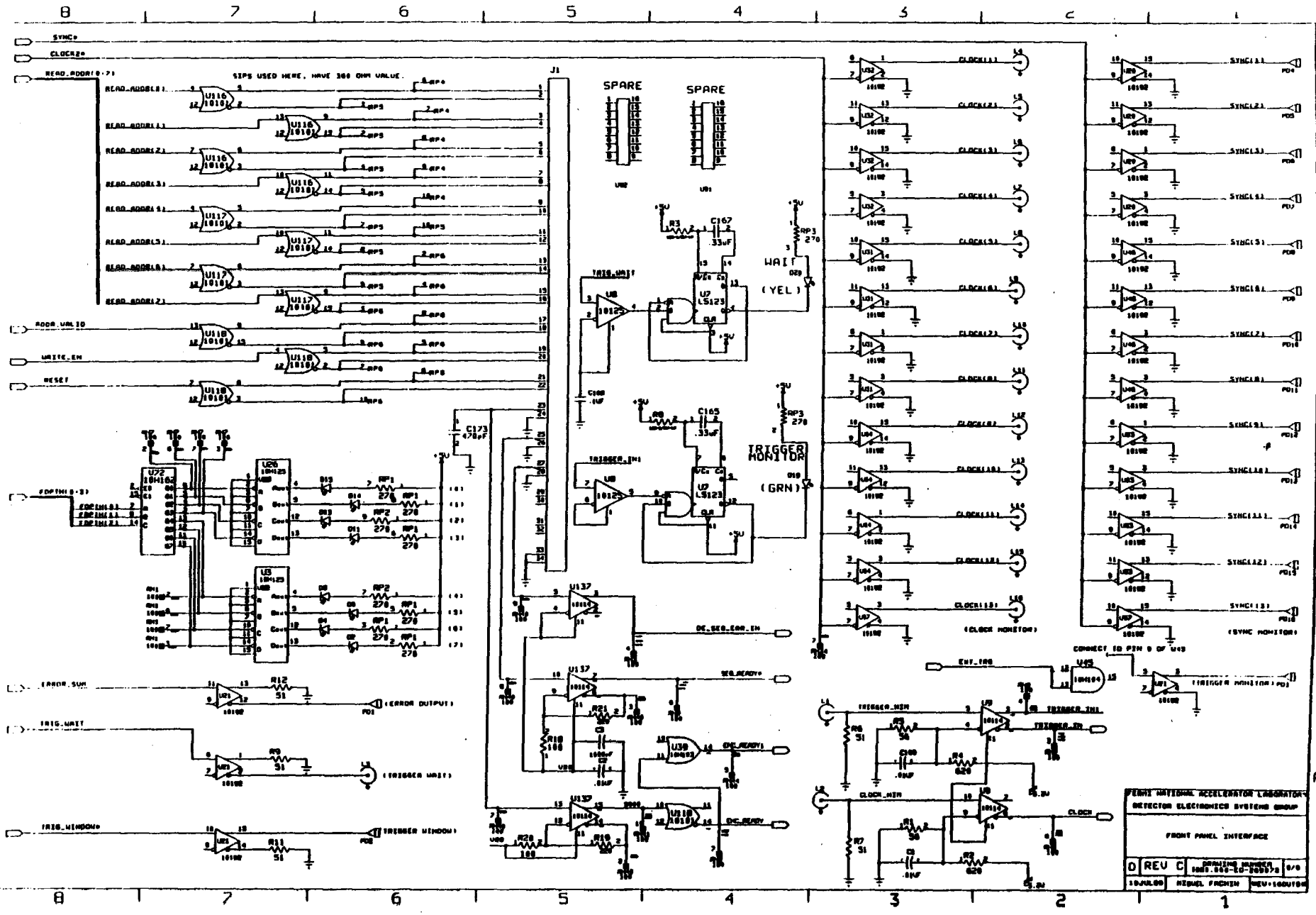

PALDOCUMENT=ATCPAL1
PALDOCUMENT=ATCPAL2
PALDOCUMENT=ATCMTA
SPARE
SPARE
SPARE
(MODULE SELECT)
5.1K
33uF
270
DIAG+
DOA+
HCSR0+
HCSR1+
HCSR2+
HCSR3+
HCSR4+
HCSR5+
HCSR6+
HCSR7+
DAD(0)
DAD(1)
DAD(2)
DAD(3)
FERMI NATIONAL ACCELERATOR LABORATORY
ELECTRONICS/ELECTRICAL DEPARTMENT
DETECTOR ELECTRONIC SYSTEMS
SILICON STRIP READOUT SYSTEM
MASTER TIMING CONTROLLER MODULE
FASTBUS SLAVE INTERFACE
FASTBUS SLAVE CONTROL AND MTA
D
REV C
DRAWING NUMBER
9/27/89
REV: 11MAY90


MODULE ID=01A2 HEX
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONIC SYSTEMS GROUP
SILICON STRIP DETECTOR SYSTEM
MASTER TIMING CONTROLLER
CSR's
REV C
MIGUEL FACHIN


TRIGGER
CLOCK
RESET
WRITE_EN
ENC_READY
FIFO_NOT_EMPTY
FB_ENC_READY
LEVEL_H
(WRITE ADDRESS)
(TRIGGER OFFSET)
(MEMORY OVERWRITE MARGIN)
(SYNC PHASE ADJUSTMENT)
ONLY 1 SWITCH ON AT A TIME
(GENERATES A TRANSITION AT COUNT 128)
FIFO_WRITE
FIFO_READ
ADDR_VALID
FIFO_NOT_EMPTY
SYNC
SYNC0
SYNC128
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONIC SYSTEMS GROUP
CLOCK GENERATION
SYSTEM SYNCHRONIZATION
FIFO SYNCHRONIZATION
DELAY MEMORY OVERWRITE ERROR
REV C
DRAWING NUMBER
6/9
REQUEL FACHIN


(TRIGGER WINDOW ADJUSTMENT)
(SET WINDOW)
(RESET WINDOW)
CLOCK10
MEM.OVWR
LEVEL_H
TRIG PHASE ERR*
TRIG WINDOW*
MEM.OVWR.ERR*
CLOCK_MISSING*
FIFO.OVFL.ERR*
DE_SEQ.ERR*
ERROR_SUM
(TRIGGER PHASE ERROR)
(MEMORY OVERWRITE ERROR)
(CLOCK MISSING)
(FIFO OVERFLOW)
(DE/SEQUENCER ERROR)
(WRITE_EN)
(CALIBRATION MODE)
(TEST MODE)
FIFO.OVFL
FB_SET_ERRORS
DE_SEQ.ERR_IN
FB_CLEAR_ERRORS
(FP CLEAR ERRORS)
CLEAR_ERRORS
FB_RESET
RESET*
EXT_TRG
CLOCK1
FB_WRITE_EN
SYNC128
WRITE_EN
CALIB_MODE
(BURST OF CONSECUTIVE TRIGGERS)
FB_TRIGGER
TRIGGER
TEST_MODE
FIFO_WAIT
SEQ_READY*
INITIALIZATION
TRIG_WAIT
SYNC
LEVEL_H
CERN NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP
MTC AUX LOG
INITIALIZATION
CALIBRATION MODE
DEBUG (TEST) MODE
D
REV C
DRAWING NUMBER
2/0
07/JUL/00
MIGUEL FACHIN


WP=WRITE POINTER
RP=READ POINTER
WP
RP
WP-RP
WP-RP>PRDTH
WP-RP
WP-RP>7
FIFO.OVFL
FIFO.WAIT
FIFO.MOT.EMPTY
(FIFO DEPTH)
(PROGR. DEPTH)
(NEXT READ ADDRESS)
TRIGGER PIPELINING
(TRIGGER FIFO)
DETECTOR ELECTRONICS SYSTEMS GROUP


SYNC*
CLOCK2*
READ_ADDR[0-7]
SIPS USED HERE, HAVE 300 OHM VALUE.
ADDR_VALID
WRITE_EN
RESET
J1
SPARE
SPARE
WAIT
(YEL)
TRIGGER MONITOR
(GRN)
C173
470pF
DATIN[0-3]
ERROR_SUM
(ERROR OUTPUT)
TRIG_WAIT
(TRIGGER WAIT)
TRIG_WINDOW*
(TRIGGER WINDOW)
(CLOCK MONITOR)
(SYNC MONITOR)
CONNECT TO PIN 9 OF U45
(TRIGGER MONITOR)
FERMI NATIONAL ACCELERATOR LABORATORY
DETECTOR ELECTRONICS SYSTEMS GROUP
FRONT PANEL INTERFACE
REV C

APPENDIX B

PAL EQUATIONS USED IN THE FASTBUS INTERFACE

MODULE MTC_NTA

```
TITLE       'FASTBUS NTA decode PAL for SSD Master Timing Controller Module
             Ken Treptow -- FERMILAB
             Sep 25, 1989 -- Revised Sep 25, 1989 '

            MTCNTA   DEVICE          'P20V8C'; " normally P20L8

"Inputs:

            CSR,!CON,!FB_WR,!FB_RD,INC      PIN 1,2,3,4,5  ;
            nc6,nc7,nc8,nc9,nc10            PIN 6,7,8,9,10 ;
            NTA0,NTA1,NTA4,NTA5             PIN 11,13,14,23  ;

"Outputs:

            !EOB,NTV,!CSR0,!CSR10           PIN 22,21,20,19  ;
            !CSR11,!CSR12,!CSR13,nc15       PIN 18,17,16,15  ;

"Constant declarations:

            X = .X. ;

            ADDR = [NTA5,NTA4,X,X,NTA1,NTA0] ;


Equations

            NTV = CON & CSR & !((ADDR == 0)                 " Not CSR0
              # ((ADDR >= ^h10) & (ADDR <= ^h13))) ;         " Not CSR10-13

            CSR0 = CON & CSR & (ADDR == 0) ;

            CSR10 = CON & CSR & (ADDR == ^h10) ;

            CSR11 = CON & CSR & (ADDR == ^h11) ;

            CSR12 = CON & CSR & (ADDR == ^h12) ;

            CSR13 = CON & CSR & (ADDR == ^h13) ;

            EOB =  CON & CSR & INC & NTV ;


END MTC_NTA
```

MODULE MTC_PAL1

```
TITLE 'FASTBUS Slave PAL1 forr SSD Master Timing Controller Module
       This is a modified FASTBUS Slave PAL1 from E706s ICBM Module '
       Ken Treptow -- FERMILAB
       Aug 25, 1989  -- Revised Aug 25, 1989 '

"Note this is a modification of FASTBUS Slave PALS done by L. PREGERNIG
" when he was at the UNIVERSITY OF ILLINOIS HIGH ENERGY PHYSICS GROUP
"DATE          1986 OCTOBER 20
"CHIP          FB009_SCL2    PAL20L8

       MTCPAL1      DEVICE       'P20V8C';  " normally P20L8

"Inputs:

       IAS,IDS,DIDS2,IMS2,IMS1,IMS0                PIN 1,2,3,4,5,6 ;
       !BCADD,DIDS1,IRD,!GA                        PIN 7,8,9,10 ;
       IEG,IWT,!INH,IAK                            PIN 11,13,14,23 ;


"Outputs:

       !DIAS,!CON,!NTARD,NTACLK                    PIN 22,21,20,19 ;
       !INTACLK,INC_LD,!OAK,nc                     PIN 18,17,16,15 ;


equations

 DIAS = IAS ; "Delayed Input AS

"CONnected (attached)

 CON = IAS & !IAK & !IRD & !IMS2 & !IMS1 & IMS0 & GA & IEG      "Geographical Add CS
       # IAS & !IAK & !IRD & !IMS2 & IMS1 & BCADD               "Broadcast Address
       # IAS & CON  ;                                           "latch while AS is up

"NTA ReaD

 NTARD = CON & IRD & !IMS2 & IMS1 & !IMS0 & IDS & !DIDS2       "set on DS up
         # NTARD & CON & IRD & !IMS2 & IMS1 & !IMS0 & IDS & DIDS2 ;
                                          "latch until new MS or Write and DS,DK up

"NTA CLocK

 NTACLK = INTACLK
          # INC_LD & !IAS                 " Inc NTA when terminating Blk xfer with AS dn
          # NTACLK & IDS & !DIDS2         "Latch while DS up
          # NTACLK & !IDS & DIDS2 ;       "Latch while DS down

"Internal NTA CLocK
```

(cont. MTC_PAL1)...

```
INTACLK = CON & !INC_LD & !IRD & !IMS2 & IMS1 & !IMS0 & DIDS1 & !DIDS2 & !NTACLK
            "NTA write cycle
          # CON & INC_LD & !INH & !IMS2 & IMS0 & IDS & !DIDS2 & !NTACLK
            "Block transfer DS up
          # CON & INC_LD & !INH & !IMS2 & IMS0 & !IDS & DIDS2 & !NTACLK
            "Block transfer DS dn
          # CON & INC_LD & !INH & IRD & !IMS2 & IMS1 & !IMS0
            & IDS & !DIDS2 & !NTACLK                    "NTA read cycle
          # CON & INC_LD & !INH & !IMS2 & !IMS1 & !IMS0
            & IDS & !DIDS2 & !NTACLK;                   "Single R/W cycle

"INCrement or LoaD the NTA

INC_LD = !(!(CON & !INH & IMS0 & IDS & DIDS2) & !INC_LD     "Set if blk
         # !IRD & !IMS2 & IMS1 & !IMS0 & IDS & !DIDS2       "Reset on NTA write
         # !IMS2 & !IMS0 & NTACLK                  "Reset on NTA or Single & after NTACLK
         # !CON ) ;                                "Reset when disconnected

"Output AK

OAK = CON & !IAK & !IWT & !BCADD                    "Send AK if not broadcast address
      # OAK & CON                                   "Latch until CON goes away
      # OAK & IWT                                   "Hold if WaiT is asserted
      # OAK & DIDS2  ;                              "Stretch until DK is off

END MTC_PAL1
```

MODULE MTC_PAL2

```
TITLE   'FASTBUS Slave PAL2 for SSD Sequencer Module
         This is a modified FASTBUS Slave PAL2 from E706s ICBM Module
         Ken Treptow -- FERMILAB
         Aug 28, 1989 -- Revised Aug 28, 1989 '

"Note this is a modification of FASTBUS Slave PALS done by L. PREGERNIG
" when he was at the UNIVERSITY OF ILLINOIS HIGH ENERGY PHYSICS GROUP
"DATE          1986 OCTOBER 20
"CHIP          FB010_SCL2    PAL20L8


      MTCPAL2      DEVICE       'P20V8C';  " normally P20L8

"Inputs:

      IAK,!CON,IDS,DIDS2,IMS2,IMS1,IMS0          PIN  1,2,3,4,5,6,7 ;
      !OAK,IRD,!BSY,!EOB,IWT,DIDS1,NTV           PIN  8,9,10,11,13,14,23 ;


"Outputs:

      !FB_WR,CSR,!OSS0,!OSS1,!OSS2               PIN  22,21,20,19,18 ;
      !ODK,!FB_RD,!OINH                          PIN  17,16,15 ;

"Constant declarations

"INHibit data transfers

 INH = CON & EOB & !IMS2 & IMS0          "If End Of Block is reached (block or pipeline)
      # CON & BSY                        "when BuSY
      # CON & NTV                        "when NoT Valid address is in the NTA
      # IMS2  ;                          "when bad MS code


Equations

"FastBus WRite strobe

 FB_WR = CON & !INH & !IRD & !IMS2 & !IMS1 & !IMS0 & DIDS1 & !DIDS2
            "MS=0 random data write, DS up only
       # CON & !INH & !IRD & !IMS2 & IMS0 & DIDS1 & !DIDS2
            "MS=1 block transfer write, DS up
            "MS=3 pipeline transfer write, DS up
       # CON & !INH & !IRD & !IMS2 & IMS0 & !DIDS1 & DIDS2  ;
            "MS=1 block transfer write, DS down
            "MS=3 pipeline transfer write, DS down

"Addressed in CSR

 CSR = CON & IMS0 & !IAK          "set if CSR at primary address time
      # CSR & CON  ;              "latch until end of CONnected (attached)


"Output Slave Status bit 0
```

(cont MTC_PAL2)

```
 OSS0 = CON & BSY & !IMS2 & !IMS1 & !IMS0 & DIDS1 & !DIDS2      "BuSY & single xfer
        # CON & BSY & !IMS2 & IMS0 & DIDS1 & !DIDS2             "BuSY and MS=1 or 3 on DS up
        # CON & BSY & !IMS2 & IMS0 & !DIDS1 & DIDS2             "BuSY and MS=1 or 3 on DS dn
        # CON & NTV & !IMS2 & IMS1 & !IMS0 & DIDS1 & !DIDS2     "NTV & 2nd Address
        # OSS0 & IDS & DIDS2 & CON                              "Latch while DS=DK=1
        # OSS0 & !IMS2 & IMS0 & !IDS & !DIDS2 & CON ;           "Latch if DS=DK=0 & MS=1or3

"Output Slave Status bit 1
 OSS1 = CON & IMS2   "any MS=4-7 on DS up or down
        # CON & !IMS2 & NTV & !BSY & DIDS1 & !DIDS2
        "any Not Valid address if not BuSY on DS up
        # CON & !IMS2 & IMS0 & NTV & !EOB & !BSY & !DIDS1 & DIDS2
        "any Not Valid address, not BuSY, not End Of Block, & MS=1or3 on DS down
        # CON & !IMS2 & IMS0 & EOB & !BSY & DIDS1 & !DIDS2
        "End Of Block if not BuSY on DS up
        # CON & !IMS2 & IMS0 & EOB & !BSY & !DIDS1 & DIDS2
        "End Of Block if not BuSY on DS down
        # OSS1 & IDS & DIDS2 & CON                              "Latch while DS=DK=1
        # OSS1 & IMS0 & !IDS & !DIDS2 & CON ;                   "Latch if DS=DK=0 & MS=1,3,5,or 7

"Slave Status bit 2
 OSS2 = CON & IMS2                                             "any MS=4-7 on DS up or down
        # CON & !IMS2 & NTV & !EOB & !BSY & DIDS1 & !DIDS2
        "any Not Valid address if not BuSY and not End Of Block on DS up
        # CON & !IMS2 & IMS0 & NTV & !EOB & !BSY & !DIDS1 & DIDS2
        "any Not Valid address, not BuSY, not End Of Block, & MS=1or3 on DS down
        # OSS2 & IDS & DIDS2 & CON                              "Latch while DS=DK=1
        # OSS2 & IMS0 & !IDS & !DIDS2 & CON ;                   "Latch if DS=DK=0 & MS=1,3,5,or 7

"Output Data acKnowlege generates DK

 ODK = !IWT & CON & OAK & DIDS2              "set if DS (delayed) and attached and not WaiT
        # IMS1 & IMS0 & CON & OAK & DIDS2
                "set if DS (delayed) and attached and MS=3 (pipeline) even if WaiT
        # ODK & CON & OAK & DIDS2             "transition hold while DS (delayed)
        # IWT & !IMS1 & ODK                   "hold if WaiT and not MS1
        # IWT & !IMS0 & ODK ;                 "hold if WaiT and not MS0
                "i.e. hold if not MS=3 (pipeline) AND WaiT, release it otherwise

"FastBus ReaD

 FB_RD = CON & !INH & IRD & !IMS2 & !IMS1 & !IMS0 & DIDS1 & !DIDS2
                "set on DS up, MS=0 random data read
          # CON & !INH & IRD & !IMS2 & IMS0 & DIDS1 & !DIDS2
                "set on DS up, MS=1 block read
                "set on DS up, MS=3 pipeline read
          # CON & !INH & IRD & !IMS2 & IMS0 & !DIDS1 & DIDS2
                "set on DS dn, MS=1 block read
                "set on DS dn, MS=3 pipeline read
          # CON & IRD & !IMS2 & !IMS1 & !IMS0 &IDS & DIDS2 & FB_RD
                "latch while MS=0 read and DS,DK up
          # CON & IRD & !IMS2 & IMS0 & FB_RD
                "latch while MS=1,3 read
          # CON & IDS & DIDS2 & FB_RD                 "latch while DS,DK up
          # CON & !IDS & !DIDS2 & FB_RD ;             "latch while DS,DK down
                                "i.e. latch until new MS or WR cycle
```

(cont MTC_PAL2)

"Output INHibit data transfers

```
OINH = INH
       # CON & !CSR ;        "This stops NTA Incrementing for the FIFO in Data Space

END MTC_PAL2
```

APPENDIX C

PARTS LIST

MTC PARTS

Last Update: 1/23/91

MTC Part No.	Manuf	Manuf PART No.	FNAL Stock N	Description	Qty	Unit Cost	Module Cost
R3,8	AB		1487-0385	10K 1/8W Res	1	0.13	0.13
R6,7,9,11,12			1478-0247	51 ohm 1/8w Res	5	0.16	0.80
R13-R17,19,21			1487-0285	220 ohm 1/8w Res	7	0.16	1.12
R18,20			1487-0620	100 ohm 1/4w Res	2	0.06	0.12
R1,R5			1487-0590	56 ohm 1/4w Res	2	0.06	0.12
R10			1487-0830	5.1K ohm 1/4w Res	1	0.06	0.06
R2,4			1487-0720	620 ohm 1/4w Res	2	0.04	0.08
RN1-RN44	AB	4610X-101		100 ohm 10-pin sip	44	0.32	14.08
RP1-RP3	AB	4610X-101		270 ohm 10-pin sip	3	0.32	0.96
RP4-RP6	AB	4610X-101		330 ohm 10-pin sip	3	0.32	0.96
				integrated circuit			
U116-U118	Motorola	10101P	1455-5801	Quad OR/NOR Gate	1	0.31	0.31
U9,137	Motorola	10114P		Triple Line Receiver	1	0.76	0.76
U25,33,35,38,48	Motorola	10124P	1455-5824	Quad TTL to ECL Translator	5	1.52	7.60
U8,34,49,50,51	Motorola	10125P	1455-5825	Quad ECL to TTL Translator	5	1.22	6.10
U106-U113	Motorola	10133p	1455-5833	Quad latch	16		
U127-U134							
U47,65,67,84,86	Motorola	10188P		Hex Buffer w/enable	9	2.20	19.80
U98,101,102							
U119							
U20,21,31,32,46	Motorola	10192P		Quad Bus Driver (ECL to Nim)	8	3.90	31.20
U64,83,97							
U1	Motorola	10198P		Monostable Multivibrator	1	8.85	8.85
U29,87,88	Motorola	10H101P		Quad OR/NOR Gate	3	0.64	1.92
U15,19	Motorola	10H102P		Quad 2 Input NOR Gate	2	0.64	1.28
U37,39,42,54,60,89	Motorola	10H103P		Quad 2 Input OR Gate	6	0.64	3.84
U28,45,61	Motorola	10H104P		Quad 2 Input AND	4	0.64	2.56
U16	Motorola	10H105P		Triple 2-3-2 Input OR/NOR	1	0.64	0.64
U30,68,85,99,120	Motorola	10H109P		Dual 5-4 Input OR/NOR	5	1.49	7.45
U75-U78	Motorola	10H016P		4-Bit Binary counter	8	7.46	59.68
U36,44,95,96							
U3,4,26,52	Motorola	10H125P		Quad ECL to TTL Translator	4	2.00	8.00
U40,41,43,53,57	Motorola	10H131P		Dual D type F/F	8	1.90	15.20
58,59,90,							
U12-U14	Motorola	10H135P		Dual JK MS Flip Flop	3	2.30	6.90
U94	Motorola	10H161P		Binary TO 1-8 Decoder (Low)	1	2.44	2.44
U72,93	Motorola	10H162P		Binary TO 1-8 Decoder (High)	2	2.44	4.88
U62,63,66,70,71,74	Motorola	10h166P		5-Bit magnitude comparator	6	3.24	19.44
U100,114,135	Motorola	10H176P		Master-slave flip-flop	6	3.54	21.24
U103-U105							
U79-U82,73	Motorola	10H181P		4-Bit alu/function generator	5	9.95	49.75
U123-U126	Motorola	10H188P		Hex Buffer w/enable	1	1.45	1.45
U115,121,136							
U122	Motorola	10H189P		Hex Inverter w/enable	1	2.20	2.20
U7,U11	T.I.	74LS123N	1455-8123	Dual Retriggerable Monostable	2	0.59	1.18
U6	T.I.	74LS02	1455-8002	Nor gate	1	0.23	0.23
		Switches					
SW2,SW3	CTS Corp.	P/N 206-8s	1455-9708	8 section dip switch	2	0.89	1.78
SW5	C&K	3M120		10 Pos. thumbwheel switch cw81ff	1	4.65	4.65
SW6	CTS Corp.	P/N 206-4s	1455-9704	4 section dip switch	1	0.75	0.75
SW1	C&K	MODEL TP11		tiny pushbutton switch	1	4.05	4.05
		connectors					
L1-L16	KINGS	K-LOCK P/N1077-3	1435-4400	Lemo PC mont	32	4.89	156.48
J1	3M	P/N 3431-5302	1435-7105	34pin reght angle header	1	1.61	1.61
FBSEG A,FBSEG B	AMP	1-102585-3		FASTBUS 130 SOCKET CONNECTOR	1	7.87	7.87
				Diodes			
D2,4,6,8,11,13,14	H.P.	P/N HLMP-1503	1445-0470	green LED	11	0.24	2.64
D15,22,23,24							
D3,5,7,9,12	H.P.	P/N HLMP-1301	1445-0475	red LED	5	0.24	1.20
D16-D20,D1	H.P.	P/N HLMP-1401	1445-0495	yellow LED	6	0.24	1.44
D10,D21		1N4001	1445-1550	Signal diode	2	0.03	0.06
				Fuse			
F1, F2	Littlefuse type 251005		1120-0250	picofuse 5A	2	0.48	0.96
F3	Littlefuse type 251010			picofuse 10A	1	0.48	0.48

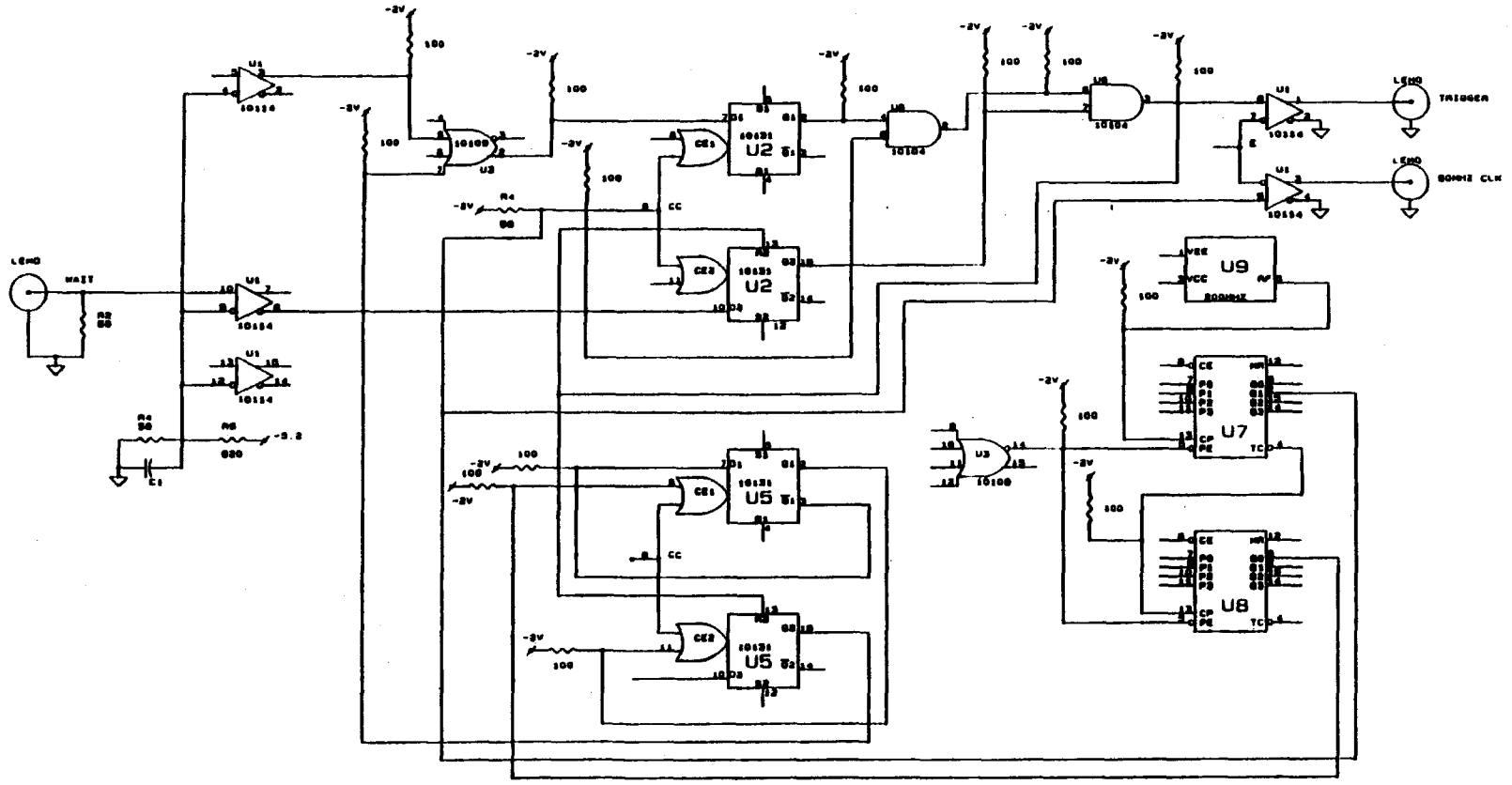
MTC PARTS

Last Update: 1/23/91

MTC Part No.	Manuf	Manuf PART No.	FNAL Stock N	Description	Qty	Unit Cost	Module Cost
				Capacitors			
C165,C167	ERIE	8131-100-651-334M	1415-3170	.33ufd ceramic cap	2	0.18	0.36
C13	SPRAGUE	P/N 10TS-T10	1415-2110	100pfd	1	0.14	0.14
C173	SPRAGUE	P/N 10TS-T47	1415-2150	470pfd cap	1	0.16	0.16
C3	SPRAGUE	P/N 10TS-D10	1415-2170	1000pfd cap	2	0.18	0.36
C174	SPRAGUE		1415-2160	680pfd	1	0.06	0.06
C10-C12,14	MALLORY	CSR13-C336K	1425-1180	33ufd cap	4	0.45	1.80
C145-164,C170-172 C4,61,68,175,176	SPRAGUE	923CZ5U104M050B		.1ufd dip cap	29	0.42	12.18
C7-C9,C15-C144 C1,2,169	SPRAGUE	923cxZr103k050b		.01ufd dip cap	137	0.36	49.32
				Delay Line			
U8	ENG.COMP.	TTLDM-100T		DELAY LINE	1	15.20	15.20
DL1	ENG.COMP.	ECLDL-50		DELAY LINE	1	13.20	13.20
U2,U25	ENG.COMP.	ECL-100K-LDM-16		DELAY LINE	2	48.70	97.40
U17	ELMEC	PDH6500		DELAY LINE	1	85.00	85.00
U16	ELMEC	FDD9010		DELAY LINE	1	9.00	9.00
				Hardware			
				MTC front panel 800.000-MD-2691(	1	40.00	40.00
				HYLON SPACER .625LONGX4-40	7	0.10	0.70

APPENDIX D

MTC TEST MODULE DIAGRAM



ORIGINATOR	NEEDLES	7/30/98	CHECKED	
DRAWN	CUNY	7/30/98	APPROVED	
FILE NAME	NEEDLES1			

FERMI NAT'L ACCELERATOR LAB.
UNITED STATES DEPT OF ENERGY

RESEARCH DIVISION
MTC TEST MODULE

SCALE	FILMED	DRAWING NUMBER	REV
NONE		0880 000-EC-269145	

APPENDIX E

CORRECTIONS TO THE PRINTED CIRCUIT BOARD

Two design changes are responsible for two modifications on the MTC' printed circuit board.

The first one is the addition of a NOR gate to invert the signal to the pin 6 of U6 (74S02). This NOR gate is in the same U6 package. The corrections to the board are shown in Fig. E-1 (the "X" represents a cut to a trace, and the hand drawn lines are the added wires).

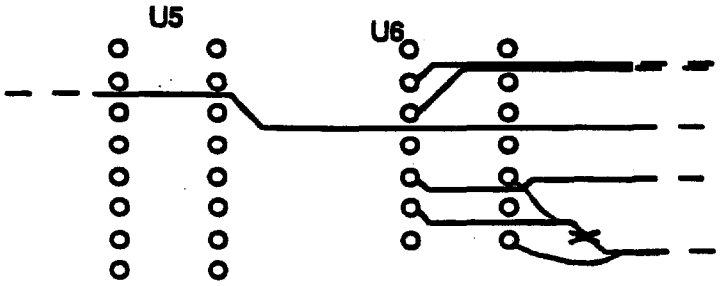


Fig E-1 - Modification number 1 of the MTC'S PCB

In the case of a new PCB design, there is a better way of implementing this logic. A suggestion is given in Fig. E-2, for the trigger rate LED and for the WAIT rate LED in schematic 9/9.

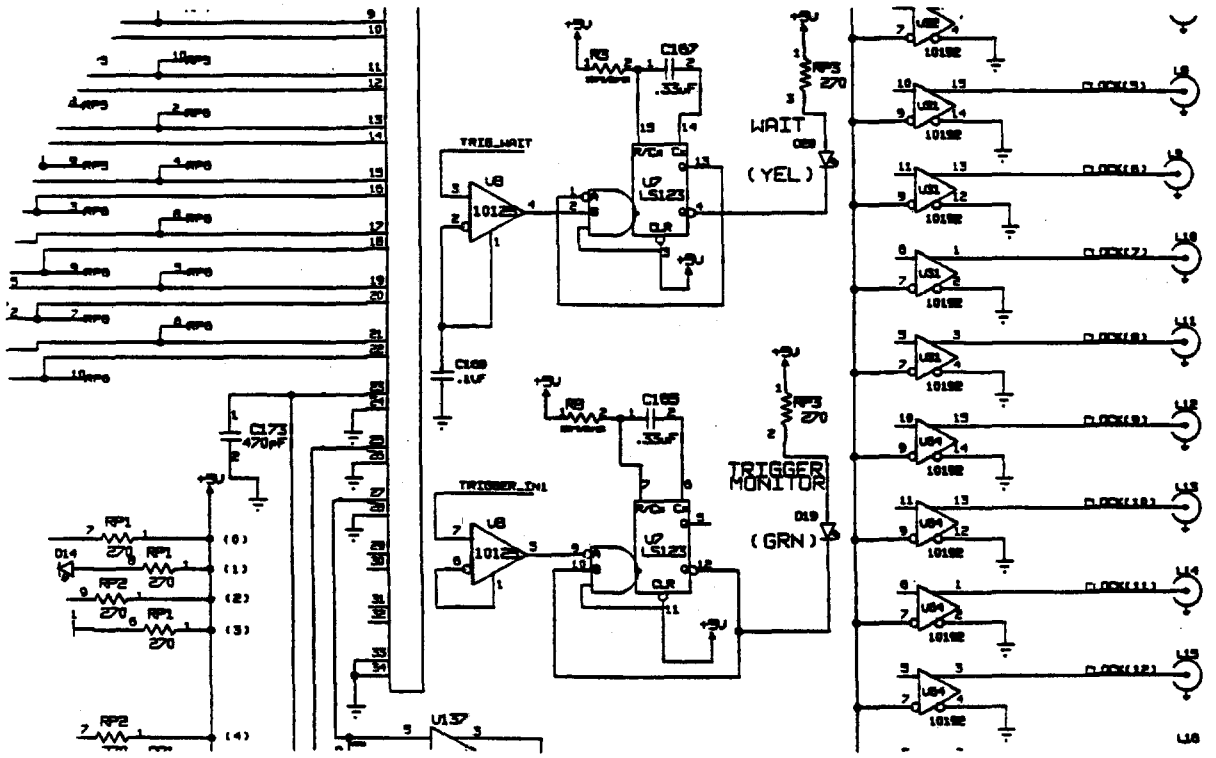


Fig E - 2 - Suggested changes to the TRG IN and TRG WAIT LED driver logic in schematic 9/9

The 2nd change was required to make the front panel TRG MON and TRG WIN signals have the right timing, as observed with a scope, for adjusting the external trigger delay to the module. In order to accomplish this, the trigger signal to the NIM driver was taken from a different point in the circuit (also an extra NAND gate was added, mainly for not messing with the ECL terminations and not to run the signal too long distances; the available gate was found in U45, inputs 12 & 13 of a 10H104, and the input signal to the gate was taken from the same U45, pin 4). The changes are shown in Fig. E-3.

The above changes in the schematics are updated in the respective Cadnetix files. The 2nd modification, however, is not handled by the Cadnetix system, since it makes use a heterogeneous gate in the 10H104 package. A note is posted in the schematic such that new PCB designs will have to find a way to bypass this problem.

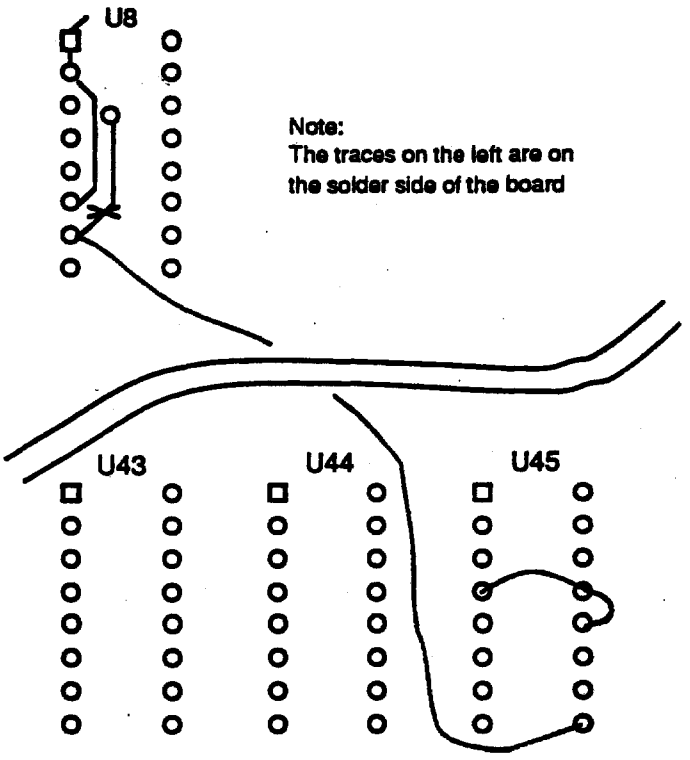
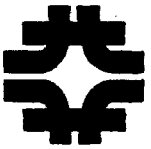


Fig E-3 - Changes to retime the TRG MON signal on the front panel coax connector



Fermilab

SSD Backplane Tests

Prototype Module

Version BP1

February 29, 1990

1. General Information

1.1. Description

This backplane is installed in a FASTBUS crate as defined in IEEE 960. It is a component part of a readout system for Silicon Strip Detectors used in experiments at Fermilab. The crate is populated as shown in Figure 1. The backplane is made in one piece to simplify the assembly of these crates. This will eliminate the requirement for external power connections to the Auxiliary connectors.

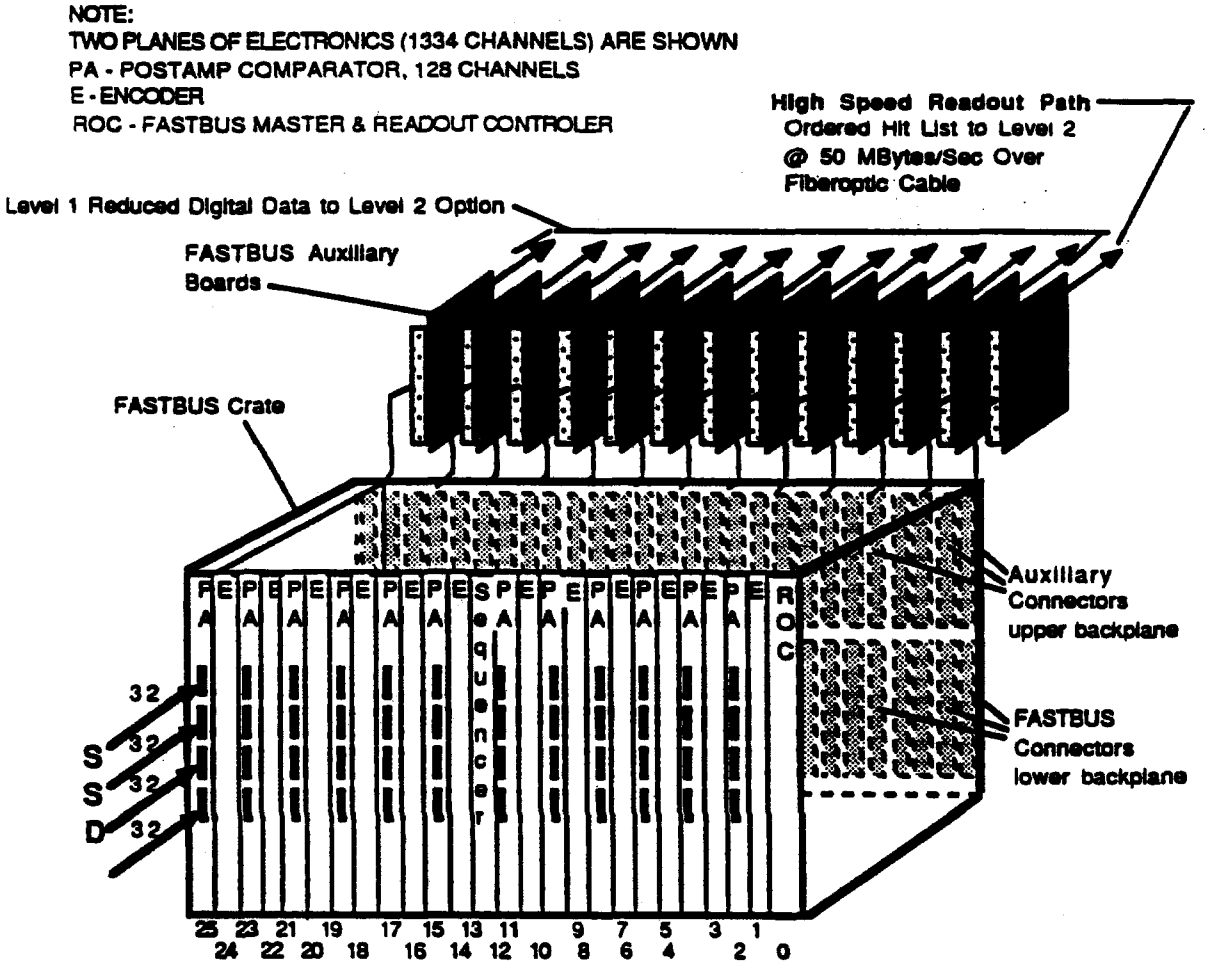


Figure 1

1.2. Specification

The backplane will be constructed of fiber glass reinforced epoxy laminate of type G-10 or FR4. All signal layers are a minimum of .003 inch copper except the Ground, +5.0 Volt and -5.2 Volt layers which are minimum of .004 inch. The thickness of the layers within the lamination must be closely controlled to provide a transmission line environment using stripline and microstrip techniques.

There are two external signal layers, four internal signal layers, three power and three ground layers. The crossection is shown in Figure 2. All signals are normal ECL. Clocks are separated physically by isolation in layers. The signals in these layers are discussed in following sections.

SSD Backplane Crossection

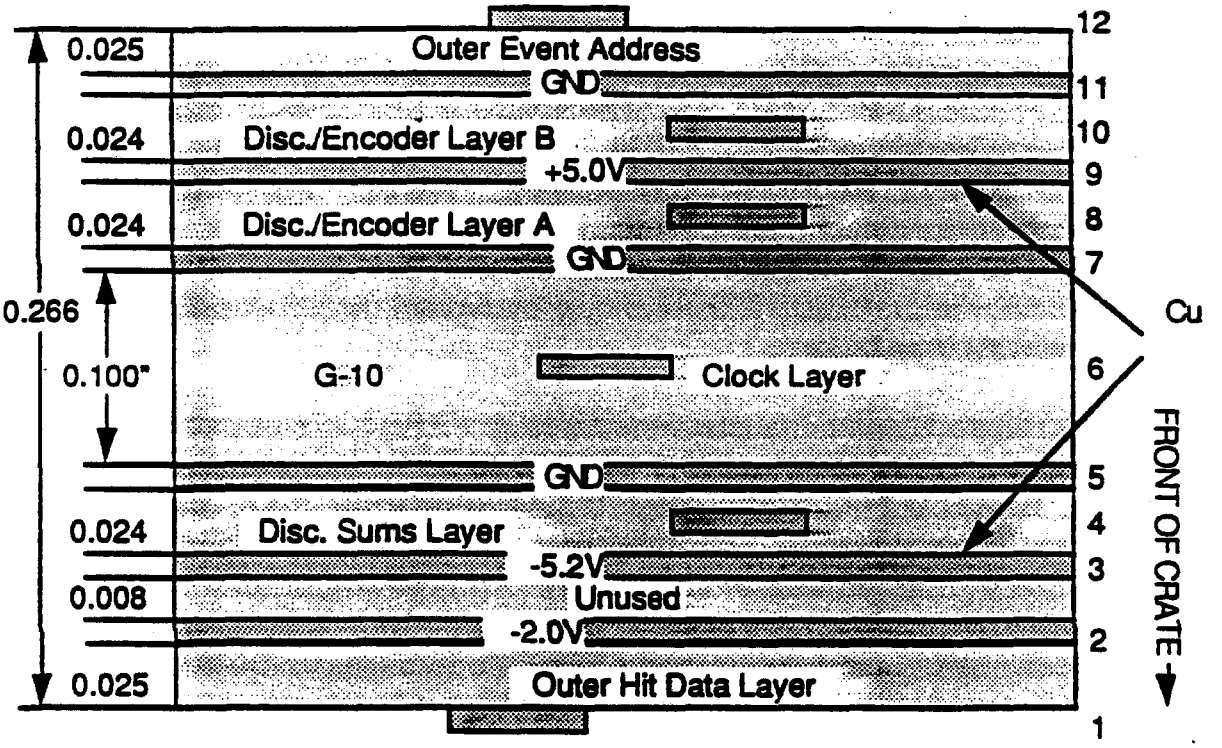


Figure 2

1.3. Test Procedure

The Prototype crates are first sent to Computing Division, Instrument Repair Group for testing of the FASTBUS Segment backplane. The Instrument Repair group has a special test stand for testing the connectivity and mechanical specifications of a FASTBUS crate.

After this is complete a crate will be sent to Kinetic Systems Inc. which has a power distribution test stand for FASTBUS crates. They have agreed to do this part of the testing.

At the same time the Computing Division, Data Acquisition Electronics Department will begin testing the Auxiliary backplane of the other prototype.

2. Mechanical

2.1. Specification

The mechanical dimensions and mounting provisions of the backplane must fit the backplane area of a Kinetics System FASTBUS Crate model # F050-A11. A mechanical drawing of the backplane, showing mounting holes and connector positions, is attached. The lower backplane is detailed in chapter 14 of the FASTBUS specification, IEEE 960. Power connections must conform to the mechanical requirements of the Kinetic Systems Crate named above. This rest of this document refers to the Auxiliary section of the backplane. Where the IEEE 960 Auxiliary backplane specifications differ, this document shall be used.

All Auxiliary backplane connectors are three row 195 pin connectors. The connectors are as specified in chapter 14 of IEEE 960 and figures 14.2 (a & b) of IEEE 960 and addendum's to the specification. The Auxiliary connectors are provided with a shroud/card guide as shown in figure 14.2(c) of IEEE 960 or guide pins as shown in detail A of the supplied mechanical drawing. The connectors have daughter cards on both sides of the Auxiliary backplane. Daughter card connectors will conform to appendix K of IEEE 960. Daughter cards will be plugged into both sides of the backplane.

2.2. Test Procedure

The Instrument Repair Group has a special test stand for FASTBUS crates and test jigs to check dimensional tolerances.

2.3. Test Result

The terminator sips for the Auxiliary backplane interfere with the insertion of modules. Interlogic will Modify crates to place the sips on the back.

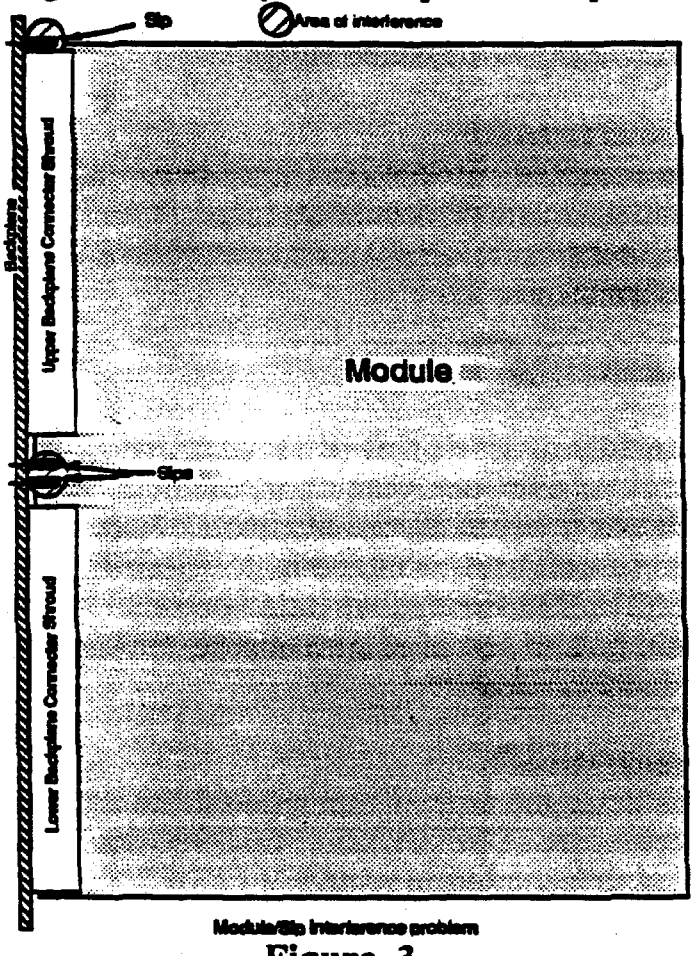


Figure 3

3. Power Distribution

3.1. Auxiliary Backplane

3.1.1. Specification

Power and ground distribution for the Auxiliary section of the backplane is through internal power layers in the backplane. There are separate layers for +5.0 Volts @ 120 Amps, -5.2 Volts @ 120 Amps, -2.0 Volts @ 50 Amps and Ground @ 290 Amps. The 26 MHZ and 53 MHZ clocks are in an internal layer with ground planes on either side. There are no active components on the Auxiliary backplane. The power should be distributed in a way that insures a voltage drop across the backplane of less than 0.01 Volts for the ground plane and 0.030 Volts for voltage planes. Termination resistors are socketed.

The power pins will be the same for all Auxiliary cards except the Sequencer module slot 13 and slot 0 which has none. The large number of pins required on some modules means it won't be possible to arrange the power pins a per IEEE 960. There is a limit of 3 amps per Auxiliary card (except the Sequencer slot Auxiliary card) for all supplies.

3.1.2. Test Procedure

As the power requirement for the Auxiliary backplane is significantly less than the Segment backplane no special tests are required if the crate passes the test at the Segment backplane. A simple check will be made of the voltage drops at the Auxiliary connector.

3.1.3. Test Result

3.1.4. Additional Power Bus Required for Postamp/Comp.

There is the possibility of using existing contacts on the FASTBUS crate segment connector for bringing the plus and minus 3.5 volt power supply voltages to the Postamp/Comparator modules. These voltages are required for some of the ASIC's used on this module. In order to determine the feasibility of this idea, a series of measurements should be conducted on a typical crate used in the SSD project.

3.1.4.1. Specification

The FASTBUS Crate segment pins are specified to handle a maximum of three amperes each. (See ANSI/IEEE Std 960-1986, section 13.2.1 (e)). Each P/C Module is expected to require +3.5 V @ 7.07 amperes and -3.5 V @ 3.52 amperes. Thus three pins for the +3.5 and 2 pins for the -3.5 should be adequate for each P/C Module.

There are to be twelve P/C Modules per crate requiring +3.5 V @ 12x7.07 = 85 amperes and -3.5 V @ 12x3.52 = 42 amperes per crate.

The following 8 existing busses are being considered.

1) +28 Volts (pin B02.stations 0 through 12)::[06 and 07]
2) +28 Volts (pin B02.stations 13 through 25)::[19 and 20]
3) +28 Volts (pin B03.stations 0 through 12)::[04 and 05]
4) +28 Volts (pin B03.stations 13 through 25)::[17 and 18]
5) +15 Volts (pin B04.stations 0 through 12)::[05 and 06]
6) +15 Volts (pin B04.stations 13 through 25)::[18 and 19]
7) -15 Volts (pin B05.stations 0 through 12)::[07 and 08]
8) -15 Volts (pin B05.stations 13 through 25)::[20 and 21]

The information inside the brackets "[ ]" represents station numbers between which the power supply connection to the bus exists.

In order to determine the quality of the bus, a few simple measurements should be made. In general, a current source will be connected across a portion of the bus structure, and the resulting voltages drops for various portions of the backplane bus will be recorded. The current source to be used should have an output of 2 amperes plus or minus 2 percent, an output impedance of at least 1 kΩ, and an output voltage of 1/2 volt,.plus or minus 1/4 volt. One might expect to measure between 5 and 10 millivolt drops between stations where current is flowing.

3.1.4.2. Test Procedure and Result

Instruments used:

HP 62278 Power Supply
Keithley 179A DVM

Bus 1 (Upper +28 volt bus, stations 0 through 12)

Connect one lead of the current source to pin B02 at station 0 , and the other lead to the bus connection between stations 06 and 07. Use a Digital Volt Meter (DVM) capable of measuring 0.1 millivolt changes. Connect one lead of the DVM to pin B02 at station 01,and connect the other lead to the pin B02 of the following stations and record the voltage.

Station	Millivolts
02	0.55
03	1.08
04	1.52
05	1.98
06	3.03
07	3.03
08	2.94
09	2.93

Connect one lead of the current source from pin B02 at station 12, and the other lead to the bus connection between stations 06 and 07. Connect one lead of the DVM to pin B02 at station 11, and connect the other lead to the pin B02 of the following stations and record the voltage.

Station	Millivolts
02	0.49
03	0.96
04	1.46
05	2.10
06	2.46
07	2.29
08	2.27
09	2.28

Bus 2 (Upper +28 volt bus, stations 13 through 25)
Connect one lead of the current source from pin B02 at station 13, and the other lead to the bus connection between stations 19 and 20. Connect one lead of the DVM to pin B02 at station 14. Connect the other lead to the pin B02 of the following stations and record the voltage.

Station	Millivolts
02	0.51
03	0.99
04	1.45
05	1.88
06	2.87
07	2.89
08	2.82
09	2.82

Connect one lead of the current source from pin B02 at station 25, and the other lead to the bus connection between stations 19 and 20. Connect one lead of the DVM to pin B02 at station 24. Connect the other lead to the pin B02 of the following stations and record the voltage.

Station	Millivolts
02	0.56
03	1.09
04	1.56
05	2.24
06	2.58
07	2.43
08	2.43
09	2.43

Bus 3 (Lower +28 volt bus, stations 0 through 12)
Connect one lead of the current source to pin B03 at station 0 , and the other lead to the bus connection between stations 04 and 05. Connect one lead of the DVM to pin B03 at station 01,and connect the other lead to the pin B03 of the following stations and record the voltage.

Station	Millivolts
02	0.70
03	1.43
04	2.19
05	2.31
06	2.31
07	2.31
08	2.31
09	2.32

Connect one lead of the current source from pin B03 at station 12, and the other lead to the bus connection between stations 04 and 05. Connect one lead of the DVM to pin B03 at station 11, and connect the other lead to the pin B03 of the following stations and record the voltage.

Station	Millivolts
02	0.75
03	1.50
04	2.24
05	2.31
06	4.27
07	5.27
08	5.81
09	5.78

Bus 4 (Lower +28 volt bus, stations 13 through 25)
Connect one lead of the current source from pin B03 at station 13, and the other lead to the bus connection between stations 17 and 18. Connect one lead of the DVM to pin B03 at station 14. Connect the other lead to the pin B03 of the following stations and record the voltage.

Station	Millivolts
02	0.77
03	1.53
04	2.34
05	2.47
06	2.46
07	2.46
08	2.46
09	2.47

Connect one lead of the current source from pin B03 at station 25, and the other lead to the bus connection between stations 17 and 18. Connect one lead of the DVM to pin B03 at station 24. Connect the other lead to the pin B03 of the following stations and record the voltage.

Station	Millivolts
02	0.78
03	1.56
04	2.32
05	3.41
06	4.41
07	5.44
08	5.99
09	5.97

Bus 5 (+15 volts, stations 0 through 12)

Connect one lead of the current source to pin B04 at station 0 , and the other lead to the bus connection between stations 05 and 06. Connect one lead of the DVM to pin B04 at station 01,and connect the other lead to the pin B04 of the following stations and record the voltage.

Station	Millivolts
02	0.20
03	0.39
04	0.55
05	0.86
06	0.85
07	0.73
08	0.69
09	0.67

Connect one lead of the current source from pin B04 at station 12, and the other lead to the bus connection between stations 05 and 06. Connect one lead of the DVM to pin B04 at station 11, and connect the other lead to the pin B04 of the following stations and record the voltage.

Station	Millivolts
02	0.20
03	0.38
04	0.51
05	0.74
06	0.98
07	1.09
08	0.95
09	0.92

Bus 6 (+15 volts, stations 13 through 25)
Connect one lead of the current source from pin B04 at station 13, and the other lead to the bus connection between stations 18 and 19. Connect one lead of the DVM to pin B04 at station 14. Connect the other lead to the pin B04 of the following stations and record the voltage.

Station	Millivolts
02	0.21
03	0.40
04	0.56
05	0.86
06	0.84
07	0.74
08	0.69
09	0.68

Connect one lead of the current source from pin B04 at station 25, and the other lead to the bus connection between stations 18 and 19. Connect one lead of the DVM to pin B04 at station 24. Connect the other lead to the pin B04 of the following stations and record the voltage.

Station	Millivolts
02	0.21
03	0.39
04	0.53
05	0.78
06	1.03
07	1.13
08	0.98
09	0.94

Bus 7 (-15 volts, stations 0 through 12)

Connect one lead of the current source to pin B05 at station 0 , and the other lead to the bus connection between stations 07 and 08. Connect one lead of the DVM to pin B05 at station 01,and connect the other lead to the pin B05 of the following stations and record the voltage.

Station	Millivolts
02	0.33
03	0.56
04	0.75
05	0.93
06	1.21
07	1.73
08	1.53
09	1.33

Connect one lead of the current source from pin B05 at station 12, and the other lead to the bus connection between stations 07 and 08. Connect one lead of the DVM to pin B05 at station 11, and connect the other lead to the pin B05 of the following stations and record the voltage.

Station	Millivolts
02	0.38
03	0.68
04	1.05
05	1.40
06	1.05
07	0.94
08	0.90
09	0.88

Bus 8 (-15 volts, stations 13 through 25)
Connect one lead of the current source from pin B05 at station 13, and the other lead to the bus connection between stations 20 and 21. Connect one lead of the DVM to pin B05 at station 14. Connect the other lead to the pin B05 of the following stations and record the voltage.

Station	Millivolts
02	0.34
03	0.57
04	0.75
05	0.96
06	1.24
07	1.75
08	1.54
09	1.35

Connect one lead of the current source from pin B05 at station 25, and the other lead to the bus connection between stations 20 and 21. Connect one lead of the DVM to pin B05 at station 24. Connect the other lead to the pin B05 of the following stations and record the voltage.

Station	Millivolts
02	0.38
03	0.69
04	1.08
05	1.43
06	1.08
07	0.97
08	0.93
09	0.91

3.2. Segment Backplane

3.2.1. Specification

The Segment backplane power distribution will be as specified in IEEE 960, Chapters 14 and 15.

3.2.2. Test Procedure

A prototype crate was tested at Kinetic Systems Inc. for evaluation on a specially built FASTBUS crate power test stand. The test stand consists of a small rack and Kinetic Systems standard FASTBUS power supply. A shunt is provided to be inserted in the power leads to allow measurement of total current for a given supply voltage. Special load modules composed of FASTBUS cards with dip and sip terminators across the boards are used to load the crate. These load modules are designed to allow use with -5.2 Volts, +5.0 Volts and -2.0 Volts. The modules can be used with only one voltage at a time, this is accomplished through the use of jumper wires along the edges of the modules. These modules thus provide a uniform current load throughout the crate for one supply voltage at a time.

The test procedure is to fill the crate with test modules jumpered for one of the supply voltages and measure the voltage drop from the power bus to the pins at the top and bottom of each slot. This yeilds only the differential voltage across the backplane. Each voltage is tested in turn.

3.2.3. Test Result

Test results for crate serial #0553 are shown in tables below:

-5.2 Volt bus

Bus to Ground voltage -4.22 V (supplies current limited @ 320 Amperes.)
Current 320 Amps. (measured with 50 mv/500 Ampere shunt)
Measured at the top and bottom of the crate segment connector, all values are in millivolts.

Slot #	0	3	5	7	11	15	17	19	20	21	22	23	24	25
Top	33	33	33	33	33	34	34	35	35	36	36	37	37	37
Bot.	23	21	20	20	20	20	21	22	23	25	27	30	31	32

+5 Volt bus

Bus to Ground voltage +4.61 V (supplies current limited @ 340 Amperes.)
Current 340 Amps. (measured with 50 mv/500 Ampere shunt)
Measured at the top and bottom of the crate segment connector, all values are in millivolts.

Slot #	0	3	5	7	11	15	17	19	20	21	22	23	24	25
Top	31	31	29	29	28	27		27				28		29
Bot.	25	24	18	17	15	14		15				20		24

-2 Volt bus

Bus to Ground voltage -2.0 V
Current 140 Amps. (measured with 50 mv/500 Ampere shunt)
Measured at the top and bottom of the crate segment connector, all values are in millivolts.

Slot #	0	3	5	7	11	15	17	19	20	21	22	23	24	25
Top	24		23	23	22	22		22			23			23
Bot.	17	15	13	12	12	11		11			12			14

The maximum voltage difference across the backplane is specified as 30 mv when fully loaded. The maximum measured was 17 mv.

4. Signal Layers

Signals for each layer are isolated due to the power planes inserted in the multilayer card. This should reduce crosstalk for data lines and prevent the clock from being coupled to other signals. Each group of signals is discussed separately below. All terminations and drivers will be on daughter cards except the 26 MHZ clock bus, Reset bus, Sync bus, Sync Err bus and Event Address Data and Strobe lines which are terminated on the backplane. Terminations are single except as noted.

All pinouts are specified in Appendix A. Slot 0 (rightmost slot viewed from front of crate) has no connections in the Auxiliary backplane. A Netlist can be supplied in any of several computer formats.

5. Electrical

5.1. Specification

All signals conform to ECL standard levels and terminations. **Impedance of transmission lines shall be specified when fully loaded by connectors and a one inch stub with a single ECL receiver on each daughter card which receives that signal.** Termination resistance not less than 25 Ohms for double terminated bus lines (10 Event Address, 2 Sync, 1 Reset) and 50 Ohms for single terminated lines. Impedance not exceeding 100 Ohms. Variation of impedance shall not be greater than 10%.

5.2. Test Procedure

Impedance will be measured using a Time Domain Reflectometer (TDR) with a resolution of at least 1 inch. The impedance will be measured without terminations in place and the TDR connected in place of the drivers. Measurements will be taken for both the loaded and unloaded condition. Loads will be simulated by attaching capacitors of appropriate size to the backplane pins. Figure 4 is a diagram of the test setup. Table 1 shows the test results.

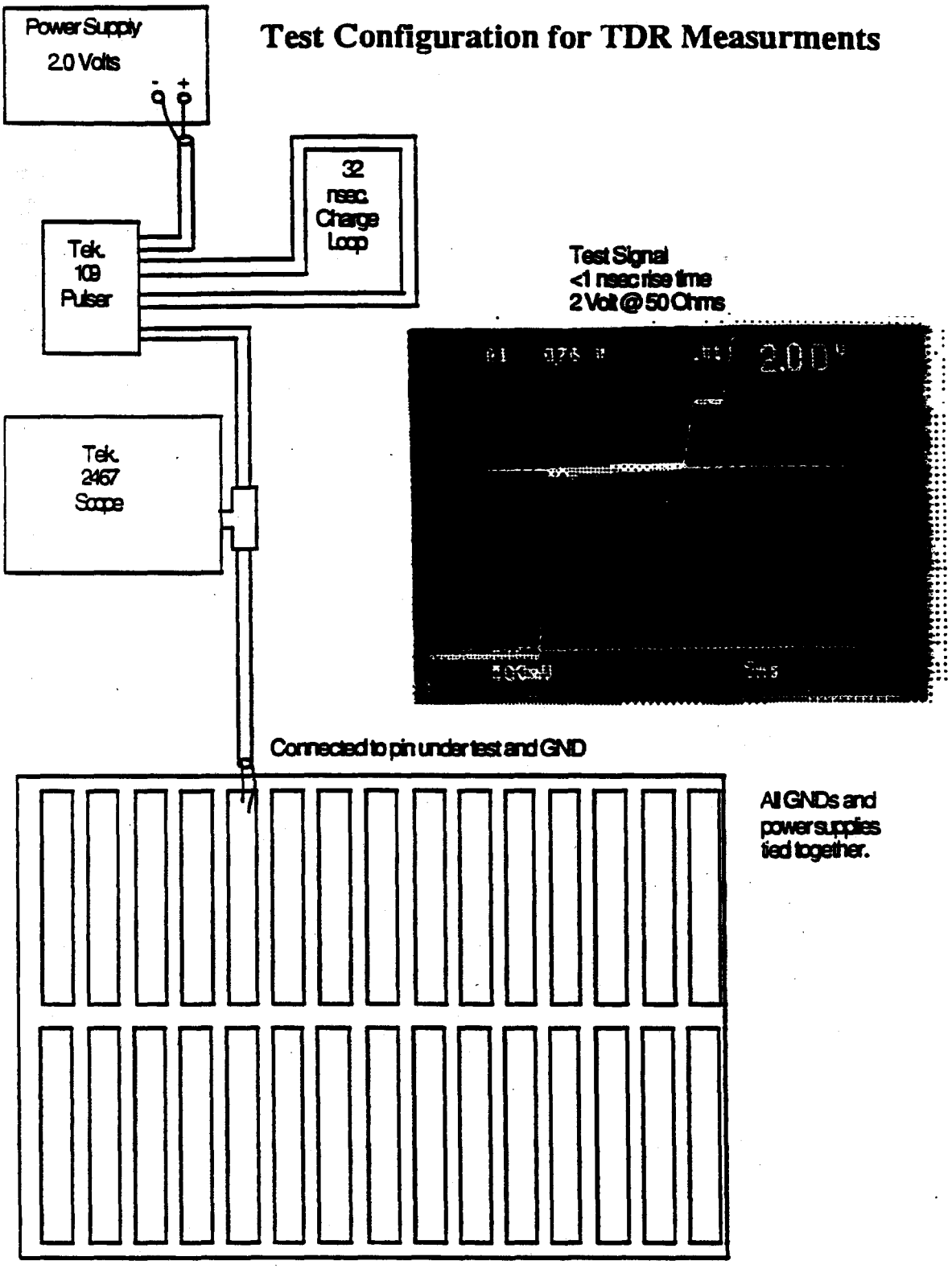

Test Configuration for TDR Measurments
Power Supply
2.0 Volts
- +
32
nsec.
Charge
Loop
Tek.
109
Pulser
Tek.
2467
Scope
Test Signal
<1 nsec rise time
2 Volt @ 50 Ohms
2.00
Connected to pin under test and GND
All GNDs and
power supplies
tied together.

Figure 4

TDR Measurements (Unloaded Lines)

2/14/1990 #550

	ρ	Zo	Zo'	Ei	Er	% Error	Calc. Imped	Meas. Delay	Calc. Delay
	(Er-Ei)/Ei	Ohms	Ohms	Volts	Volts		Ohms	Nanosecs (Delay*2)	Nanosecs (Loaded)
	75 Ohm line measurement for calibration								
	0.23	50	80	2.00	2.46	6.5	75.0	-	-
	Ideal 75 Ohm Measurements								
Note 1	0.20	50	75	2.00	2.40	0.0	75.0	-	-
Note 2		50	75	2.00	2.40				
	53 Mhz Clock Lines								
Note 1	0.38	50	110	2.00	2.75	25.4	87.7	4.4	1.5
Note 2		50	110	2.00	2.75				
Note 4	0.31	50	94	2.00	2.61	7.1	87.7	4.4	1.5
Note 3	**Reset Line(1-B8)**								
Note 1	0.00	50	50	2.00	2.00	-51.5	103.0	10.9	4.0
Note 2		50	50	2.00	2.00				
	Hit Data 0(13-C3)								
Note 1	0.21	50	76	2.00	2.41	17.7	64.4	4.4	1.8
	Hit Data 0(13-A3)								
Note 1	0.21	50	76	2.00	2.41	17.7	64.4	4.4	1.8
Note 3	**26 Mhz Clock(13-C33)**								
Note 1	0.12	50	63	2.00	2.23	-38.8	103.0	5.4	2.3
Note 3	**26 Mhz Clock(13-A33)**								
Note 1	0.12	50	63	2.00	2.23	-38.8	103.0	5.4	2.3
Note 3	**Sync(1-C9)**								
Note 1	0.09	50	59	2.00	2.17	-42.4	103.0	9.0	4.6
Note 3	**Sync Error(1-C10)**								
Note 1	0.09	50	59	2.00	2.17	-42.4	103.0	9.0	4.6
Note 3	**Data Valid(13-C11)**								
Note 1	0.24	50	81	2.00	2.47	25.3	64.4		4.6
Note 3	**AD Lines(25-A22)**								
Note 1	0.00	50	50	2.00	2.00	-22.4	64.4		4.6

Note 1 1st Formula: Zo'=(1+((Er-Ei)/Ei))/(1-((Er-Ei)/Ei))*Zo

Note 2 2nd Formula: Zo'=((Zo*Er)+(Zo*Ei))/(Ei-Er)

Note 3 These lines are Partialy Loaded as they are bussed and have connectors installed.

Note 4 Crate #553

Table 1

The Measured impedance of lines in the backplane is consistently high. The calculations are based on the manufacturers layup and specification of .010 inch trace width, the dielectric constant is assumed to be 4.7 based on industry practice for fiber-glass laminate material.

The parameters most likely to case the observed discrepancy are trace width and dielectric constant. A 50% reduction in trace width to .005 inch would cause the observed error as would a dielectric constant of 3.

To test the trace width the 53 Mhz clock lines were calculated to be .176 Ohm for the given copper thickness and width. A clock line was measured to be .4 Ohm. This is consistent with the error we have observed. See Table 4 for data.

The capacitance of a 53 Mhz clock line was measured to be 20 pf for 7.95 inches of line, or 30 pf/foot. The geometries given by the manufacturer should yield a capacitance of approximately 27 pf/foot (from table 3-10 on page 47 of the MECL System design Handbook) and a .005 inch line width would be 25 pf/foot. The difference is within the measurement and construction tolerance.

An attempt was made to measure the dielectric constant by using the identical layers 2 and 3 which have matching areas and are solid copper.

Dielectric Constant Measurement

Formula: Er = (C x 10**12 x t) / 8.85 A

Er is the relative Dielectric Constant

C is the capacitance in Farads

t is the dielectric thickness in meters

A is the area of one plate in square meters

	Er	C	Layers 2 & 3 Mult.	t	Constant	A
Crate #550	5.73529473	36900.E-12	1.00E+12	.20320E-03	8.85	0.147724
Crate #553	5.81300875	37400.E-12	1.00E+12	.20320E-03	8.85	0.147724

5.3. Segment Backplane

5.3.1. Connectivity

5.3.1.1. Specification

Connectivity as specified in IEEE 960, Chapter 14, no special modifications.

5.3.1.2. Test Procedure

The Computing Division, Instrument Repair Group will test connectivity with a special FASTBUS crate test stand. This procedure is documented in PN 383, "Diagnostic Test for FASTBUS Crate Backplanes", which is obtainable from the Instrument Repair Group of the Fermilab Computing Department.

5.3.1.3. Test Result

The measurements taken by Instrument Repair were checked by hand and all connections were found to be correct. There was some misunderstanding about the power supplies which caused the results shown in Table 3 and Table 4. These tables are from the Instrument Repair test facility.

Summary fo rate 1 Ø553 09-MAR-89 Fastbus crat est V1.0

Slot Written	Slot Read	Pin Written	Pin Open	Pin Shorted
2	1	B 63	B 14	
2	1	B 63	B 15	
2	1	B 63	B 22	
2	1	B 63	B 52	
2	1	B 63	B 63	
2	1	B 63	B 64	
10	9	B 63	B 14	
10	9	B 63	B 15	
10	9	B 63	B 22	
10	9	B 63	B 52	
10	9	B 63	B 63	
10	9	B 63	B 64	
10	11	B 63	B 14	
10	11	B 63	B 15	
10	11	B 63	B 22	
10	11	B 63	B 52	
10	11	B 63	B 63	
10	11	B 63	B 64	
12	13	B 4	B 4	
12	13	B 5	B 5	
12	13	B 6	B 6	

Total errors detected = 21

Table 3

Summary for Crate # Ø550 09-MAR-89 Fastbus Crate Test V1.0

Slot Written	Slot Read	Pin Written	Pin Open	Pin Shorted
0	1	B 38	B 38	
2	1	B 63	B 14	
2	1	B 63	B 15	
2	1	B 63	B 22	
2	1	B 63	B 52	
2	1	B 63	B 63	
2	1	B 63	B 64	
12	13	B 4	B 4	
12	13	B 5	B 5	
12	13	B 6	B 6	
24	25	B 63	B 14	
24	25	B 63	B 15	
24	25	B 63	B 22	
24	25	B 63	B 52	
24	25	B 63	B 63	
24	25	B 63	B 64	

Total errors detected = 15

Brian Anderson

Table 4

5.3.2. Impedance

5.3.2.1. Specification

Impedance is as specified in IEEE 960, Chapter 14, there are no special modifications.

5.3.2.2. Test Procedure

Impedance of the Segment backplane will be tested as described in section 5.2 with the exception that actual modules will be installed to provide a load.

5.3.2.3. Test Result

The Impedance of the Segment backplane was measured to be 50 Ohms with the connector pins installed but not loaded with modules. See Table 1.

5.4. Auxiliary Backplane

5.4.1. 53 MHZ Clock

5.4.1.1. Specification

The 53 MHZ clock has the special requirement that it must be distributed with a very small skew to the Post Amp/Comp. modules and to the Encoder modules. The specification given is +/- 250 picoseconds. It is possible to match the distribution line length for 24 modules (Post Amp & Encoder) but receivers and drivers must be kept to a minimum as the typical skew of ECL 10KH is 800 picoseconds min. to max. per IC. There are two 53 MHZ clocks labeled phase 1 and phase 2. The phase 1 clock goes to the Post Amp/Comp. boards and the phase 2 clock to the Encoder boards. A MC10E111 clock driver is used. as shown in the circuit of Figure 5, the skew for this part is specified as 100 picoseconds min. to max.

The Skew is then 100 picoseconds for the MC10E111 plus the skew variation of the backplane. The backplane lines must be matched to 150 picoseconds delay. As the driver and loads affect the skew, all lines on daughter cards must be less than 1 inch and only one load per Post Amp or Encoder.

The clock is driven from the Sequence in slot 13 of the FASTBUS crate. The top two pins and the bottom two pins on the outside rows of the three row connector will be used to drive the series terminations which are located as close as possible to the connector. All striplines are made as close as possible to the same propagation delay.

The stripline impedance is calculated from the equation on page 42 of the MECL System Design Handbook as follows;

Zo (Stripline)	Er (Dielectric)	b (Thickness)	t (Copper)	w (Strip Width)
85 Ohms	5	100 mills	0.003 mills	10 mills

A correction must be made as shown by the equation on page 152 to allow for loading. This is the line capacitance plus connector and gate capacitance. A connector load of 7 picofarads (2.2 for pins, 2 for a stub and 2.8 connectors) and a gate load of 3 picofarads (page 141) was assumed. The line capacitance is from the equation on page 142.

Zo' (Stripline)	CT	Co	Cg	Ccon
51 Ohms	10pf	2.2 pf/inch	3pf	7pf

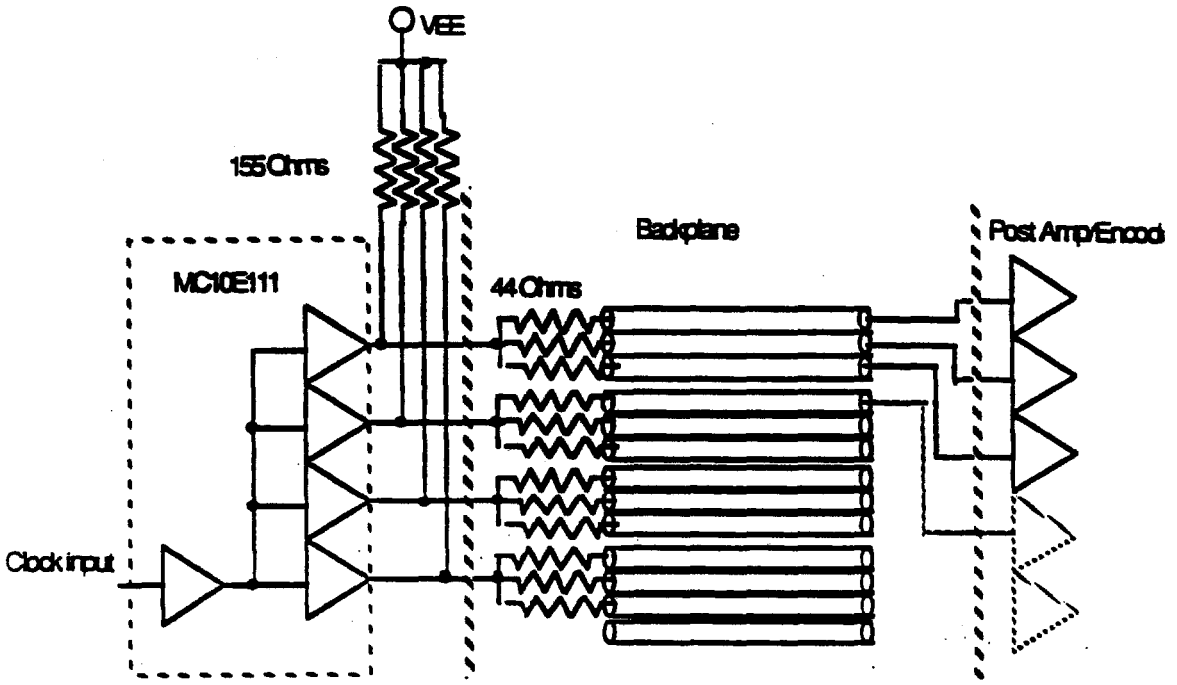


Figure 5

The series resistor value is given by subtracting the nominal output impedance for MECL 10K of 7 Ohms (page 46) from Zo', thus Rs = 44 Ohms.
The value of the emitter pulldown resistor is from the formula on page 48. This yields the maximum pulldown resistor which may be used for a given fanout and Zo'. The minimum value is determined by the maximum output current. The maximum current for ECL 10K is 50 ma and the recommended value is 25 ma.

RE(Max)	Zo'	Rs	n (fanout)	Isource
155	51 Ohms	44 Ohms	3	28ma

The clock will be received by the Post Amp module on the top left pin (C01) and the fourth from bottom left (C62) on the Encoder modules (viewed from the front of the crate).

5.4.1.2. Connectivity

Connectivity is checked by comparing Ohm meter readings of appropriate pins against schematic above and pin-out list. All signals are correctly connected.

5.4.1.3. Impedance

Impedance of the 53 Mhz Clock lines were tested as described in section 5.2. See Table 1 for unloaded Impedance. Resistence measurements were made to try to determine the cause of the impedance error. The resistence was measured using a known current through the line under test and measuring the voltage drop. Ohms law was then used to calculate the resistence. A check was done by using a sensitive Ohmmeter, the results agreed with the current method. See Table 4 for resistence results and section 5.2 and Table 1 for impedance tests.

R of 53 MHZ Lines

A	V	R	Crate #550	
1.200	.55	.458		Sh25-C1
1.204	.55	.457		23-C1
1.195	.55	.460		21-C1
1.215	.55	.453		17-C1
1.205	.55	.456		17-C1
1.182	.55	.465		15-C1
1.145	.56	.489		12-C1
1.201	.55	.458		10-C1
1.160	.55	.466		8-C1
1.161	.56	.482		6-C1
1.190	.55	.462		4-C1
1.190	.55	.462		2-C1
1.114	.55	.494		1 C62
1.127	.55	.447		3 C62
1.230	.53	.430	Crate # 553	24-C62
1.250	.52	.416		20-C62
1.188	.51	.432		1 C62
1.181	.51	.432		3 C62

Table 4

5.4.2. 26 MHZ Clock

5.4.2.1. Specification

The 26 MHZ clock is bussed to all Encoder slots (12) and is driven from the center by the sequencer card with two MC10123 bus drivers. The drivers will be on the Sequencer, the bus is terminated at each end of the backplane with a resistor to -2.0 Volts. These are driven independently and are single terminated.

The left (C33) and right (A33) connector pins are used to drive the clock busses. The Encoder modules receive the clock on pin C49 (crate viewed from the front) of the Auxiliary connector.

All drivers and receivers are on modules and lead lengths must be less than 1 inch.

5.4.2.2. Connectivity

Connectivity is checked by comparing Ohm meter readings of appropriate pins against pin-out list. Connectivity is correct.

5.4.2.3. Impedance

Impedance of the Segment backplane will be tested as described in section 5.2 with the exception that actual modules will be installed to provide a load. See Table 1 for unloaded impedance.

5.4.3. Post Amp Data

5.4.3.1. Specification

There are 128 data channels transmitted from the Post Amp to the Encoder module. The modules are always adjacent and line lengths including connectors must be less than 2 inches.

As the 128 channels will require two rows of pins (65 pins/row), the adjacent rows of the connectors for the Post Amp and Encoder modules are used for I/O. This leaves one row of Encoder module pins for communication with the Sequencer module. The Auxiliary card signals are picked from the same pins used for signals transmitted to the encoder. This means one Auxiliary card will pick off row B on the Encoder, which is connected to row A on the Post Amp and the other will Pick up row B directly from the Post Amp module. See Figure 6.

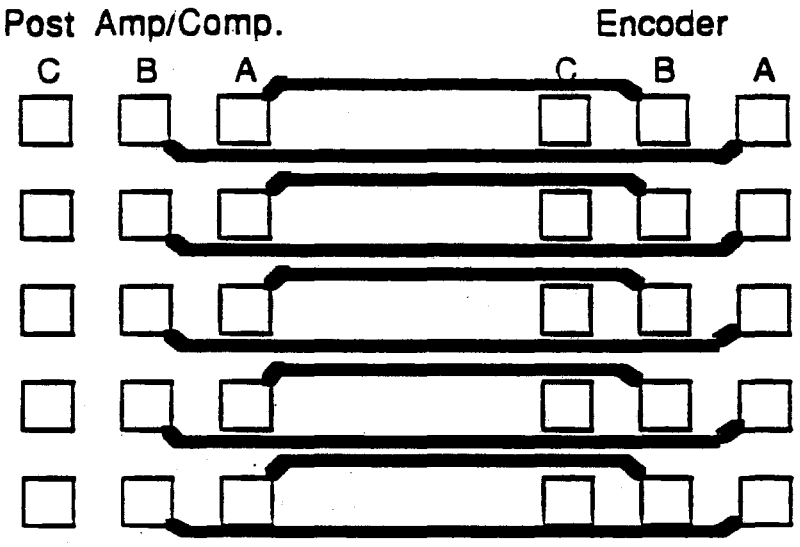


Figure 6

There are 16 analog and 16 digital sums which must be transmitted by the Auxiliary cards to the trigger logic. These are connected using a similar scheme which allows the Auxiliary cards to be identical for Post Amp and Encoder slots. See Figure 7.

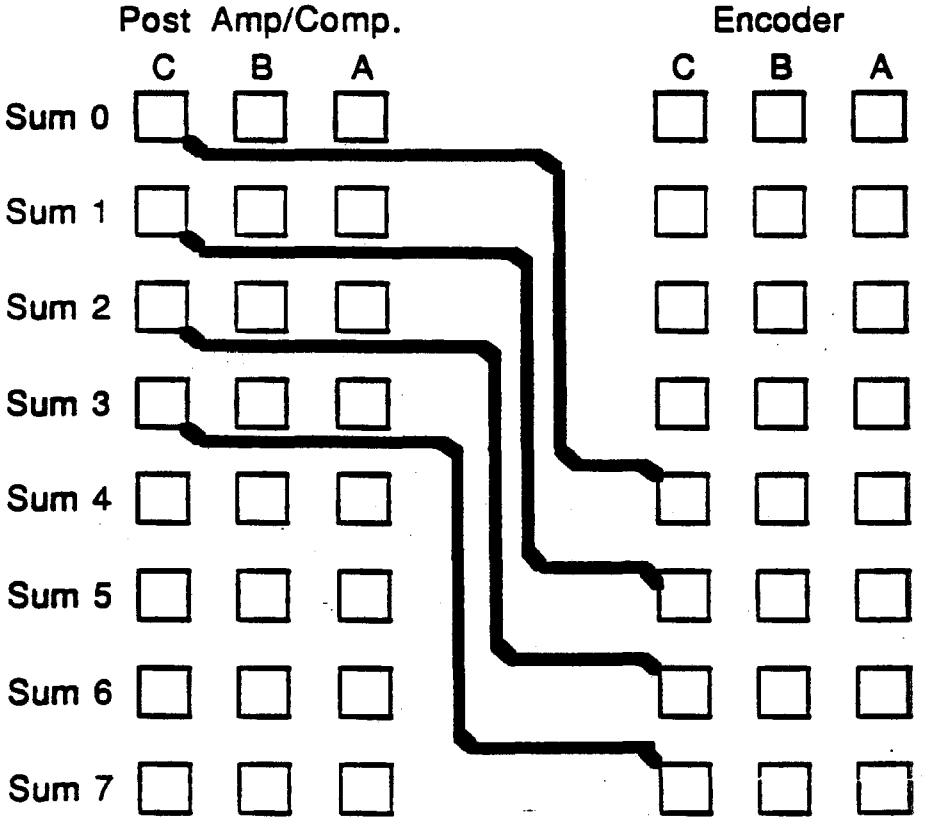


Figure 7

5.4.3.2. Connectivity

Connectivity is checked by comparing Ohm meter readings of appropriate pins against schematic above and pin-out list. Connectivity is as specified.

5.4.3.3. Impedance

Impedance of the Segment backplane will be tested as described in section 5.2. See Table 1 for unloaded impedance.

5.4.4. Hit Data

5.4.4.1. Specification

There are eight Hit Data lines and one Data Valid for each Encoder module. These are connected from row C pins not used by the Post Amp data lines, to the Sequencer card.

5.4.4.2. Connectivity

Connectivity is checked by comparing Ohm meter readings of appropriate pins against schematic above and pin-out list. All signals are correctly connected.

5.4.4.3. Impedance

Impedance of the Segment backplane will be tested as described in section 5.2. See Table 1 for unloaded impedance.

5.4.5. Event Address

5.4.5.1. Specification

The Event Address lines are bussed in the outer layers of the board. There are eight data bits, Event Address Enable and one Event Address Strobe line. Because these lines are bussed to the 12 Encoder slots, the effect of capacitive loading on impedance must be considered as the load is distributed over eight inches of backplane. These lines are terminated at each end of the backplane with a resistor to -2.0 Volts. They are driven from the Sequencer module and terminated at the last backplane connectors. The lines on the Encoder cards are stubs off this bus and must be kept to less than one inch.

5.4.5.2. Connectivity

Connectivity is checked by comparing Ohm meter readings of appropriate pins against schematic above and pin-out list. Connectivity is as specified.

5.4.5.3. Impedance

Impedance of the Segment backplane will be tested as described in section 5.2. See Table 1 for unloaded impedance.

5.4.6. Reset and Sync

5.4.6.1. Specification

The Reset is bussed to all slots except slot 0. The two Sync lines are bussed to all Encoder card slots. These lines are driven from the Sequencer module pins B01, B10 and B09, and are terminated at each end of the backplane with a resistor to -2.0 Volts. The Reset signals are connected to all Post Amp and Encoder cards. The Traces on the daughter cards are stubs off this bus and must be kept to less than one inch.

5.4.6.2. Connectivity

Connectivity is checked by comparing Ohm meter readings of appropriate pins against schematic above and pin-out list. Connectivity is correct.

5.4.6.3. Impedance

Impedance of the Segment backplane will be tested as described in section 5.2. See Table 1 for unloaded impedance.

Fermi National Accelerator Laboratory

MODULE AND CRATE SUBSYSTEM TEST HARDWARE CONFIGURATIONS

HN 102

H. Gonzalez, W. Kowald, D. Slimmer, C. Swoboda

June 10, 1991
Rev 1

Table of Contents

1. General Description

The FASTBUS modules that are used in the Silicon Strip Detector (SSD) Readout System are listed below:

MAIN SYSTEM MODULES
- Postamp/Comparator (PC)
- Delay/Encoder (DE)
- Sequencer (SEQ)
- FASTBUS Smart Crate Controller (FSCC)
- Main Timing Controller (MTC)

DIAGNOSTIC MODULES
- Test Stand Module (TSM)
- Level Shifter Module(LSM)

Each of the modules listed above must be tested before being integrated into crate subsystems as shown in Figure 1. This document details the hardware configurations required to test both individual PC and DE modules and modules assembled into crate subsystems. Detailed diagnostic test information for the FSCC, MTC, and SEQ can be found in the individual hardware and software description manuals for each module. The hardware and software description documents are listed in Table I.

MODULE	HARDWARE DESC. #	SOFTWARE DESC. #	
		SINGLE BOARD	SYSTEM
FSCC	HN 96	PN 417	--
MTC	HN 98	PN 434	PN 436
SEQ	HN 99	PN 434	PN 436
PC	HN 100	PN 434	PN 436
DE	HN 101	PN 434	PN 436
TEST HARDWARE	HN 102	PN 434	PN 436

Table I: Documentation

In addition this document covers the current and appropriate fusing for each module, full crate power and difference between a SSD crate and a FASTBUS crate.

1.1 Application

The FSCC, MTC, and SEQ diagnostic tests are performed using a minimum of other module types. For example, the FSCC is tested in a stand alone mode while the MTC and SEQ need the FSCC acting as a FASTBUS master to be fully tested. The individual hardware description documents for the FSCC, MTC, and SEQ contain the hardware test configurations for those particular modules. **This document focuses on the hardware configurations required to test the PC, DE modules and the crate subsystem tests.** To simplify the description of the different hardware configurations, a series of module symbols is provided in Figure 2. These symbols are not a detailed representation of the modules front panel.The two diagnostic modules, TSM and

LSM, are used in conjunction with the FSCC, SEQ, and MTC to perform all the PC - DE tests.

All the diagnostic software used in testing the SSD readout modules is contained in EPROM that are installed in the FSCC module. These tests are stand alone requiring only a terminal. There are actually two EPROM sets (four EPROMs each set), one for system tests (SSD SYST DIAG), and one for individual module tests (SSD MOD DIAG). The user will have to swap EPROM sets (using bank 2) when changing from system to single board test unless two FSCC are available. **Please see Software Document PN 434 and PN 436 for complete software details.**

1.1.1. Packaging

The SSD readout system is package in a FASTBUS crate, but several modifications where made to accommodate the system requirements. The following are the special changes:

- The backplane is an integral unit that includes the FASTBUS segment as well as the auxiliary segment,
- The auxiliary connector of all slots are specified for readout purposes.
- The power pins +15v, -15v and 28v are allocated for +3.5v and -3.5 as required by the PC.
- A key bar is used to prevent non SSD readout modules from being plugged into the crate.

These modifications are documented in the SSD backplane specifications.

1.1.2. Physical Size

The main modules are all FASTBUS size boards. Auxiliary backplane sized modules used in individual module tests include the following:

FSCC	-	Output Port Auxiliary Backplane test module.
SEQ	-	Output Port Auxiliary Backplane test module.
DE Aux.	-	Auxiliary card for DE slot. Used for DE testing.
TSM Aux.	-	Auxiliary card for TSM slot. Used for DE testing.
SEQ Aux.	-	Auxiliary card for SEQ. Used for DE testing.

1.2 Auxiliary Backplane Module Position Pinout

The pinout of the FSCC, SEQ, PC, DE, TSM and LSM are provided in appendix A. The following is a module-slot allocation list.

SEQ	-	Slot 13.
FSCC	-	Slot 0.
DE	-	Slots 24, 22, 20, 18, 16, 14, 12, 10, 8, 6, 4, 2.

PC - Slots 25, 23, 21, 19, 17, 15, 11, 9, 7, 5, 3, 1.

1.3 Power Requirements

	+5v	-5.2v	-2v	+3.5v	-3.5v	WATTS
PC	2.74A	3.8A	4.5A	4.62A	5.63A	78.1W
DE	0.02A	14.0A	6.0A			84.9W
SEQ	9.0A	3.0A	2.0A			65.0W
FSCC	5.4A	1.2A	0.2A			33.4W
MTC	1.0A	6.0A	1.5A			58.6W
TSM	2.5A	9.0A	4.0A			67.3W
LSM	0.007A	6.5A	2.2A			38.0W

1.3.1. Module Fusing

Each FASTBUS module is both fuse and Transzorb protected. The table below shows the fusing level for each module with fuses in parallel shown in the following format; NUMBER IN PARALLEL@(FUSE VALUE).

	+5v	-5.2v	-2v	+3.5v	-3.5v
PC	1@(10A)	1@(10A)	1@(10A)	1@(10A)	1@(10A)
DE	1@(.5A)	2@(10A)	1@(10A)		
SEQ	3@(4A)	3@(2A)	1@(5A)		
FSCC	1@(10A)	1@(10A)	1@(5A)		
MTC	1@(5A)	1@(10A)	1@(5A)		
TSM	1@(4A)	2@(10A)	1@(10A)		
LSM	1@(.5A)	1@(10A)	1@(3A)		

Full Crate Power Requirements

	+5.0v	-5.2v	-2.0v	+3.5v	-3.5v
PC(12)	32.88A	45.12A	53.16A	55.44A	67.56A
DE(12)	0.24A	168.0A	72.0A		
SEQ(1)	9.0A	3.0A	2.0A		
FSCC(1)	5.4A	1.2A	0.2A		
TOTAL	47.52A	217.32A	127.36A	55.4A	67.6A

The power dissipated in the crate using the nominal voltages and currents is 2053 watts.

2. General Information

The test configurations described in this document can be divided into tests that can be performed on a FASTBUS crate (i.e. FASTBUS interface tests) and those that must be run on a SSD crate (i.e. SSD readout tests). Since the tests performed on a FASTBUS crate can also be performed on the SSD crate it will be assumed for this document that the user is always using a SSD crate. For the SSD crate the keybar will ensure that each module reside in the proper slot. Note that for the hardware configurations described in this document the only module that does not have a prespecified slot is the MTC. Since the MTC does not use the auxiliary connector it will be keyed to use any of the slots.

There are several parameters or options that have to be properly initialized. Among these there are two that require special attention; the CLK1 to CLK2 delay and the address correction parameter. The CLK1 to CLK2 delay is set by switches in the SEQ on a read only FASTBUS register. For all test using the SEQ the switches must be set to a value of 28. The address correction parameter is used to account for the difference in start time and propagation delays between different modules. This parameter essentially aligns the source and the destination. For example, when using the TSM/LS to drive the PC, data from address 0 of the TSM will not be stored in address 0 of the DE. The difference is provided to the software by the address correction parameter.

2.1 Test Hardware

The SSD readout system is based on four modules; DE, FSCC, SEQ and PC. In this section the special hardware used to test DE and PC is described. The DE test is performed with the TSM. For the PC test the TSM and LSM are used as a unit. The abbreviation TSM/LSM is used when referring to both modules.

The TSM/LSM is composed of two FASTBUS modules as shown in the block diagrams of Figure 3a and 3b. The TSM and LSM are keyed like the PC and DE, respectively. A detailed hardware description of these modules is provided in TSM/LSM Hardware Description. The TSM is composed of memory and control logic that allows the generation of 128 parallel bits of simulated hit data at the 53MHz accelerator clock rate. A different pattern can be generated every 18.9 ns. These 128 bits patterns can be used to drive the DE or to drive the LSM. The TSM implements a mode switch that selects the clock source for it's internal operation. Each of these modes is specified in such a manner that the required synchronization is provided. There are four modes (modes 0 to 3) and in this document a mode is specified for each of the hardware configurations.

The LSM module is used to convert single ended ECL data to differential with the option of selecting four possible amplitudes. These amplitudes are selected to be in the operating range of the PC inputs.

3. Single Board Test

In this section the hardware configurations required to exercise the software covered on software document PN 434 are described.

3.1 DE Test

The DE does not have a FASTBUS interface. The tested function of the module is therefore it's ability to input hit data, encode the 128 channel hit pattern into encoded hit addresses and output that data to a SEQ channel emulator or a SEQ module.

The hardware configuration in Figure 4 is used when a SEQ module is not available and testing of the DE is required. To run this test a working FSCC, SEQ and Auxiliary cards are required. The TSM is used to simulate the PC and a single input channel of the SEQ. The TSM resides in a PC module slot with the DE under test in the normal adjacent slot. The SEQ port connection is made by the auxiliary card and the backplane provides connection for the 128 data channels (this is the normal PC data connection). For this test a termination card is needed at the SEQ slot to replace the terminations provided by the SEQ. The address correction parameter for this test is set to 0.

To tests the DE the TSM is set for mode 0 or mode 1 operation, but for mode 1 a NIM clock (normally 55MHz) must be supplied for the TSM. Acceptance criteria for the DE requires that the test be run with the 55MHz clock.

Once the hardware has been properly set the user can start testing DEs. The software is menu driven with pattern data selection, looping options, stoping conditions, etc. When the test is started the appropriate bits are set on the TSM such that data transfer to the DE begins. After a delay the DE will start writing data synchronously into it's memory. If the DE and the TSM are properly synchronized the error light of the DE should be off. Synchronization problems must be fixed before proceeding with the test.

In general testing of the DE is a based on writing an event address and forcing a trigger through the TSM. After recognizing the trigger the DE encodes the event and transmits it to the TSM (SEQ channel of the TSM). The FSCC reads the event and compares it with the expected event. If errors occur they are reported to the user in a table that displays the expected data with the received data. The diagnostic software required for this test is covered in detail in document PN434.

3.2 PC Test

3.2.1. PC FASTBUS Diagnostics

When the PC diagnostics are selected from the SSD module test menu, a new menu will be displayed that includes separate detailed tests for each function of the module. To run this test a working FSCC is required. In addition there is a menu option which automatically tests all the module functions without operator intervention. If the module passes these tests without an error, then by definition the FASTBUS interface is operational. Refer to software document PN 434 for more details. The features of the module that are tested are shown below:

Geographical Addressing
Primary Address
Secondary Addressing
NTA Register
CSR00
CSR01
CSR10

Test Counter
Channel Enable/Disable
Halt Mode
Run Mode
Module I.D.
Digital-to-Analog Converters and Read back
Normal and sum channel analog inputs.

4. System Tests

The system software implements different features to perform incremental SSD readout tests. This section describes the hardware configurations that are associated with system tests covered in software document PN 436. The tests are grouped under the system tests category because they require multiple modules of the SSD readout system to be used. The are three tests covered: TSM as PC, TSM/LSM driving the input port of the PC and PC counter test.

4.1 TSM Based Test

This configuration adds the normal DE readout path to the DE tests described in section 3.1. The test setup for this configuration is shown in Figure 5. In this test the TSM replaces a PC and drives the associated DE with the data patterns specified by the user.

For this test the TSM is set for mode 2. In this mode the TSM uses CLK1 from the backplane to synchronize to the DE and it will start transmitting data when CLK1 is enable on the backplane. To check that the TSM is synchronized with the rest of the readout system the user can check the TSM front panel TC* against the DE front panel TC*. A sketch of these two signals for properly synchronized modules is shown in Figure 5. Note that DE synchronization errors will be flagged by the error led of the DE. For this test the address offset parameter is set to 0.

The encoding of an event is requested from the MTC and the encoded event is readout from the SEQ. The read back data is compared to the transmitted data and errors are reported to the user. The diagnostic software required for this test is covered in detail in document PN436.

4.2 PC Input Port Test

To test the PC input port the TSM/LSM are used with the hardware configuration shown in Figure 6. To run this test a working FSCC, SEQ, TSM/LSM and MTC are required. Note that the TSM and LSM occupy a PC and DE slot, respectively. For this test it is required that the TSM/LSM, DE, SEQ, FSCC and MTC be operational before the test is started. The TSM is set for mode 3 operation and the LSM outputs are connected to the PC inputs. In addition the TSM front panel 53MHz and SYNC are supplied by the MTC. The TSM uses the front panel clock and the sync signal to synchronize with the rest of the readout system. If the TSM has been properly initialized it will be synchronized with the rest of the SSD readout system. Note that DE synchronization is flagged by an off error LED, but for the TSM there is no such LED. TSM synchronization can be checked by referencing the front panel TC* of the DE with that of the TSM. For the cable length specified on figure 6 and for a CLK1 delay value of 39 a sketch of these two signal is

provided on the same figure. Note that the timing between these two signals changes with the CLK1 delay.

Once the software has initialized everything pattern data is generated by the TSM, converted by the LSM to the proper amplitudes, the PC discriminates and latches the input data and drives the DE. For the cable length provided the address correction parameter should be initialized to 0 and the CLK1 delay value to 39.

The encoding of an event is requested by the FSCC through the MTC and the encoded event is transmitted by the DE to the SEQ. The encoded data is read by the FSCC and compared to the test pattern. Since TSM/LSM, DE, SEQ and MTC are known to be operational, errors found are attributed to the PC under test.

4.3 PC Test Counter Test

The PC counter test is used to check the SSD readout system and to decide when a group of PCs, DEs and SEQ is functional. The test uses the internal counter of the PC to generate known data patterns. The PC is intialized such that the counter data bypasses the analog input data circuitry. This test is essentially a module integration test and is intended to be run after all the individual module tests have been successfully run and the modules are integrated into crate sub-systems. For this test the address correction parameter is set to 1.

The test is basically a module substitution tests and therefore requires that all the modules shown in Figure 7 operate correctly before tests of either the PC or the DE are attempted. The test requires that the user define the addresses of the PC to be active in the test. Note that for each PC address defined the associated DE must be present, otherwise errors will occur because encoded data is expected from a defined PC. Note that synchronization problems with the DE are reported to the MTC, but there is no report when a PC counter is out of synchronization. Because of the peculiar data patterns generated by the counter it is possible to attribute certain type of errors to a PC counter out of synchronization.

In general the test is run with a maximum of 11 DE-PC pairs which allows the MTC to reside in the same SSD crate. This minimizes communication times and increases the event testing rate. The basic test uses rolling counter with all trigger address checking (refer to software document 436) . The acceptance criteria for the crate subsystem test requires that this test run for a million events without error. Note that this criteria is for the crate subsystem and all the module used in this test must satisfy the single module test criteria.

5. Appendix A: Module Auxiliary Backplane Pinouts

Encoder Module Auxiliary Connector Pin List

(Viewed From Front of Crate-10/1/90)

C01-N/C	B01-Post/Disc Ch.00	A01-Post/Disc Ch.01
C02-GND	B02-Post/Disc Ch.02	A02-Post/Disc Ch.03
C03-N/C	B03-Post/Disc Ch.04	A03-Post/Disc Ch.05
C04-GND	B04-Post/Disc Ch.06	A04-Post/Disc Ch.07
C05-GND	B05-Post/Disc Ch.08	A05-Post/Disc Ch.09
C06-GND	B06-Post/Disc Ch.10	A06-Post/Disc Ch.11
C07-GND	B07-Post/Disc Ch.12	A07-Post/Disc Ch.13
C08-Reset	B08-Post/Disc Ch.14	A08-Post/Disc Ch.15
C09-Sync	B09-Post/Disc Ch.16	A09-Post/Disc Ch.17
C10-GND	B10-Post/Disc Ch.18	A10-Post/Disc Ch.19
C11-GND	B11-Post/Disc Ch.20	A11-Post/Disc Ch.21
C12-Sync Err	B12-Post/Disc Ch.22	A12-Post/Disc Ch.23
C13-GND	B13-Post/Disc Ch.24	A13-Post/Disc Ch.25
C14-GND	B14-Post/Disc Ch.26	A14-Post/Disc Ch.27
C15-GND	B15-Post/Disc Ch.28	A15-Post/Disc Ch.29
C16-GND	B16-Post/Disc Ch.30	A16-Post/Disc Ch.31
C17-GND	B17-Post/Disc Ch.32	A17-Post/Disc Ch.33
C18-GND	B18-Post/Disc Ch.34	A18-Post/Disc Ch.35
C19-GND	B19-Post/Disc Ch.36	A19-Post/Disc Ch.37
C20-GND	B20-Post/Disc Ch.38	A20-Post/Disc Ch.39
C21-Hit Data 0	B21-Post/Disc Ch.40	A21-Post/Disc Ch.41
C22-Hit Data 1	B22-Post/Disc Ch.42	A22-Post/Disc Ch.43
C23-GND	B23-Post/Disc Ch.44	A23-Post/Disc Ch.45
C24-Hit Data 2	B24-Post/Disc Ch.46	A24-Post/Disc Ch.47
C25-Hit Data 3	B25-Post/Disc Ch.48	A25-Post/Disc Ch.49
C26-GND	B26-Post/Disc Ch.50	A26-Post/Disc Ch.51
C27-Hit Data 4	B27-Post/Disc Ch.52	A27-Post/Disc Ch.53
C28-Hit Data 5	B28-Post/Disc Ch.54	A28-Post/Disc Ch.55
C29-GND	B29-Post/Disc Ch.56	A29-Post/Disc Ch.57
C30-Hit Data 6	B30-Post/Disc Ch.58	A30-Post/Disc Ch.59
C31-GND	B31-Post/Disc Ch.60	A31-Post/Disc Ch.61
C32-Hit Data 7	B32-Post/Disc Ch.62	A32-Post/Disc Ch.63
C33-Data VAlid	B33-Post/Disc Ch.64	A33-Post/Disc Ch.65
C34-GND	B34-Post/Disc Ch.66	A34-Post/Disc Ch.67
C35-26 MHZ Clock	B35-Post/Disc Ch.68	A35-Post/Disc Ch.69
C36-GND	B36-Post/Disc Ch.70	A36-Post/Disc Ch.71
C37-Event Address Valid	B37-Post/Disc Ch.72	A37-Post/Disc Ch.73
C38-Event Address Wrt. En.	B38-Post/Disc Ch.74	A38-Post/Disc Ch.75
C39-GND	B39-Post/Disc Ch.76	A39-Post/Disc Ch.77
C40-Event Address 0	B40-Post/Disc Ch.78	A40-Post/Disc Ch.79
C41-Event Address 1	B41-Post/Disc Ch.80	A41-Post/Disc Ch.81
C42-GND	B42-Post/Disc Ch.82	A42-Post/Disc Ch.83
C43-Event Address 2	B43-Post/Disc Ch.84	A43-Post/Disc Ch.85
C44-Event Address 3	B44-Post/Disc Ch.86	A44-Post/Disc Ch.87
C45-GND	B45-Post/Disc Ch.88	A45-Post/Disc Ch.89
C46-Event Address 4	B46-Post/Disc Ch.90	A46-Post/Disc Ch.91
C47-Event Address 5	B47-Post/Disc Ch.92	A47-Post/Disc Ch.93
C48-GND	B48-Post/Disc Ch.94	A48-Post/Disc Ch.95
C49-Event Address 6	B49-Post/Disc Ch.96	A49-Post/Disc Ch.97
C50-Event Address 7	B50-Post/Disc Ch.98	A50-Post/Disc Ch.99
C51-GND	B51-Post/Disc Ch.100	A51-Post/Disc Ch.101
C52-GND	B52-Post/Disc Ch.102	A52-Post/Disc Ch.103

C53-GND	B53-Post/Disc Ch.104	A53-Post/Disc Ch.105
C54-GND	B54-Post/Disc Ch.106	A54-Post/Disc Ch.107
C55-GND	B55-Post/Disc Ch.108	A55-Post/Disc Ch.109
C56-GND	B56-Post/Disc Ch.110	A56-Post/Disc Ch.111
C57-GND	B57-Post/Disc Ch.112	A57-Post/Disc Ch.113
C58-GND	B58-Post/Disc Ch.114	A58-Post/Disc Ch.115
C59-GND	B59-Post/Disc Ch.116	A59-Post/Disc Ch.117
C60-GND	B60-Post/Disc Ch.118	A60-Post/Disc Ch.119
C61-GND	B61-Post/Disc Ch.120	A61-Post/Disc Ch.121
C62-H53MHZ,Ø2 Clock	B62-Post/Disc Ch.122	A62-Post/Disc Ch.123
C63-L53MHZ,Ø2 Clock	B63-Post/Disc Ch.124	A63-Post/Disc Ch.125
C64-N/C	B64-Post/Disc Ch.126	A64-Post/Disc Ch.127
C65-N/C	B65-GND	A65-N/C

PC Module Auxiliary Connector Pin List

(Viewed From Front of Crate-10/1/90)

C01-H53MHZ,Ø1 Clock	B01-Post/Disc Ch.01	A01-Post/Disc Ch.00
C02-GND	B02-Post/Disc Ch.03	A02-Post/Disc Ch.02
C03-L53MHZ,Ø1 Clock	B03-Post/Disc Ch.05	A03-Post/Disc Ch.04
C04-GND	B04-Post/Disc Ch.07	A04-Post/Disc Ch.06
C05-GND	B05-Post/Disc Ch.09	A05-Post/Disc Ch.08
C06-GND	B06-Post/Disc Ch.11	A06-Post/Disc Ch.10
C07-GND	B07-Post/Disc Ch.13	A07-Post/Disc Ch.12
C08-Reset	B08-Post/Disc Ch.15	A08-Post/Disc Ch.14
C09-GND	B09-Post/Disc Ch.17	A09-Post/Disc Ch.16
C10-GND	B10-Post/Disc Ch.19	A10-Post/Disc Ch.18
C11-GND	B11-Post/Disc Ch.21	A11-Post/Disc Ch.20
C12-Analog Sum 0*	B12-Post/Disc Ch.23	A12-Post/Disc Ch.22
C13-Analog Sum 0	B13-Post/Disc Ch.25	A13-Post/Disc Ch.24
C14-Digital Sum 0*	B14-Post/Disc Ch.27	A14-Post/Disc Ch.26
C15-Digital Sum 0	B15-Post/Disc Ch.29	A15-Post/Disc Ch.28
C16-Analog Sum 1*	B16-Post/Disc Ch.31	A16-Post/Disc Ch.30
C17-Analog Sum 1	B17-Post/Disc Ch.33	A17-Post/Disc Ch.32
C18-Digital Sum 1*	B18-Post/Disc Ch.35	A18-Post/Disc Ch.34
C19-Digital Sum 1	B19-Post/Disc Ch.37	A19-Post/Disc Ch.36
C20-GND	B20-Post/Disc Ch.39	A20-Post/Disc Ch.38
C21-Analog Sum 2*	B21-Post/Disc Ch.41	A21-Post/Disc Ch.40
C22-Analog Sum 2	B22-Post/Disc Ch.43	A22-Post/Disc Ch.42
C23-Digital Sum 2*	B23-Post/Disc Ch.45	A23-Post/Disc Ch.44
C24-Digital Sum 2	B24-Post/Disc Ch.47	A24-Post/Disc Ch.46
C25-Analog Sum 3*	B25-Post/Disc Ch.49	A25-Post/Disc Ch.48
C26-Analog Sum 3	B26-Post/Disc Ch.51	A26-Post/Disc Ch.50
C27-Digital Sum 3*	B27-Post/Disc Ch.53	A27-Post/Disc Ch.52
C28-Digital Sum 3	B28-Post/Disc Ch.55	A28-Post/Disc Ch.54
C29-GND	B29-Post/Disc Ch.57	A29-Post/Disc Ch.56
C30-GND	B30-Post/Disc Ch.59	A30-Post/Disc Ch.58
C31-Analog Sum 4*	B31-Post/Disc Ch.61	A31-Post/Disc Ch.60
C32-Analog Sum 4	B32-Post/Disc Ch.63	A32-Post/Disc Ch.62
C33-Digital Sum 4*	B33-Post/Disc Ch.65	A33-Post/Disc Ch.64
C34-Digital Sum 4	B34-Post/Disc Ch.67	A34-Post/Disc Ch.66
C35-Analog Sum 5*	B35-Post/Disc Ch.69	A35-Post/Disc Ch.68
C36-Analog Sum 5	B36-Post/Disc Ch.71	A36-Post/Disc Ch.70
C37-Digital Sum 5*	B37-Post/Disc Ch.73	A37-Post/Disc Ch.72
C38-Digital Sum 5	B38-Post/Disc Ch.75	A38-Post/Disc Ch.74
C39-GND	B39-Post/Disc Ch.77	A39-Post/Disc Ch.76
C40-Analog Sum 6*	B40-Post/Disc Ch.79	A40-Post/Disc Ch.78

C41-Analog Sum 6	B41-Post/Disc Ch.81	A41-Post/Disc Ch.80
C42-Digital Sum 6*	B42-Post/Disc Ch.83	A42-Post/Disc Ch.82
C43-Digital Sum 6	B43-Post/Disc Ch.85	A43-Post/Disc Ch.84
C44-Analog Sum 7*	B44-Post/Disc Ch.87	A44-Post/Disc Ch.86
C45-Analog Sum 7	B45-Post/Disc Ch.89	A45-Post/Disc Ch.88
C46-Digital Sum 7*	B46-Post/Disc Ch.91	A46-Post/Disc Ch.90
C47-Digital Sum 7	B47-Post/Disc Ch.93	A47-Post/Disc Ch.92
C48-GND	B48-Post/Disc Ch.95	A48-Post/Disc Ch.94
C49-GND	B49-Post/Disc Ch.97	A49-Post/Disc Ch.96
C50-GND	B50-Post/Disc Ch.99	A50-Post/Disc Ch.98
C51-GND	B51-Post/Disc Ch.101	A51-Post/Disc Ch.100
C52-GND	B52-Post/Disc Ch.103	A52-Post/Disc Ch.102
C53-GND	B53-Post/Disc Ch.105	A53-Post/Disc Ch.104
C54-GND	B54-Post/Disc Ch.107	A54-Post/Disc Ch.106
C55-GND	B55-Post/Disc Ch.109	A55-Post/Disc Ch.108
C56-GND	B56-Post/Disc Ch.111	A56-Post/Disc Ch.110
C57-GND	B57-Post/Disc Ch.113	A57-Post/Disc Ch.112
C58-GND	B58-Post/Disc Ch.115	A58-Post/Disc Ch.114
C59-GND	B59-Post/Disc Ch.117	A59-Post/Disc Ch.116
C60-GND	B60-Post/Disc Ch.119	A60-Post/Disc Ch.118
C61-GND	B61-Post/Disc Ch.121	A61-Post/Disc Ch.120
C62-GND	B62-Post/Disc Ch.123	A62-Post/Disc Ch.122
C63-GND	B63-Post/Disc Ch.125	A63-Post/Disc Ch.124
C64-N/C	B64-Post/Disc Ch.127	A64-Post/Disc Ch.126
C65-N/C	B65-GND	A65-N/C

Sequencer Module Auxiliary Connector Pin List

(Viewed From Front of Crate-10/10/90)

C01-H53MHZ,Ø1 Clock, Slot 25,23,	B01-Reset	A01-H53MHZ,Ø1 Clock,Slot 11,
C02-L53MHZ,Ø1 Clock, 21,19,17,15	B02-GND	A02-L53MHZ,Ø1 Clock,9,7,5,3,1
C03-Hit Data 0, Slot 24	B03-Fiber Error	A03-Hit Data 0, Slot 2
C04-Hit Data 1, Slot 24	B04-GND	A04-Hit Data 1, Slot 2
C05-Hit Data 2, Slot 24	B05-Fiber Wait	A05-Hit Data 2, Slot 2
C06-Hit Data 3, Slot 24	B06-GND	A06-Hit Data 3, Slot 2
C07-Hit Data 4, Slot 24	B07-Fiber Clock	A07-Hit Data 4, Slot 2
C08-HIt Data 5, Slot 24	B08-GND	A08-Hit Data 5, Slot 2
C09-Hit Data 6, Slot 24	B09-Sync	A09-Hit Data 6, Slot 2
C10-Hit Data 7, Slot 24	B10-Sync Err	A10-Hit Data 7, Slot 2
C11-Data Valid, Slot 24	B11-GND	A11-Data Valid, Slot 2
C12-N/C	B12-Fiber D00	A12-N/C
C13-Hit Data 0, Slot 20	B13-GND	A13-Hit Data 0, Slot 6
C14-Hit Data 1, Slot 20	B14-Fiber D01	A14-Hit Data 1, Slot 6
C15-Hit Data 2, Slot 20	B15-GND	A15-Hit Data 2, Slot 6
C16-Hit Data 3, Slot 20	B16-Fiber D02	A16-Hit Data 3, Slot 6
C17-Hit Data 4, Slot 20	B17-GND	A17-Hit Data 4, Slot 6
C18-HIt Data 5, Slot 20	B18-Fiber D03	A18-Hit Data 5, Slot 6
C19-Hit Data 6, Slot 20	B19-GND	A19-Hit Data 6, Slot 6
C20-Hit Data 7, Slot 20	B20-Fiber D04	A20-Hit Data 7, Slot 6
C21-Data Valid, Slot 20	B21-GND	A21-Data Valid, Slot 6
C22-GND	B22-Fiber D05	A22-GND
C23-Hit Data 0, Slot 16	B23-GND	A23-Hit Data 0, Slot 10
C24-Hit Data 1, Slot 16	B24-Fiber D06	A24-Hit Data 1, Slot 10
C25-Hit Data 2, Slot 16	B25-GND	A25-Hit Data 2, Slot 10
C26-Hit Data 3, Slot 16	B26-Fiber D07	A26-Hit Data 3, Slot 10
C27-Hit Data 4, Slot 16	B27-GND	A27-Hit Data 4, Slot 10

C28-HIt Data 5, Slot 16	B28-Fiber D08	A28-Hit Data 5, Slot 10
C29-Hit Data 6, Slot 16	B29-GND	A29-Hit Data 6, Slot 10
C30-Hit Data 7, Slot 16	B30-Fiber D09	A30-Hit Data 7, Slot 10
C31-Data Valid, Slot 16	B31-GND	A31-Data Valid, Slot 10
C32-GND	B32-Fiber D10	A32-N/C
C33-Left 26 MHZ Clock	B33-GND	A33-Right 26 MHZ Clock
C34-Hit Data 0, Slot 14	B34-Fiber D11	A34-Hit Data 0, Slot 12
C35-Hit Data 1, Slot 14	B35-GND	A35-Hit Data 1, Slot 12
C36-Hit Data 2, Slot 14	B36-Fiber D12	A36-Hit Data 2, Slot 12
C37-Hit Data 3, Slot 14	B37-GND	A37-Hit Data 3, Slot 12
C38-Hit Data 4, Slot 14	B38-Fiber D13	A38-Hit Data 4, Slot 12
C39-HIt Data 5, Slot 14	B39-GND	A39-Hit Data 5, Slot 12
C40-Hit Data 6, Slot 14	B40-Fiber D14	A40-Hit Data 6, Slot 12
C41-Hit Data 7, Slot 14	B41-GND	A41-Hit Data 7, Slot 12
C42-Data Valid, Slot 14	B42-Fiber D15	A42-Data Valid, Slot 12
C43-GND	B43-GND	A43-N/C
C44-Hit Data 0, Slot 18	B44-Fiber Mux Enable	A44-Hit Data 0, Slot 8
C45-Hit Data 1, Slot 18	B45-Fiber User 2	A45-Hit Data 1, Slot 8
C46-Hit Data 2, Slot 18	B46-GND	A46-Hit Data 2, Slot 8
C47-Hit Data 3, Slot 18	B47-GND	A47-Hit Data 3, Slot 8
C48-Hit Data 4, Slot 18	B48-Event Address Valid	A48-Hit Data 4, Slot 8
C49-HIt Data 5, Slot 18	B49-Event Address Wrt En	A49-Hit Data 5, Slot 8
C50-Hit Data 6, Slot 18	B50-Event Address 0	A50-Hit Data 6, Slot 8
C51-Hit Data 7, Slot 18	B51-Event Address 1	A51-Hit Data 7, Slot 8
C52-Data Valid, Slot 18	B52-Event Address 2	A52-Data Valid, Slot 8
C53-N/C	B53-Event Address 3	A53-GND
C54-Hit Data 0, Slot 22	B54-Event Address 4	A54-Hit Data 0, Slot 4
C55-Hit Data 1, Slot 22	B55-Event Address 5	A55-Hit Data 1, Slot 4
C56-Hit Data 2, Slot 22	B56-Event Address 6	A56-Hit Data 2, Slot 4
C57-Hit Data 3, Slot 22	B57-Event Address 7	A57-Hit Data 3, Slot 4
C58-Hit Data 4, Slot 22	B58-GND	A58-Hit Data 4, Slot 4
C59-HIt Data 5, Slot 22	B59-GND	A59-Hit Data 5, Slot 4
C60-Hit Data 6, Slot 22	B60-Fiber User 1	A60-Hit Data 6, Slot 4
C61-Hit Data 7, Slot 22	B61-Fiber User 0	A61-Hit Data 7, Slot 4
C62-Data Valid, Slot 22	B62-GND	A62-Data Valid, Slot 4
C63-N/C	B63-Fiber Mux Control	A63-GND
C64-H53MHZ,Ø2 Clock, Slot 24,22,	B64-GND	A64-H53MHZ,Ø2 Clock,Slot 12,10,
C65-L53MHZ,Ø2 Clock, 20,18,16,14	B65-N/C	A65-L53MHZ,Ø2 Clock 8,6,4,2

TSM Module Auxiliary Connector Pin List

(Viewed From Front of Crate-10/9/90)

C01-H53MHZ,Ø1 Clock	B01-Post/Disc Ch.01	A01-Post/Disc Ch.00
C02-GND	B02-Post/Disc Ch.03	A02-Post/Disc Ch.02
C03-L53MHZ,Ø1 Clock	B03-Post/Disc Ch.05	A03-Post/Disc Ch.04
C04-GND	B04-Post/Disc Ch.07	A04-Post/Disc Ch.06
C05-GND	B05-Post/Disc Ch.09	A05-Post/Disc Ch.08
C06-GND	B06-Post/Disc Ch.11	A06-Post/Disc Ch.10
C07-GND	B07-Post/Disc Ch.13	A07-Post/Disc Ch.12
C08-Reset	B08-Post/Disc Ch.15	A08-Post/Disc Ch.14
C09-GND	B09-Post/Disc Ch.17	A09-Post/Disc Ch.16
C10-GND	B10-Post/Disc Ch.19	A10-Post/Disc Ch.18
C11-GND	B11-Post/Disc Ch.21	A11-Post/Disc Ch.20
C12-RES-OUT	B12-Post/Disc Ch.23	A12-Post/Disc Ch.22
C13-SYNC	B13-Post/Disc Ch.25	A13-Post/Disc Ch.24
C14-GND	B14-Post/Disc Ch.27	A14-Post/Disc Ch.26

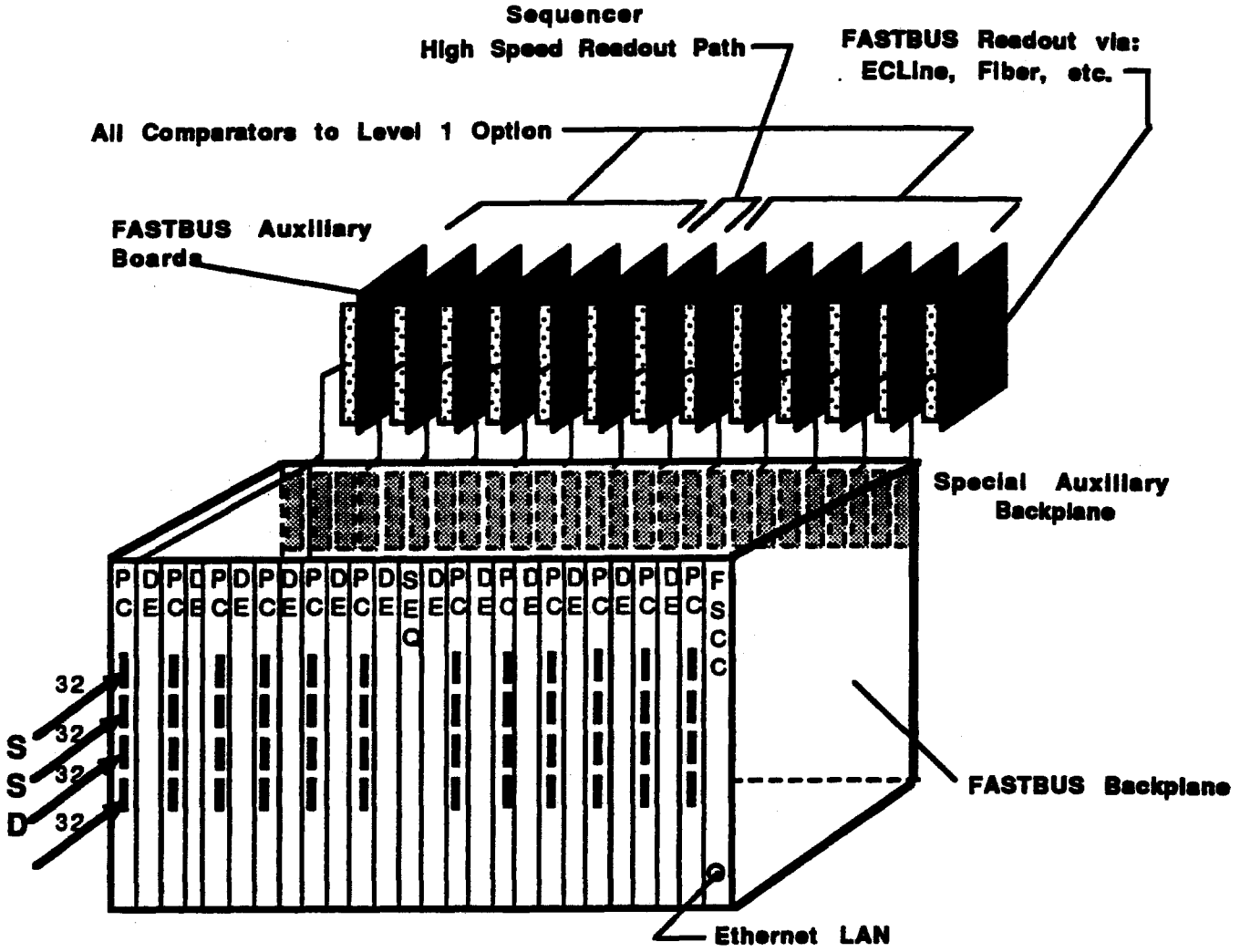
C15-SYNC-ERR	B15-Post/Disc Ch.29	A15-Post/Disc Ch.28
C16-GND	B16-Post/Disc Ch.31	A16-Post/Disc Ch.30
C17-GND	B17-Post/Disc Ch.33	A17-Post/Disc Ch.32
C18-HIT-DATA0	B18-Post/Disc Ch.35	A18-Post/Disc Ch.34
C19-HIT-DATA1	B19-Post/Disc Ch.37	A19-Post/Disc Ch.36
C20-GND	B20-Post/Disc Ch.39	A20-Post/Disc Ch.38
C21-HIT-DATA2	B21-Post/Disc Ch.41	A21-Post/Disc Ch.40
C22-HIT-DATA3	B22-Post/Disc Ch.43	A22-Post/Disc Ch.42
C23-GND	B23-Post/Disc Ch.45	A23-Post/Disc Ch.44
C24-HIT-DATA4	B24-Post/Disc Ch.47	A24-Post/Disc Ch.46
C25-HIT-DATA5	B25-Post/Disc Ch.49	A25-Post/Disc Ch.48
C26-GND	B26-Post/Disc Ch.51	A26-Post/Disc Ch.50
C27-HIT-DATA6	B27-Post/Disc Ch.53	A27-Post/Disc Ch.52
C28-HIT-DATA7	B28-Post/Disc Ch.55	A28-Post/Disc Ch.54
C29-GND	B29-Post/Disc Ch.57	A29-Post/Disc Ch.56
C30-GND	B30-Post/Disc Ch.59	A30-Post/Disc Ch.58
C31-DATA-VALID	B31-Post/Disc Ch.61	A31-Post/Disc Ch.60
C32-26MHZ(CLK3)	B32-Post/Disc Ch.63	A32-Post/Disc Ch.62
C33-GND	B33-Post/Disc Ch.65	A33-Post/Disc Ch.64
C34-EVENT-ADDR-VALID	B34-Post/Disc Ch.67	A34-Post/Disc Ch.66
C35-EVENT-WR-ENABLE	B35-Post/Disc Ch.69	A35-Post/Disc Ch.68
C36-GND	B36-Post/Disc Ch.71	A36-Post/Disc Ch.70
C37-ADDR0	B37-Post/Disc Ch.73	A37-Post/Disc Ch.72
C38-ADDR1	B38-Post/Disc Ch.75	A38-Post/Disc Ch.74
C39-GND	B39-Post/Disc Ch.77	A39-Post/Disc Ch.76
C40-ADDR2	B40-Post/Disc Ch.79	A40-Post/Disc Ch.78
C41-ADDR3	B41-Post/Disc Ch.81	A41-Post/Disc Ch.80
C42-GND	B42-Post/Disc Ch.83	A42-Post/Disc Ch.82
C43-ADDR4	B43-Post/Disc Ch.85	A43-Post/Disc Ch.84
C44-ADDR5	B44-Post/Disc Ch.87	A44-Post/Disc Ch.86
C45-GND	B45-Post/Disc Ch.89	A45-Post/Disc Ch.88
C46-ADDR6	B46-Post/Disc Ch.91	A46-Post/Disc Ch.90
C47-ADDR7	B47-Post/Disc Ch.93	A47-Post/Disc Ch.92
C48-GND	B48-Post/Disc Ch.95	A48-Post/Disc Ch.94
C49-GND	B49-Post/Disc Ch.97	A49-Post/Disc Ch.96
C50-GND	B50-Post/Disc Ch.99	A50-Post/Disc Ch.98
C51-GND	B51-Post/Disc Ch.101	A51-Post/Disc Ch.100
C52-GND	B52-Post/Disc Ch.103	A52-Post/Disc Ch.102
C53-GND	B53-Post/Disc Ch.105	A53-Post/Disc Ch.104
C54-GND	B54-Post/Disc Ch.107	A54-Post/Disc Ch.106
C55-GND	B55-Post/Disc Ch.109	A55-Post/Disc Ch.108
C56-GND	B56-Post/Disc Ch.111	A56-Post/Disc Ch.110
C57-GND	B57-Post/Disc Ch.113	A57-Post/Disc Ch.112
C58-GND	B58-Post/Disc Ch.115	A58-Post/Disc Ch.114
C59-GND	B59-Post/Disc Ch.117	A59-Post/Disc Ch.116
C60-GND	B60-Post/Disc Ch.119	A60-Post/Disc Ch.118
C61-GND	B61-Post/Disc Ch.121	A61-Post/Disc Ch.120
C62-53MHZ+	B62-Post/Disc Ch.123	A62-Post/Disc Ch.122
C63-53MHZ-	B63-Post/Disc Ch.125	A63-Post/Disc Ch.124
C64-N/C	B64-Post/Disc Ch.127	A64-Post/Disc Ch.126
C65-N/C	B65-GND	A65-N/C

Level Shifter Module Auxiliary Connector Pin List

(Viewed From Front of Crate-10/1/90)

C01-N/C	B01-Post/Disc Ch.00	A01-Post/Disc Ch.01
C02-GND	B02-Post/Disc Ch.02	A02-Post/Disc Ch.03
C03-N/C	B03-Post/Disc Ch.04	A03-Post/Disc Ch.05
C04-GND	B04-Post/Disc Ch.06	A04-Post/Disc Ch.07
C05-GND	B05-Post/Disc Ch.08	A05-Post/Disc Ch.09
C06-GND	B06-Post/Disc Ch.10	A06-Post/Disc Ch.11
C07-GND	B07-Post/Disc Ch.12	A07-Post/Disc Ch.13
C08-N/C	B08-Post/Disc Ch.14	A08-Post/Disc Ch.15
C09-N/C	B09-Post/Disc Ch.16	A09-Post/Disc Ch.17
C10-GND	B10-Post/Disc Ch.18	A10-Post/Disc Ch.19
C11-GND	B11-Post/Disc Ch.20	A11-Post/Disc Ch.21
C12-N/C	B12-Post/Disc Ch.22	A12-Post/Disc Ch.23
C13-GND	B13-Post/Disc Ch.24	A13-Post/Disc Ch.25
C14-GND	B14-Post/Disc Ch.26	A14-Post/Disc Ch.27
C15-GND	B15-Post/Disc Ch.28	A15-Post/Disc Ch.29
C16-GND	B16-Post/Disc Ch.30	A16-Post/Disc Ch.31
C17-GND	B17-Post/Disc Ch.32	A17-Post/Disc Ch.33
C18-GND	B18-Post/Disc Ch.34	A18-Post/Disc Ch.35
C19-GND	B19-Post/Disc Ch.36	A19-Post/Disc Ch.37
C20-GND	B20-Post/Disc Ch.38	A20-Post/Disc Ch.39
C21-N/C	B21-Post/Disc Ch.40	A21-Post/Disc Ch.41
C22-N/C	B22-Post/Disc Ch.42	A22-Post/Disc Ch.43
C23-GND	B23-Post/Disc Ch.44	A23-Post/Disc Ch.45
C24-N/C	B24-Post/Disc Ch.46	A24-Post/Disc Ch.47
C25-N/C	B25-Post/Disc Ch.48	A25-Post/Disc Ch.49
C26-GND	B26-Post/Disc Ch.50	A26-Post/Disc Ch.51
C27-N/C	B27-Post/Disc Ch.52	A27-Post/Disc Ch.53
C28-N/C	B28-Post/Disc Ch.54	A28-Post/Disc Ch.55
C29-GND	B29-Post/Disc Ch.56	A29-Post/Disc Ch.57
C30-N/C	B30-Post/Disc Ch.58	A30-Post/Disc Ch.59
C31-GND	B31-Post/Disc Ch.60	A31-Post/Disc Ch.61
C32-N/C	B32-Post/Disc Ch.62	A32-Post/Disc Ch.63
C33-N/C	B33-Post/Disc Ch.64	A33-Post/Disc Ch.65
C34-GND	B34-Post/Disc Ch.66	A34-Post/Disc Ch.67
C35-N/C	B35-Post/Disc Ch.68	A35-Post/Disc Ch.69
C36-GND	B36-Post/Disc Ch.70	A36-Post/Disc Ch.71
C37-N/C	B37-Post/Disc Ch.72	A37-Post/Disc Ch.73
C38-N/C	B38-Post/Disc Ch.74	A38-Post/Disc Ch.75
C39-GND	B39-Post/Disc Ch.76	A39-Post/Disc Ch.77
C40-N/C	B40-Post/Disc Ch.78	A40-Post/Disc Ch.79
C41-N/C	B41-Post/Disc Ch.80	A41-Post/Disc Ch.81
C42-GND	B42-Post/Disc Ch.82	A42-Post/Disc Ch.83
C43-N/C	B43-Post/Disc Ch.84	A43-Post/Disc Ch.85
C44-N/C	B44-Post/Disc Ch.86	A44-Post/Disc Ch.87
C45-GND	B45-Post/Disc Ch.88	A45-Post/Disc Ch.89
C46-N/C	B46-Post/Disc Ch.90	A46-Post/Disc Ch.91
C47-N/C	B47-Post/Disc Ch.92	A47-Post/Disc Ch.93
C48-GND	B48-Post/Disc Ch.94	A48-Post/Disc Ch.95
C49-N/C	B49-Post/Disc Ch.96	A49-Post/Disc Ch.97
C50-N/C	B50-Post/Disc Ch.98	A50-Post/Disc Ch.99
C51-GND	B51-Post/Disc Ch.100	A51-Post/Disc Ch.101
C52-GND	B52-Post/Disc Ch.102	A52-Post/Disc Ch.103
C53-GND	B53-Post/Disc Ch.104	A53-Post/Disc Ch.105
C54-GND	B54-Post/Disc Ch.106	A54-Post/Disc Ch.107
C55-GND	B55-Post/Disc Ch.108	A55-Post/Disc Ch.109
C56-GND	B56-Post/Disc Ch.110	A56-Post/Disc Ch.111

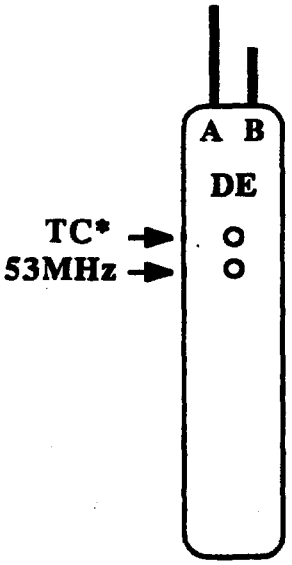
C57-GND	B57-Post/Disc Ch.112	A57-Post/Disc Ch.113
C58-GND	B58-Post/Disc Ch.114	A58-Post/Disc Ch.115
C59-GND	B59-Post/Disc Ch.116	A59-Post/Disc Ch.117
C60-GND	B60-Post/Disc Ch.118	A60-Post/Disc Ch.119
C61-GND	B61-Post/Disc Ch.120	A61-Post/Disc Ch.121
C62-N/C	B62-Post/Disc Ch.122	A62-Post/Disc Ch.123
C63-N/C	B63-Post/Disc Ch.124	A63-Post/Disc Ch.125
C64-N/C	B64-Post/Disc Ch.126	A64-Post/Disc Ch.127
C65-N/C	B65-GND	A65-N/C



SINGLE CRATE SUBSYSTEM (1536 CHANNELS)

PC - POSTAMP/COMPARATOR
DE - DELAY/ENCODER
SEQ - SEQUENCER
FSCC - FASTBUS SMART CRATE CONTROLLER

Figure 1

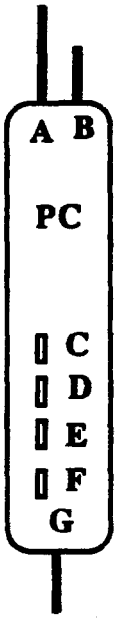


Connection:

A : 128 Hit Data.
B : Sequencer Port.

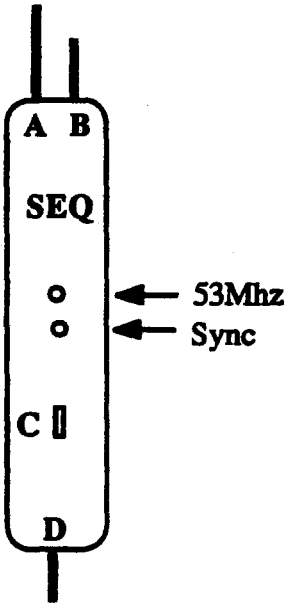
Connection:

A : 128 Input Data
B->E : 128 Diff. Output Data.



Connection:

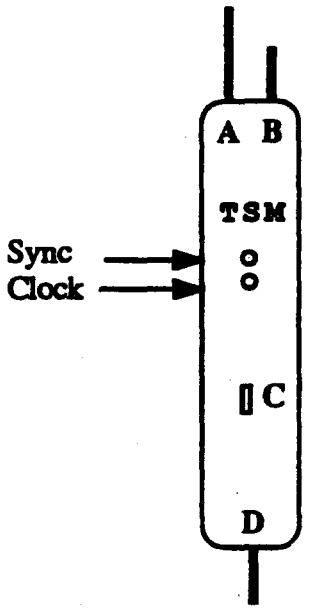
A : 128 Discriminated Output Data.
B : Sequencer Port.
C->F : 128 Differential Inputs Data.
G : FASTBUS Port.



Connection:

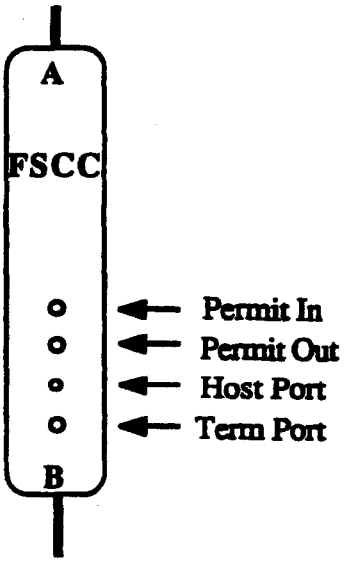
A : SSD Readout and Control.
B : Auxiliary Port Event Readout.
C : SEQ/MTC Control and Status.
D : FASTBUS Port.

Figure 2a: Module Symbols



Connection:

A - 128 Out Data.
B - Sequencer Port.
C - Fiber Receiver.
D - FASTBUS Port.



Connection:

A - Auxiliary Readout Port.
B - FASTBUS Port.

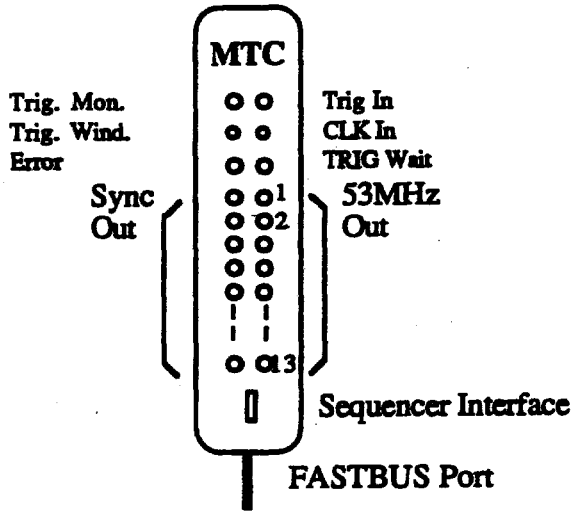


Figure 2b: Module Symbols

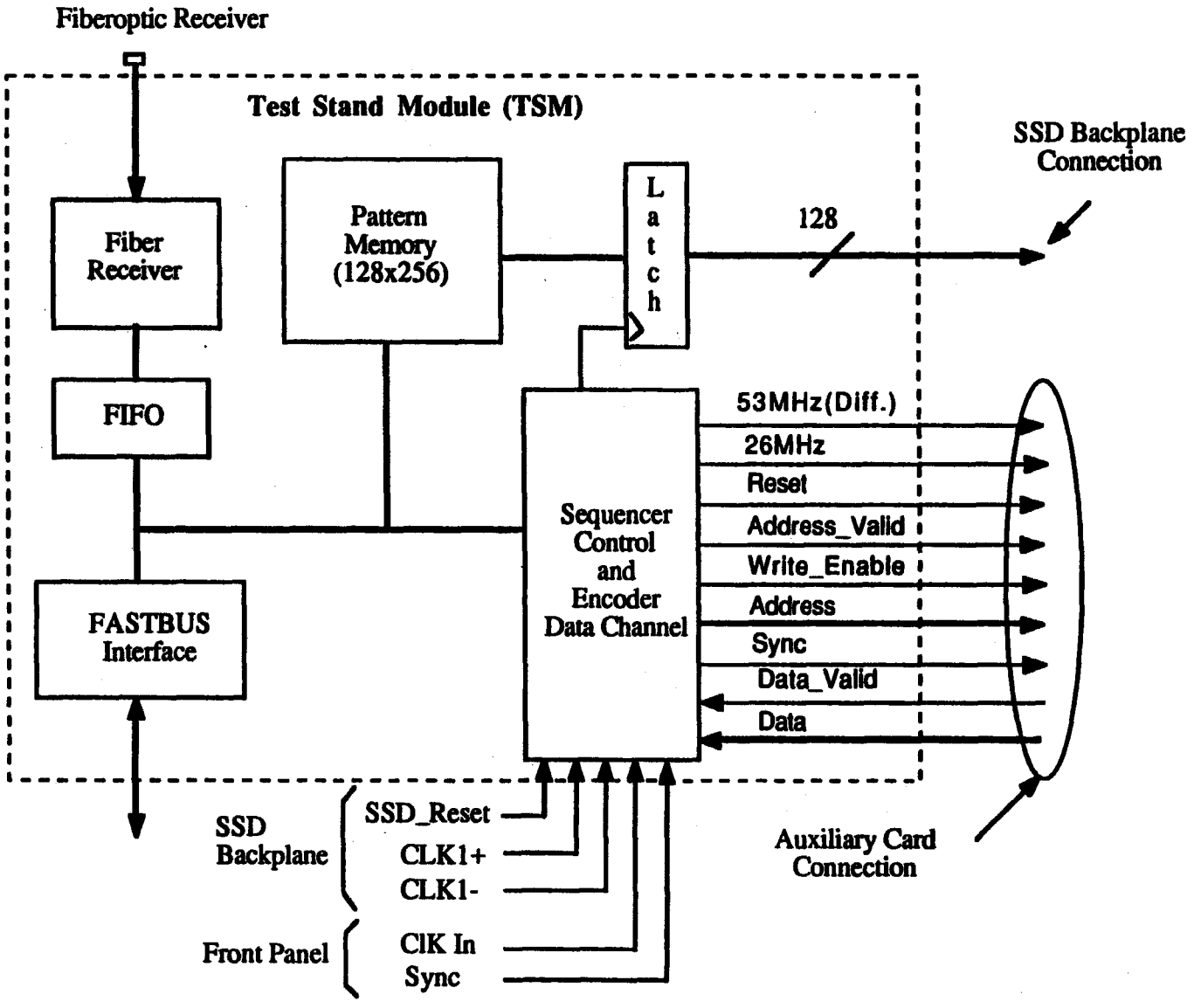
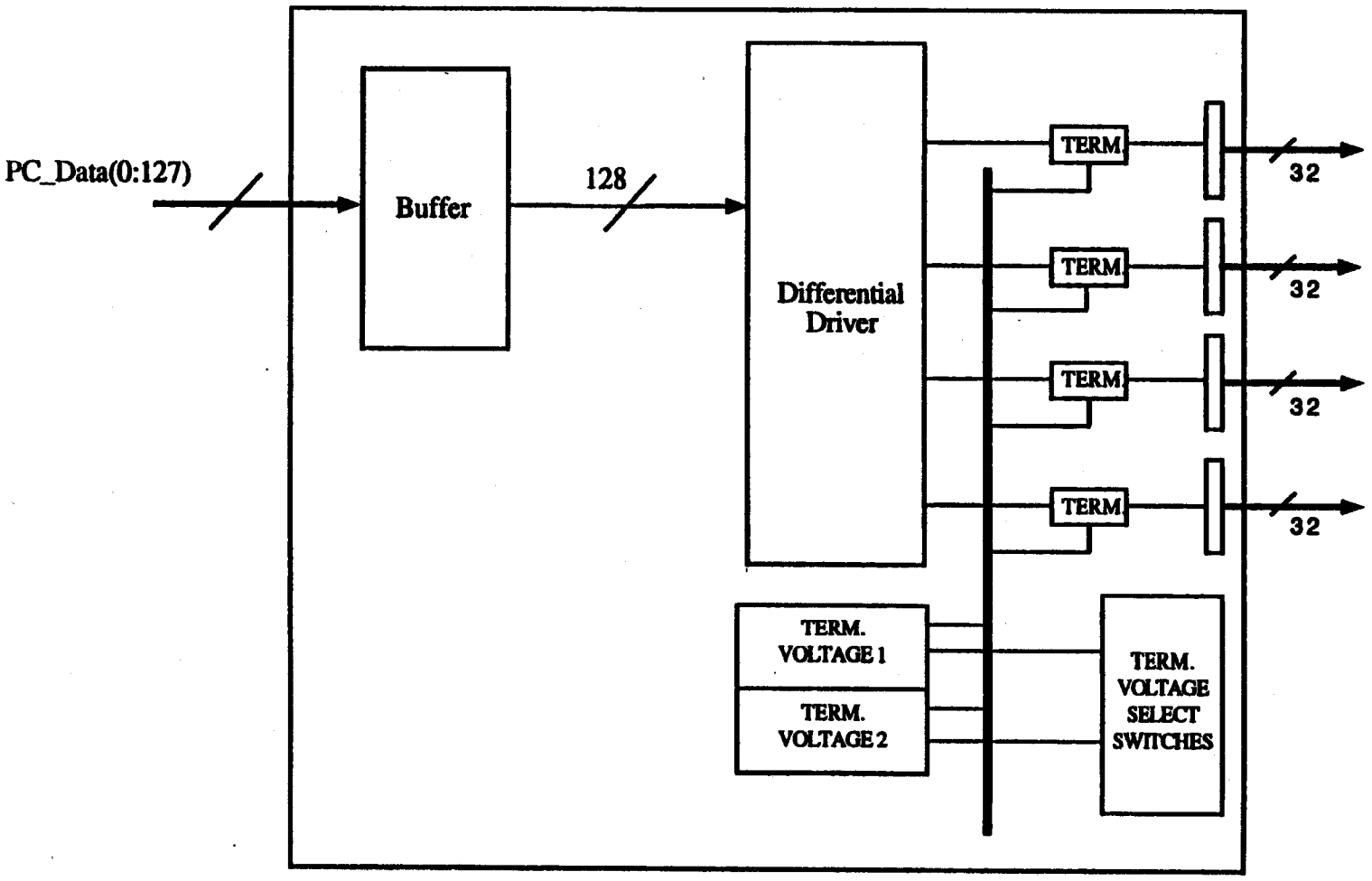


Figure 3a: TSM Block Diagram



Note: LS uses PC slot key.

Figure 3b: Level Shifter Block Diagram

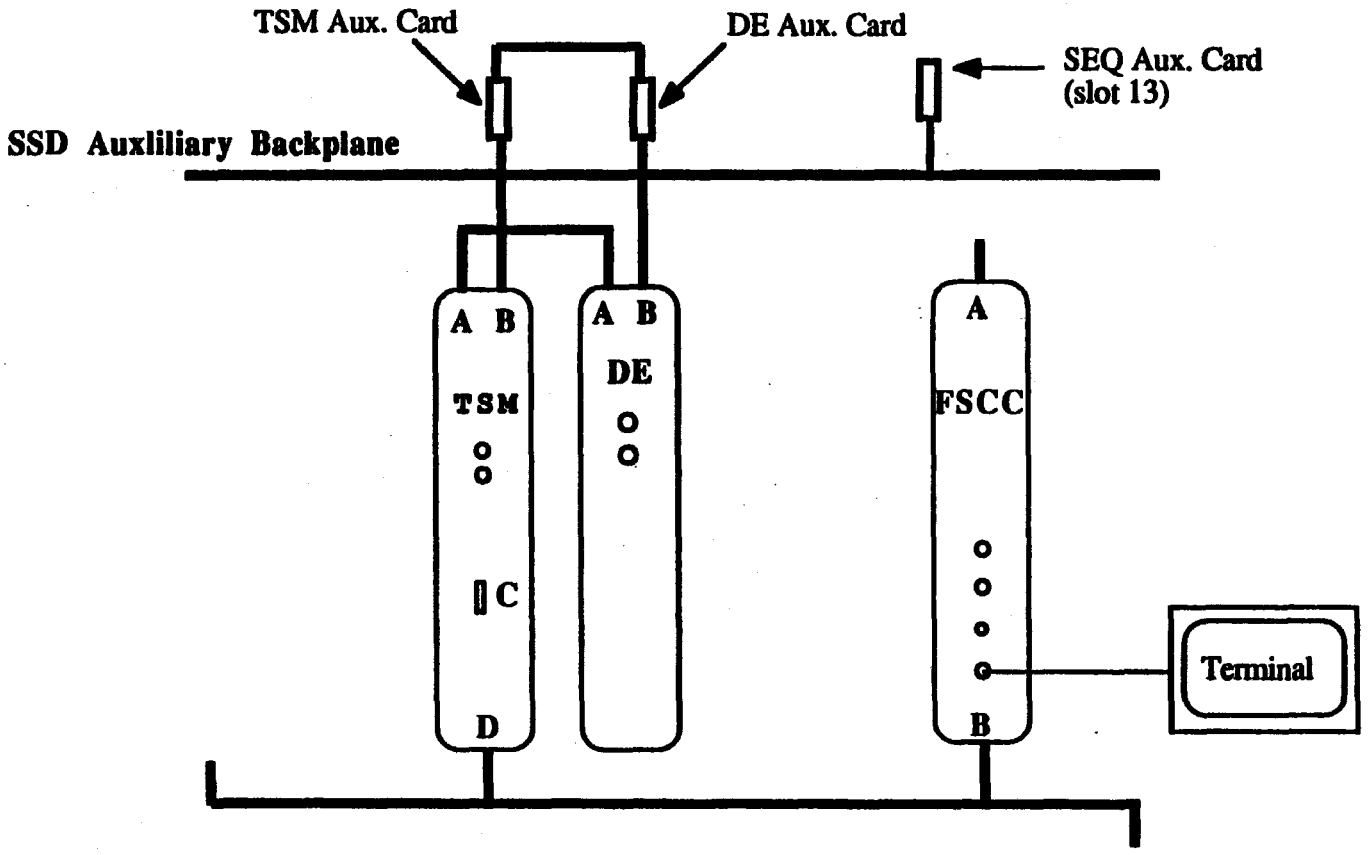


Figure 4: DE Testing with TSM

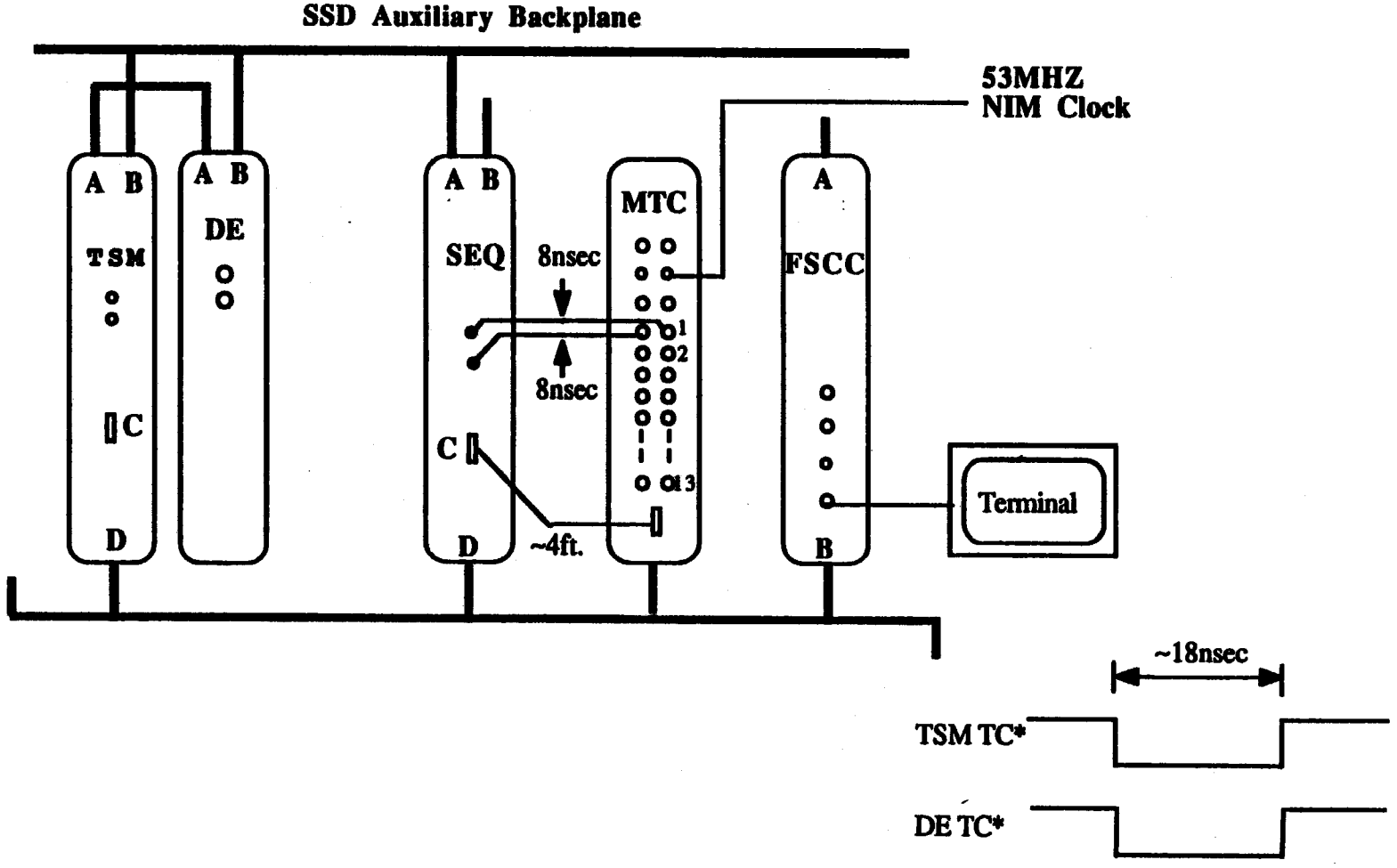


Figure 5: Configuration for SSD Single Channel Readout Test With TSM

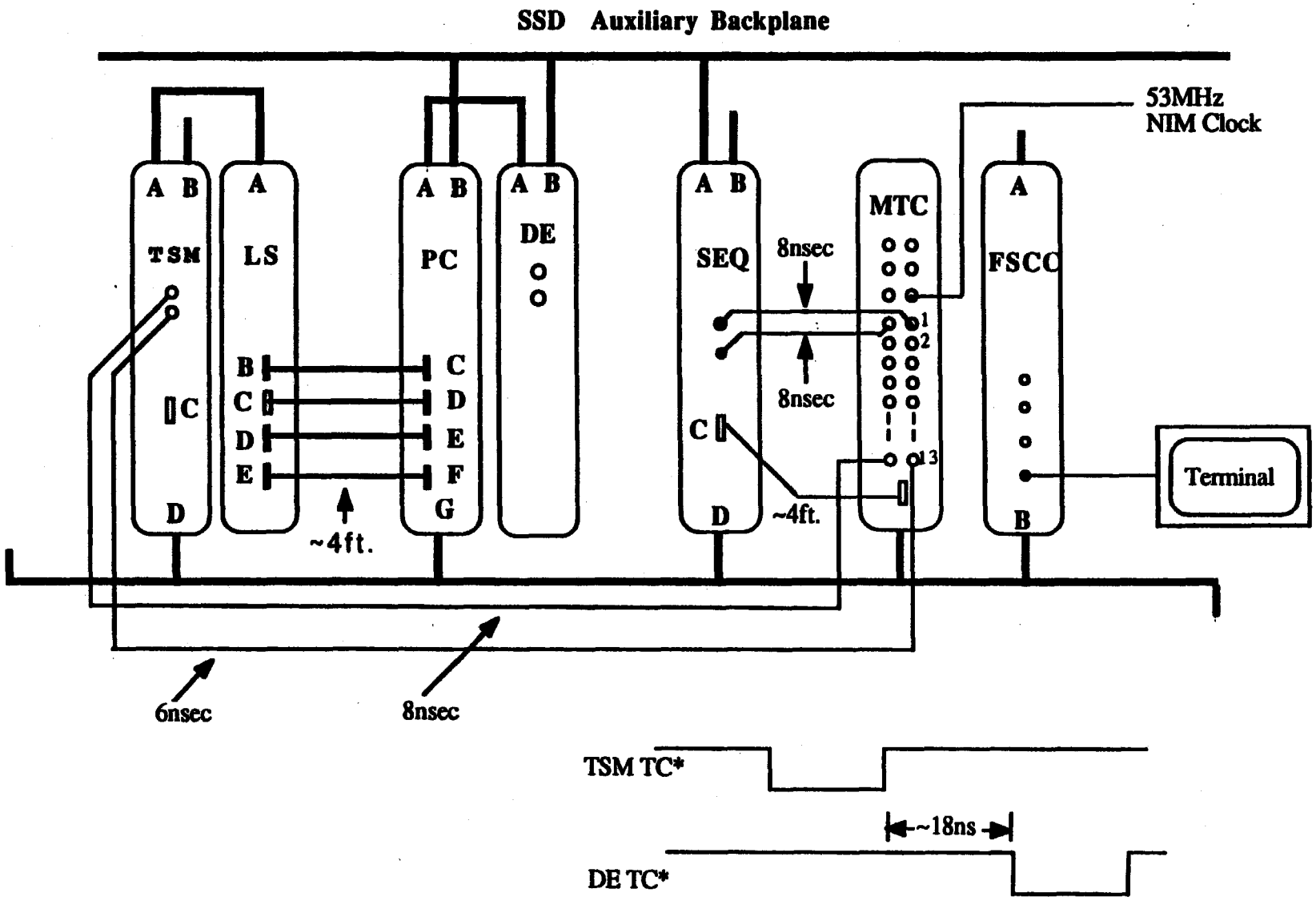


Figure 6: Configuration to Test PC Input Port

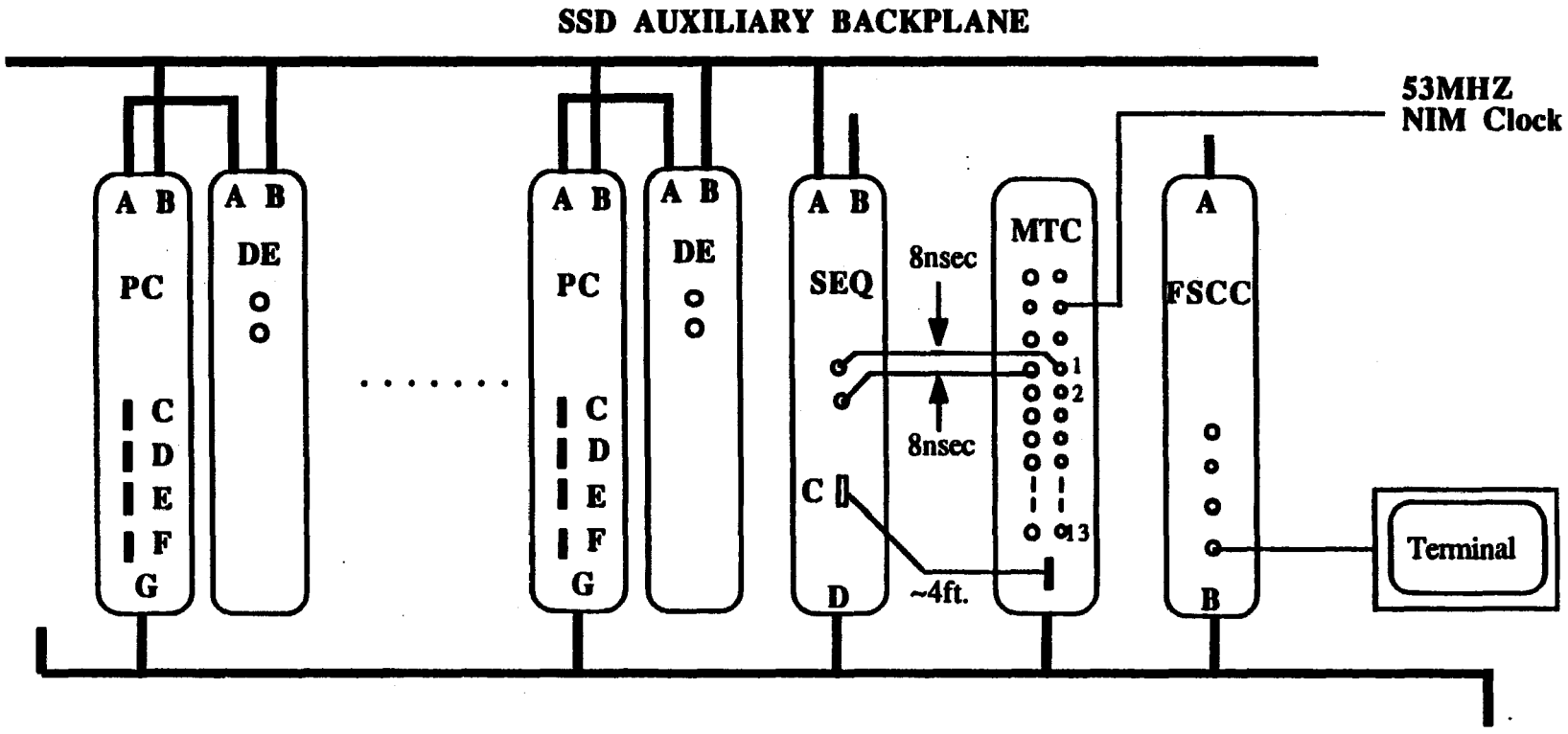


Figure 7: Configuration Test fo SSD Readout Test

Fermi National Accelerator Laboratory

SSD TEST STAND

--PRELIMINARY--

April 2, 1990

Carl Swoboda, Mark Bowden, Dave Christian, Miguel Fachin, Hector Gonzalez
Wolfgang Kowald, Garry Moore, Panayotis Spentzouris

1. GENERAL INFORMATION

The Silicon Strip Detector Readout System will contain hundreds of modules. An automated test stand is required to initially test the integration of each of the modules into a system that operates properly together at the crate level . The techniques and hardware developed for the initial tests will be used to test the production modules as they become available and before they are installed in the field. The Test Stand will also be used after installation into the experiment by support personnel to maintain the system and the modules.

1.1 Purpose

The Test Stand will be used to test the modules used in the Silicon Strip Detector System at the crate level. It is intended to be used as an automated test system to test production quantities of all the modules in the system. The SSD Test Stand is not intended to be a module development station. Each module in the system is assumed to have been fully developed. Typical problems detected in Test Stand tests might be due to failed parts, wrong parts, board solder shorts, etc. Design problems such as signal cross-talk, logic design errors, etc. should have been resolved long before these tests are run. On the other hand, it will be possible to use a special diagnostic module (please see Test Stand Module (TSM) specification) to perform tests on an installed system to help resolve problems that may be out of the ordinary.

1.11 Goals

1. Test modules at the crate/module level
2. Menu driven diagnostics
3. Testing as close as possible to actual system operation
4. Automatic test pattern selection
5. Field use desireable

The Test Stand will be capable of testing the modules that comprise the SSD system together at the crate level. The test will be as close to normal system operation as practical. It will be possible to install a test module in a field installed system to test the modules in an individual crate. Software diagnostics will be menu driven and will be user friendly. The Test Stand must be capable of helping the operator determine if any of the modules in the system are operating properly at the module level.

2. THEORY OF OPERATION AND OPERATING MODES

The Test Stand will require a working module of each type of module since the test is a module substitution technique. The Test Stand is not intended to be a module development station. The use of the Test Stand assumes that the Test Stand has been tested by running the software diagnostics on modules that are known to be good. After that test is complete the module under test can be substituted for the known good module and the tests repeated. This should allow rapid automated testing of each of the system modules in an environment as close as possible to the real system. Errors at the board level will be detected and as much detail as possible will be reported. This of course assumes that the individual modules have error registers that can be read.

2.1 Basic Operation

A block diagram of the Test Stand is shown in Figure 1. The Test Stand Module (TSM), shown shaded in Figure 1, contains a pattern memory, control logic, and a fiberoptic receiver. The FASTBUS Smart Crate Controller (FSCC) will be down-loaded with a test pattern through a RS232 or Ethernet link. The FSCC will load the pattern memory in the TSM with that pattern and start system data acquisition. A simulated trigger signal will be generated that causes the Main Timing Controller (MTC) to generate an event encode address to the Delay/Encoder (D/E). The D/E will encode the pattern and place it into the Sequencer (SEQ) event buffer. The SEQ will also transmit the data over the fiberoptic link to the fiber FIFO in the TSM. The FSCC will then read the SEQ event buffer and the fiber buffer and compare the data to the pattern loaded into the TSM pattern memory. Errors will be reported to the operator.

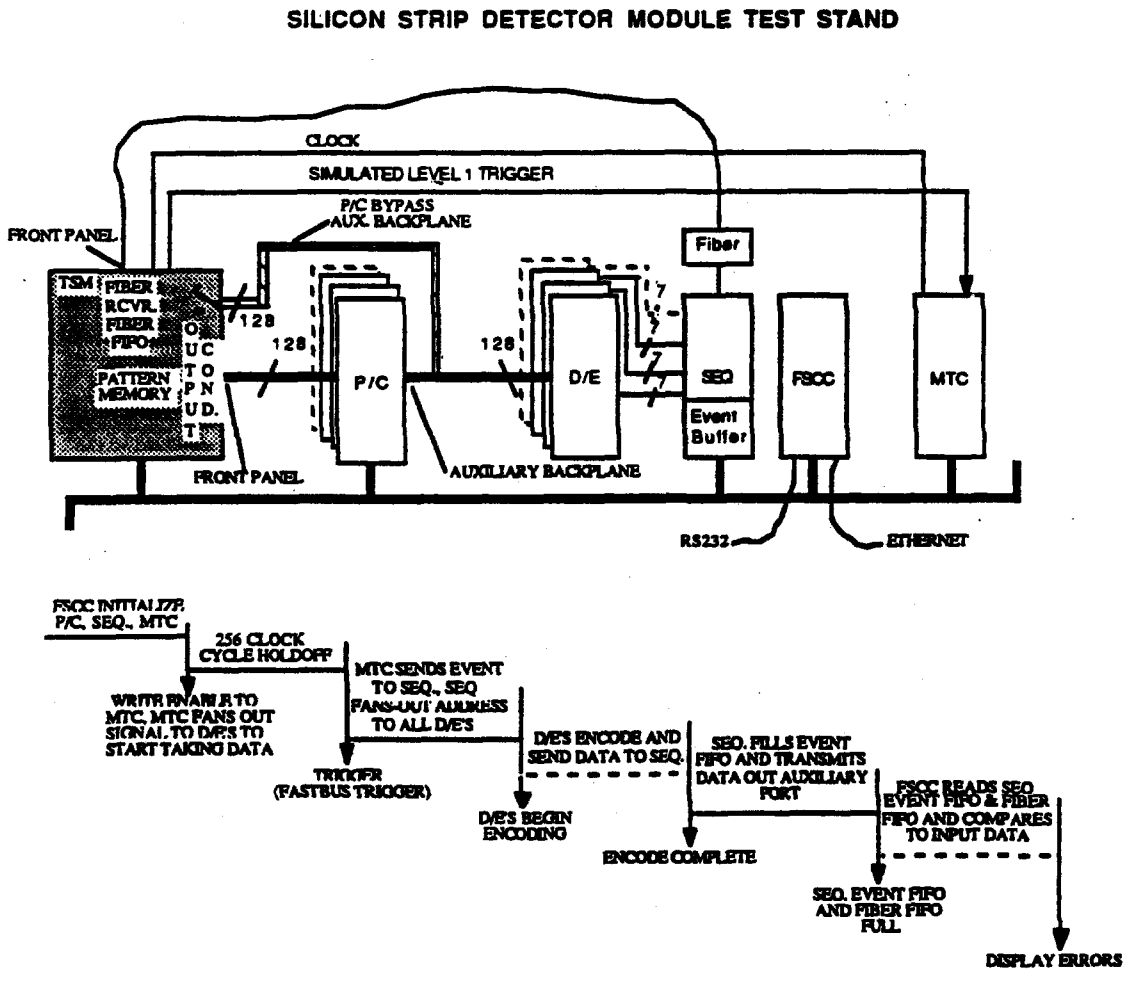


Figure 1; Test Stand Block Diagram

3. SYSTEM SOFTWARE DESCRIPTION

The Test Stand application software will incorporate software that was written for each module during the development stage if possible. Overall system test software requirements are similar to the application programs that must be written for actual system operation. The FSCC will be used as the FASTBUS Master in the Test Stand. The FSCC will be down loaded with a program that will initialize the individual modules in the system, start data acquisition and encoding, and finally compare transmitted hit patterns with those received by the SEQ and the fiberoptic FIFO buffer. Inconsistencies in the received data will be flagged and reported to the operator. The test options and selections will be available from a menu driven display.

The application software must be capable of performing the following operations on the SSD system modules through the FASTBUS interface on the appropropriate modules. The operations listed are basically initialization routines and must be augmented by the software that performs tests on the integrated system.

Test Stand Module
- Write/Read Pattern Memory
- Read Fiberoptic FIFO
- Start Data Acquisition (Simulated Level 1 to MTC)

Master Timing Controller
- Reset
- Write Enable: Start Data Acquisition
- Calibration Mode
- Debug Mode
- Clock Phase Adjust (Programmable Delay)
- Trigger Pipeline Depth
- Read Address Offset
- Errors

Postamp/Comparator
- CSR0
- Run/Halt
- Enable Disable Latches
- Clear Latches
- Enable Sum Channel Latches
- Clear Sum Channel Latches
- Enable/Disable Clock to Diagnostic Counter
- Enable/Disable Clock to D/A(s)
- CSR1
- Data to Diagnostic Counter
- D/A Data (Thresholds)
- Errors

Sequencer
- Write/Read Programmable Delay
- Write/Read Plane/Encoder I.D. RAM
- Write/Read Block/Word Count
- Read Event Buffer
- Errors

3.1 SUGGESTED TESTS TO PERFORM USING THE SSD TEST STAND

INITIALIZATION (Performed by FSCC)
- Postamp/Comparator
- Sequencer
- MTC
- TSM (Pattern Generator)

1. FSCC sends write or read to selected module register
   a. Continuous until stopped by operator.

2. FSCC sends MTC Address/Write Enable/Sync
   a. Continuous until stopped by operator.

3. Input to Postamp/Compartor module from DCC's preamp card.
   a. Input to P/C is a known fixed pattern.
   b. Delay/Encoder encodes pattern.
   c. Sequencer places addresses into event buffer.
   d. Continuous until stopped by operator.

4. FSCC reads event buffer in Sequencer until stopped by oeprator. (empty FIFO ??)

5. FSCC reads event buffer in Sequencer and compares readout with DCC fixed pattern preamp board.

6. FSCC loads pattern into TSM
   a. TSM transmits pattern to Postamp/Comparator.
   b. FSCC reads Sequencer event buffer and compares to pattern loaded into TSM.
   c. FSCC changes pattern in TSM - repeat starting at a.
   d. Continuous until stopped by operator.

FULL CRATE TEST *

7. Tests using Postamp/Comparator pattern generation mode - No TSM required.
   a. Load pattern into P/C.
   b. Start data acquisition.
   c. Read pattern (addresses) from Sequencer event buffer.
   d. Compare Pattern set into P/C to addresses read from Sequencer.
   e. Change pattern loaded into P/C - repeat starting at a.
   f. Change P/C crate address - repeat starting at a.
   g. Continuous until stopped by operator.

* A variation of this test can be used by vendors to test production quantities of the Postamp/Comparator and Delay/Encoder modules.

3.2 Software Approach

INPUT/OUTPUT SPECIFICATIONS

Test Stand Module (TSM)

FASTBUS Interface
128 outputs formatted for output to the Delay/Encoder over the Auxiliary backplane
128 outputs formatted for output to the Postamp/Comparator through the front panel
Fiber Receiver/Driver; four optical inputs, one optical output
Simulated Clock Output
Simulated Level 1 Trigger

Master Timing Controller (MTC)

FASTBUS Interface
Clock Input
System Control as Detailed in MTC Specification

Postamp/Comparator (P/C)

FASTBUS Interface
128 inputs through front panel

Sequencer

FASTBUS Interface
System Control as Detailed in Sequencer Specification

FASTBUS Smart Crate Controller

FASTBUS Interface
RS232
Ethernet

4.1 Communication Interfaces

Communication with the outside world will take place through the RS232 or the Ethernet ports on the FSCC.

5. PHYSICAL SPECIFICATIONS

5.1 Packaging

The Test Stand will will be packaged in FASTBUS .

5.11 Physical Size

5.2 Power Requirements

5.3 Cooling Requirements

5.4 Integrated Circuits Used

5.5 Pin Configurations

APPENDIX A - Info

April 2, 1990
Fermi National Accelerator Laboratory
Computer Division
Data Acquisition Electronics Department

Draft Specification
Fiberoptic Data Transmission Link

Carl Swoboda/Garry Moore
March 21, 1989
Revision A : April 2, 1990

1.0 General Information

1.1 Overview

This fiberoptic data link can be thought of as a general purpose, optically isolated, point-to-point port through which information can be transmitted at relatively high data rates. A commercially available transmitter/receiver chipset, manufactured by Advanced Micro Devices (AMD) and called TAXI, is used. The transmitter chip encodes up to 10 parallel data bits, serializes them, and outputs the result at 125 Mbits/sec. This serial data stream is then fed to an optical transmitter which passes the data through a fiberoptic cable to a remote optical receiver. This optical receiver then converts the optical signal into an electrical bit stream that is applied to the input of the AMD receiver chip. The receiver chip decodes the data, performs a serial-to-parallel conversion, and then outputs the data in parallel along with a strobe which occurs at a data valid point. The result of this operation is the transfer of parallel data over a serial optical link that is basically transparent to the user.

In the Silicon Strip Detector (SSD) application, the fiberoptic data link is configured to transfer data at 50 Mbytes/sec. This rate requires that four individual fiber links be used, each transmitting 10 data bits (a total of 40 bits; 32 bits of data, 8 bits of status), at a rate of 10.42 Mbytes/sec.

1.11 Standard Bus System Used

The SSD readout system is being implemented in the FASTBUS standard. The fiberoptic data link aquires power (+5V) from the FASTBUS crate. Both the optical transmitter and receiver are physically mounted on FASTBUS auxiliary boards but have no physical connection to the FASTBUS backplane.

1.12 Number of Channels

Basically, the fiberoptic data link is a single channel port, connecting a transmitting node to a receiving node operating at a data transfer rate of 41.68 Mbytes/sec. Word width at the transmitting and receiving node is 40 bits (32 bits of data, 8 bits of status).

1.2 Application

The fiberoptic data link is a point-to-point pipelined data mover presently capable of transmitting data at 41.68 Mbytes/sec. Any application that may require data transfers at this rate or higher is a potential customer for this link.

1.3 Packaging

The fiberoptic data link requires no special packaging, and may be implemented in any system that provides the proper voltages. In the SSD case, the transmitting node is packaged on a FASTBUS auxiliary board. The receiving node can be implemented in any appropriate configuration or system capable of accepting 40 bits of parallel data and a strobe every 96 ns.

1.32 Pinout

The connections between the FASTBUS board and the auxiliary board containing the fiberoptic data link components will be as defined in Table I of this specification. The connection to the outside world will be through 5 fiberoptic cable connectors located on the auxiliary backplane board.

Table 1 : Sequencer Module Pin List

(Viewed From Front of Crate-3/7/89)

,Ø1 Clock, Slot 25,23,21	B01-Reset	A01-53MHZ,Ø1 Clock, Slot 12,10,8
,Ø1 Clock, Slot 19,17,15	B02-S0	A02-53MHZ,Ø1 Clock, Slot 6,4,2
ata 0, Slot 24	B03-S1	A03-Hit Data 0, Slot 1
ata 1, Slot 24	B04-S2	A04-Hit Data 1, Slot 1
ata 2, Slot 24	B05-S3	A05-Hit Data 2, Slot 1
ata 3, Slot 24	B06-S5	A06-Hit Data 3, Slot 1
ata 4, Slot 24	B07-LINK ERROR	A07-Hit Data 4, Slot 1
ata 5, Slot 24	B08-SSTROBE	A08-Hit Data 5, Slot 1
ata 6, Slot 24	B09-Sync	A09-Hit Data 6, Slot 1
ata 7, Slot 24	B10-Sync Err	A10-Hit Data 7, Slot 1
Valid, Slot 24	B11-D00	A11-Data Valid, Slot 1
-5.0 Volts	B12-D01	A12-VEE, -5.2 Volts
Data 0, Slot 20	B13-D02	A13-Hit Data 0, Slot 5
ata 1, Slot 20	B14-D03	A14-Hit Data 1, Slot 5
Data 2, Slot 20	B15-D04	A15-Hit Data 2, Slot 5
Data 3, Slot 20	B16-D05	A16-Hit Data 3, Slot 5
Data 4, Slot 20	B17-D06	A17-Hit Data 4, Slot 5
Data 5, Slot 20	B18-D07	A18-Hit Data 5, Slot 5
ata 6, Slot 20	B19-D08	A19-Hit Data 6, Slot 5
a 7, Slot 20	B20-D09	A20-Hit Data 7, Slot 5
lid, Slot 20	B21-D10	A21-Data Valid, Slot 5
	B22-D11	A20-GND
ata 0, Slot 16	B23-D12	A23-Hit Data 0, Slot 9
ata 1, Slot 16	B24-D13	A24-Hit Data 1, Slot 9
ata 2, Slot 16	B25-D14	A25-Hit Data 2, Slot 9
ata 3, Slot 16	B26-D15	A26-Hit Data 3, Slot 9
Data 4, Slot 16	B27-D16	A27-Hit Data 4, Slot 9
ata 5, Slot 16	B28-D17	A28-Hit Data 5, Slot 9
Data 6, Slot 16	B29-D18	A29-Hit Data 6, Slot 9
Data 7, Slot 16	B30-D19	A30-Hit Data 7, Slot 9
Valid, Slot 16	B31-D20	A31-Data Valid, Slot 9
	B32-D21	A32-VEE, -5.2 Volts
26 MHZ Clock	B33-D22	A33-Right 26 MHZ Clock
Data 0, Slot 14	B34-D23	A34-Hit Data 0, Slot 11
Data 1, Slot 14	B35-D24	A35-Hit Data 1, Slot 11
Data 2, Slot 14	B36-D25	A36-Hit Data 2, Slot 11
Data 3, Slot 14	B37-D26	A37-Hit Data 3, Slot 11
Data 4, Slot 14	B38-D27	A38-Hit Data 4, Slot 11
Data 5, Slot 14	B39-D28	A39-Hit Data 5, Slot 11
Data 6, Slot 14	B40-D29	A40-Hit Data 6, Slot 11
Data 7, Slot 14	B41-D30	A41-Hit Data 7, Slot 11
Valid, Slot 14	B42-D31	A42-Data Valid, Slot 11
	B43-D32	A43-VCC, +5.0 Volts
Data 0, Slot 18	B44-D33	A44-Hit Data 0, Slot 7
Data 1, Slot 18	B45-D34	A45-Hit Data 1, Slot 7
Data 2, Slot 18	B46-D35	A46-Hit Data 2, Slot 7
Data 3, Slot 18	B47-CLOCK	A47-Hit Data 3, Slot 7
Data 4, Slot 18	B48-Event Address Valid	A48-Hit Data 4, Slot 7
Data 5, Slot 18	B49-Event Address Wrt. En.	A49-Hit Data 5, Slot 7
Data 6, Slot 18	B50-Event Address 0	A50-Hit Data 6, Slot 7
Data 7, Slot 18	B51-Event Address 1	A51-Hit Data 7, Slot 7
Valid, Slot 18	B52-Event Address 2	A52-Data Valid, Slot 7
+5.0 Volts	B53-Event Address 3	A53-GND
Data 0, Slot 22	B54-Event Address 4	A54-Hit Data 0, Slot 3
Data 1, Slot 22	B55-Event Address 5	A55-Hit Data 1, Slot 3
Data 2, Slot 22	B56-Event Address 6	A56-Hit Data 2, Slot 3
Data 3, Slot 22	B57-Event Address 7	A57-Hit Data 3, Slot 3
Data 4, Slot 22	B58-D36	A58-Hit Data 4, Slot 3
Data 5, Slot 22	B59-D37	A59-Hit Data 5, Slot 3
Data 6, Slot 22	B60-D38	A60-Hit Data 6, Slot 3
Data 7, Slot 22	B61-D39	A61-Hit Data 7, Slot 3
Valid, Slot 22	B62-STROBE	A62-Data Valid, Slot 3
-5.2 Volts	B63-ACKNOWLEDGE	A63-GND
Z,Ø2 Clock, Slot 24,22,20	B64-GND	A64-53MHZ,Ø2 Clock, Slot 11,9,7
Z,Ø2 Clock, Slot 18,16,14	B65-VTT, -2 Volts	A65-53MHZ,Ø2 Clock, Slot 5,3,1

1.33 N/A

1.4 Power Requirements

The auxiliary card implementation will require the following voltages and currents:

Receiver	+5v @ 1.75A
Transmitter	+5v @ 1.5A

1.41 Control and Monitoring Requirements

The fiberoptic data link uses a simple standard protocol to communicate between transmitting and receiving nodes.

1.5 Cooling Requirements

The fiberoptic data link auxiliary board will be cooled by air passing over the surface of the auxiliary board.

1.6 Unusual Requirements N/A

2.0 Theory of Operation

2.1 Basic Operation

The fiberoptic data link is a data pipeline port through which parallel data is encoded, serialized, and transmitted optically over fiberoptic cable. At the receiving node the data is optically received, converted to electrical signals, decoded, de-serialized and then made available as parallel multi-bit words.

A block diagram of the fiberoptic data link is shown in Figure 1. This description assumes that there is a one-to-one correspondence between the transmitting node and the receiving node. In other words, there is one receiver connected to one transmitter. An alternative arrangement might be to connect multiple transmitting nodes to a single receiving node. In this case a Permit to Transmit is used at each SEQ or FSCC to control the number of nodes transmitting and the data is or'ed optically or electrically at the receiver.

In either case, the fiberoptic data link is operated according to a protocol that defines a very simple communication interchange between a transmitting and receiving node. The protocol does not stipulate a data transfer rate since future implementations of this link will likely be at higher data transfer rates.

The link itself has no intelligence. Data is applied to the link in parallel, received at the remote end, and then presented in parallel as though a multiconductor cable were connecting the two devices.

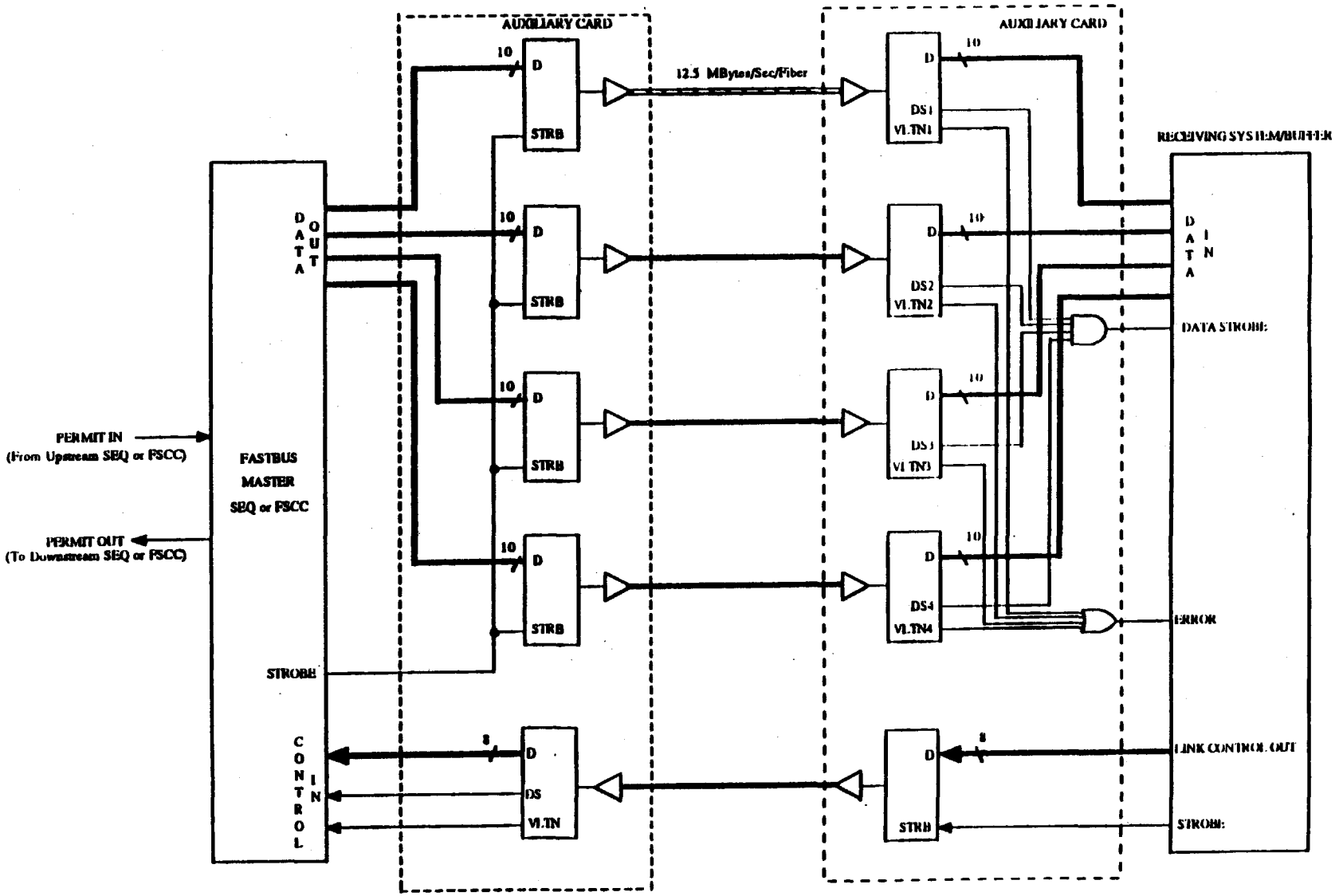


Figure 1; Fiberoptic ink Block Diagram

The fiberoptic data link contains five separate fiberoptic channels, each operating at 10.42 Mbytes/sec. Four of the fibers carry pipeline data and status information to the remote node. The fifth fiber is used to transmit acknowledge or error information from the remote node back to the originating node. The four fibers carrying data to the remote node are operating at an aggregate rate of 41.68 Mbytes/sec, with the acknowledge link operating at 12.5 Mbytes/sec. Commercially available communication chips and optical components are used in the implementation of the link.

Appendix A of this document details the communication chips (AMD TAXI Chips) as well as the optical components used in this implementation.

2.2 N/A

2.3 Addressing Modes

The fiberoptic data link contains no system interface and is therefore not addressable.

2.31 Data Transfer Description and Transfer Rates

The FASTBUS data acquisition system communicates with the fiberoptic link through the FASTBUS auxiliary backplane. A FASTBUS module connects to the link through an auxiliary backplane connector. The fiberoptic data link components are mounted on a FASTBUS auxiliary board that plugs into the auxiliary backplane at the back of a standard FASTBUS crate. The link accepts 40 bit wide (32 bits of data, 8 bits of status) data words and a strobe using TTL logic levels. The FASTBUS transmitting module is responsible for implementing the simple communication protocol as detailed in Appendix A. through the use of the 8 status bits. The module must also supply the link with 32 bits of parallel data and a strobe. The rate of data transfer to the fiberoptic link is controlled by the FASTBUS transmitting module. A maximum rate of transfer between the module and the link is a 40 bit word every 80 ns. The transmitting link has the ability to signal the FASTBUS transmitter that is cannot accept data. This status flag can be used by the module to time the transmissions over the link. The optical receiving node can output a 40 bit data word and strobe at a maximum data rate of one per 80 ns.

An example of a data transfer using the link protocol is shown in Figure 2. The transmitter begins the communication session by transmitting a Start of Message (SOM) control code. The transmitting node waits for an acknowledge transmission from the receiving node for a specific amount of time (timeout protection). If and when the acknowledge signal arrives at the transmitter, data transmission begins. At the end of data an End of Message (EOM) is transmitted to the receiving node. The receiving node acknowledges that it has received the EOM code. Additional data transmissions could follow without waiting for the initial SOM/acknowledge again. The SOM merely tests the integrity of the link before transmitting data.

Figure 2 also contains an example of a communication session in which the receiving node detects an error. In this case an error message is transmitted back to the transmitting node. The transmitting node will continue to transmit data during the time it takes for the receiving node to detect an error and transmit that information back to the transmitting node. When the transmitting node realizes that an error message has arrived, an error handling routine must be activated that resolves the problem based on system level considerations.

3.0 Input/Output Specifications

3.1 Communication Interfaces

This link is in itself a communication port between a transmitting and receiving no implemented in almost any system. The fiberoptic link and a simple data pipeline communicatio.. protocol provides both electrical and logical system isolation. The transmitting node accepts data in parallel, transmits the data, receives the data at the remote node, and makes the data available in parallel as though a muticonductor cable were connected between the two points. This transparency makes the connection of two different systems relatively easy. The "bridge" between different systems is implemented in the design of the transmitting and receiving node interfaces to the fiber link.

3.11 Description and Operation of Ports

The fiberoptic link has a 40 bit (32 bits of data, 8 bits of status) parallel input port at the transmitting node and a 40 bit (32 bits of data, 8 bits of status) parallel output port at the receiving node. The link input port is accessed through a connection between a FASTBUS module and the FASTBUS auxiliary card containing the fiberoptic link components. The link output port will be available for use in any system implementation that is capable of accepting a 40 bit parallel data word.

3.12 Communication Protocol

The communication protocol will be implemented according to the emerging fiberoptic data link standard presently under review by the FASTBUS Standards Committee.

3.2 Analog/Digital Signals

3.21 Logic Levels

Digital signal levels at the input to the optical transmitting node and at the output of the opt.. receiving node are at TTL levels.

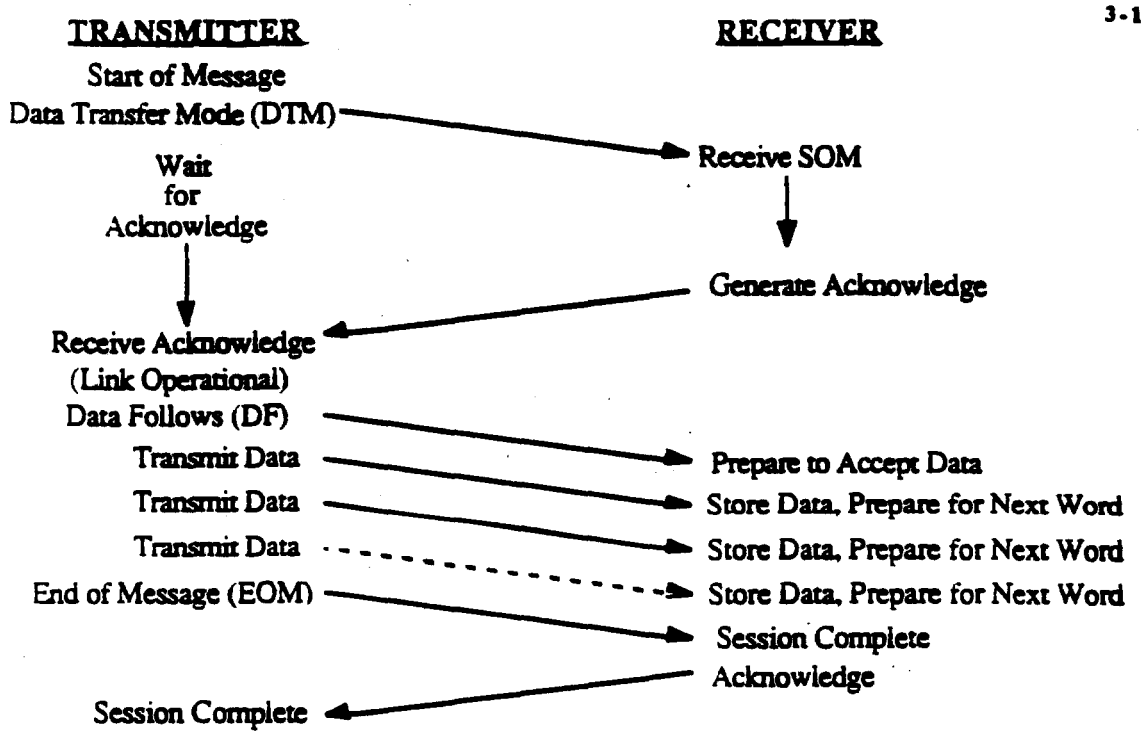
4.0 System Software

4.1 Initialization N/A

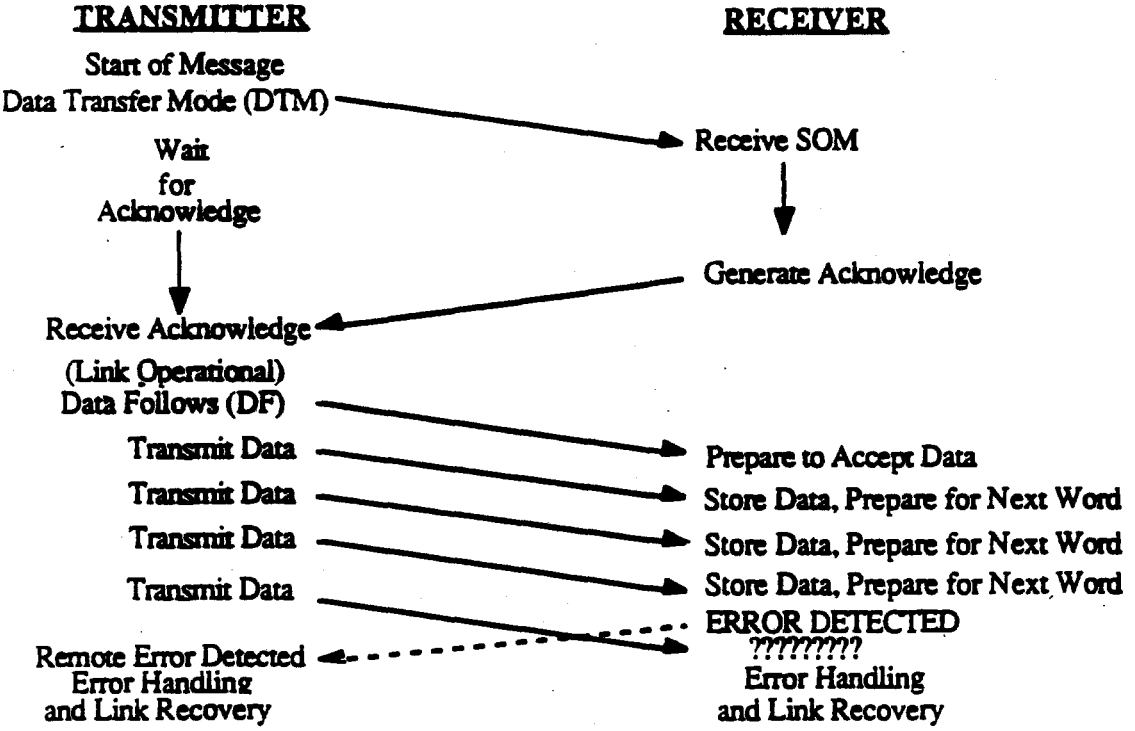
4.2 System Software

Control of the link both at the transmitting and receiving end is the responsibility of the devices to which the link is interfaced. The SSD FASTBUS Smart Crate Controller and/or Sequencer will contain the required link control logic.

COM SESSION
3-13-8



Normal Communication Session



Communication Session With Remote Error Detected

Figure 2: Communication Session Example

PN 434
Version : V1.0
12-Nov-1990

Silicon Strip Detector System Single Board Diagnostic Tests

Wolfgang Kowald
Duke University, E771

Panagiotis Spentzouris
UOA, E771

Marcus Larwill, Garry Moore
Data Acquisition Electronics Group
Dave Slimmer
Data Acquisition Software Group
Fermilab Computing Division

+++DRAFT+++

This document describes the diagnostic test software for individual boards in the Silicon Strip Detector (SSD) system.

KEYWORDS: Diagnostic, FASTBUS, SSD, Silicon Strip Detector

Systems Supported: pSOS, VMS V5.3
Software Version:

CONTENTS

CHAPTER 1

SSD SINGLE BOARD TESTS

1.1 INTRODUCTION

The single board tests are a collection of SSD board level diagnostics. Tests are included for the Postamp Comparator (PC), Delay Encoder (DE), Sequencer (SE), and Master Timing Controller (MTC). Each of the tests assume a working FASTBUS Smart Crate Controller (FSCC), and a working SSD FASTBUS crate and power supplies. In addition, some of the tests require special test modules. The purpose of these tests is twofold. First, the tests are intended for initial board qualification; and, second to diagnose and requalify a board that has broken in a previously working SSD system. In the case of a previously working system, it is assumed that a bad board is isolated by substitution or some other system level diagnosis method.

1.2 SINGLE BOARD TEST OPERATION FORMATS

The diagnostic can be run in a variety of ways, but the method with the fastest execution speed and most convenience is using the ROM version of the code. The ROM version of the diagnostic can be installed in FSCC ROM bank 2. (Copies can be made of the latest version in release from Computing Department master prom set.) If FSCC ROM bank 2 is not available, the diagnostic can be downloaded from the SSD_DIAG$tests product directory. Both the ROM version and downloaded version of the diagnostic run in FSCC RAM, so execution speed is the same. Downloading the diagnostic can take up to 10 minutes over a serial line, making this method far less convenient than using the ROM version. The diagnostic may also be run using remote procedure calls, using either the ethernet or serial front panel ports on the FSCC. Although the ethernet version does offer faster execution speed, both methods have relatively slow execution speed. Details of using the ROMed version follows.

1.3 ROM

The following sequence of commands copies the diagnostic code from FSCC ROM bank 2 to FSCC RAM, and starts the diagnostic.

{reset the FSCC manually with the front panel reset button}

```
$ SET HOST/DTE ttnn:          (establish serial communications with FSCC)
pROBE> go 40000               (run EPROM to RAM copy program)
pROBE> gs                     (start pSOS and do other board initializations)
pROBE> go                     (start diagnostic)
```

1.4 ORGANIZATION

The Single Board Tests are organized under a top level menu. Each of the Single Board Tests contain a main menu and submenus. There are also options in the top level menu to provide an interface for tailoring FASTBUS evironment parameters. The top level menu:

```
        SSD Single Board Test Menu
        ==========================
    1) Initialize FASTBUS
    2) Arbitrate (status = xxxx)
    3) Release Bus
    4) Set Long Timer (xxxx usec)
    5) Set Short Timer (xxxx usec)
    6) Set Primary Addresses
    7) Postamp Comparator menu
    8) TSM/Delay Encoder menu
    9) Sequencer menu
   10) Master Timing Controller menu
   11) Select Call mode (mode = Get Menu)
   12) Loop on tests switch (F)
   99) Exit to pROBE
   Enter Command:
```

1.4.1 Initialize FASTBUS

When the Single Board Test diagnostic is started, the FASTBUS environment is automatically initialized by the FASTBUS routines called by this menu option. Each FASTBUS Standard Routine that is part of this initializaion is displayed in a banner above the top level menu. The initialization creates a FASTBUS environment ID that is passed to the module tests, so an error will occur if this menu option is called more than once before calling menu option 3) Release Bus. This FASTBUS initialization routines are:

- o GPMINI - initialize interrupts and internal database
- o FBOPEN - open FASTBUS routines for use
- o FCIENV - allocate an environment ID and associated storage
- o FNPALL - allocate FASTBUS port for use
- o FBPSET(EG) - set environment parameter to enable geograhic addressing
- o FBPSET(COEN) - set environment parameter to enable FSCC data FIFO to processor FIFO copy
- o FBPSET(EOBI) - set environment parameter to enable end of block interrupt

1.4.2 Arbitrate

The menu option shows the current arbitor status of the FSCC, one of either "Master", "Slave", or "unknown". If this option is chosen, the FSCC tries to arbitrate for bus mastership using the selected arbitration level. It is not necessary for the FSCC to be bus master to run these tests.

1.4.3 Release Bus

This option will release the bus and delete the current FASTBUS environment ID using the following routines:

- o FRLENV
- o FBCLOS

1.4.4 Set Long Timer

The menu option displays the current environment value for the long timer. Selecting this option allows the long timer to be set to a new value. After selecting menu item 1) Initialize FASTBUS, the long timer will always be reset to 5000000 usec.

1.4.5 Set Short Timer

Same as above, but the short timer can only be enabled or disabled. Selecting 1500 enables and selecting 0 disables the timer.

1.4.6 Set Primary Address Menu

Set the primary addresses for the SSD modules. The submenu displays the current values of the PADs.

1.4.7 Postamp Comparator Menu

Queries user if 53MHz clock is present. After user [Y/N] response, calls the main menu for the PC tests.

1.4.8 TSM/Delay Encoder Menu

Calls the main menu for the Delay Encoder tests (which use the TSM).

1.4.9 Sequencer Menu

Calls the Sequencer tests main menu.

1.4.10 Master Timing Controller Menu

Calls the Master Timing Controller tests main menu.

1.4.11 Select Call Mode

Allows the call mode for the test to be modified. The call mode currently active is displayed by this menu item. There are four call modes possible, the default mode being to get the main menu of a single board test. The three other call modes include:

- Continue on Error - Tests do not terminate when an error is encountered.

- Stop on Error - Tests terminate when an error is encountered.

- Get Menu on Error - Tests terminate when an error is encountered and the appropriate menu is displayed.

Depending on implementation, an error message may or may not be displayed in each of the call modes described above.

1.4.12 Loop On Tests Switch

If this menu item displays (T), any test selected from the top level menu will repeat until a keyboard <Return> is entered. Otherwise, any test selected will execute only once.

1.4.13 Exit To PROBE

In the ROM version, this option will terminate the Single Board Tests with a pROBE break.

CHAPTER 2

POSTAMP COMPARATOR SINGLE BOARD TEST

2.1 INTRODUCTION

PC_TEST is a menu driven software tool capable of effectively testing and/or exercising all FASTBUS accessable circuitry of the PostAmp Comparator module. This software is designed to be used as an integral part of the Single Board Test software package.

2.2 TEST HARDWARE SETUP

The PostAmp Comparator stand-alone tests which are described here require a known working FSCC (Fastbus Smart Crate Controller) located in slot 0 of a functional SSD crate, and a PostAmp Comparator module. The PostAmp Comparator module must be located in a valid PostAmp Comparator slot of the SSD crate. The PostAmp Comparator Primary Address can be verified or changed in the top level menu.

The PostAmp Comparator module test can be executed in either automatic or manual test mode. In automatic mode, the user cannot select individual PostAmp Comparator Test options, but will perform an overall GO/NO-GO test of all FASTBUS circuitry on board the module, displaying errors where appropriate. The features tested using this mode are as follows :

Test All Geographical Addressing Circuitry

Data Space Addressing Mode Response

Response to Multiple CSR Space Addresses

Test All Secondary Addressing Ciruitry

Test the NTA Register

Write/Read Response to Valid NTA Addresses

Oh, 1h, 10h, C0000000 - C00000FF (64 DACs Onboard)

Write/Read Response to Invalid NTA Address

Test the CSR0 Register

Check Module ID

Set/Reset/Read Counter Clock Disable/Enable Bits

Set/Reset/Read Channel Force Zero Disable/Enable Bits

Set/Reset/Read Individual Channel Force Zero Disable/Enable Bits

Set/Reset/Read Mode Halt/Run Bits

Test Functionality of CLK1 On Bit

Write/Read Response to Unused CSR0 Bits

Test the CSR1 Register

Write/Read Test Counter Preset Byte (00 - FF)

Write/Read Response to Unused CSR1 Bits

Test the CSR10 Register

Read Onboard DAC Count

Write Response to Unused CSR10 Bits

Test Write/Read of Onboard DACs

Write/Read All Accessable DACs with FF Pattern

Write/Read All Accessable DACs with 00 Pattern

Write/Read All Accessable DACs with Alternating 55/AA Pattern

Write/Read All Accessable DACs with DAC Address

In manual mode, the PostAmp Comparator Test will be called, displaying the full PC_TEST menuing system, which will be described as follows. This option will also select the default error handling mode (see Chapter 1.3.11 for details). The features tested using this mode are as described above for the automatic mode, except for the added features of allowing the user to write a single DAC location using a user defined value,

read a single DAC location, write/read all accessable DACs with a user defined pattern, and allowing continuous looping of a process of series of processes for oscilloscope based testing.

2.3 PC_TEST MENUING SYSTEM

```
PC_TEST
Copyright (C) 1990 Universities Research Association, Inc.
All Rights Reserved.

      ----->  PC_TEST Main Menu
              -----------------

      -----> 1  Test Geographical Addressing
      -----> 2  Test Secondary Addressing
      -----> 3  Exercise PostAmp Comparator

      -----> 99  Exit PC_TEST Environment

       Enter Menu Option (1 - 3, 99) :
```

In response to the prompt, select an option by typing the option number located to the left. Menu selection errors will be flagged and the user will be prompted to re-enter the required option.

2.3.1 Option 1 Test Geographical Addressing:

```
      -----> Test Geographical Addressing
             ---------------------------

      -----> 1  Define Primary Address
      -----> 2  Address PostAmp Comparator
      -----> 3  Release AS-AK Lock
      -----> 4  Exercise Geographical Addressing

      -----> 99  Exit Geographical Addressing Menu

       Enter Menu Option (1- 4, 99) :
```

In response to the prompt, select an option by typing the option number located to the left. Menu selection errors will be flagged the user will be prompted to re-enter the required option.

2.3.1.1 Option 1 Define Primary Address: -

This option prompts the user to enter a valid slot location for the PostAmp Comparator module. Valid slot numbers are displayed for the user. Invalid slot selections will be flagged and the user will be prompted to re-enter a valid slot number.

2.3.1.2 Option 2 Address PostAmp Comparator: -

This option will attempt to perform a primary address to the specified PostAmp Comparator module. If unsuccessful, the user will be notified.

2.3.1.3 Option 3 Release AS-AK Lock: -

This option will attempt to release the current primary address AS-AK lock. If unsuccessful, the user will be notified.

2.3.1.4 Option 4 Exercise Geographical Addressing: -

This option will fully exercise the primary address circuitry of the PostAmp Comparator module. Situations tested for include response to multiple addresses and response to Data Space Addressing. The user will be notified of all occurring errors.

2.3.1.5 Option 99 Exit Geographical Addressing Menu: -

This option will return the user to the previous menu.

2.3.2 Option 2 Test Secondary Addressing:

```
-----> Test Secondary Addressing
       ------------------------

-----> 1  Test NTA Register
-----> 2  Test CSR 00
-----> 3  Test CSR 01
-----> 4  Test CSR 10
-----> 5  Test DAC Load and Read-back
-----> 6  Exercise Secondary Addressing
```

```
-----> 99  Exit Secondary Addressing Menu

 Enter Menu Option (1 - 6, 99) :
```

In response to the prompt, select an option by typing the option number located to the left. Menu selection errors will be flagged and the user will be prompted to re-enter the required option.

2.3.2.1 Option 1 Test NTA Register: -

```
-----> Test NTA Register
       -----------------

-----> 1  Write to NTA Register
-----> 2  Read from NTA Register
-----> 3  Exercise NTA Register

-----> 99  Exit NTA Register Menu

 Enter Menu Option (1 - 3, 99) :
```

In response to the prompt, select an option by typing the option number located to the left. Menu selection errors will be flagged and the user will be prompted to re-enter the required option.

2.3.2.1.1 Option 1 Write To NTA Register: -

This option will write a user specified value to the PostAmp Comparator NTA register. Valid NTA values are displayed for the user. Invalid NTA entries will be flagged and the user will be prompted to re-enter a valid NTA value. If unsuccessful, the user will be notified.

2.3.2.1.2 Option 2 Read From NTA Register: -

This option will allow the user to view the current value stored in the PostAmp Comparator NTA register. If unsuccessful, the user will be notified.

2.3.2.1.3 Option 3 Exercise NTA Register: -

This option will fully exercise the PostAmp Comparator NTA register. All valid NTA values are tested as well as their boundaries. The user will be notified of all occurring errors.

2.3.2.1.4 Option 99 Exit NTA Register Menu: -

This option will return the user to the previous menu.

2.3.2.2 Option 2 Test CSR 00: -

```
-----> Test CSR 00
       -----------

-----> 1  Set a Specific Bit in CSR 00
-----> 2  Read the Status of a Specific Bit in CSR 00
-----> 3  Read CSR 00
-----> 4  Exercise CSR 00

-----> 99  Exit CSR 00 Menu

 Enter Menu Option (1 - 4, 99) :
```

In response to the prompt, select an option by typing the option number located to the left. Menu selection errors will be flagged and the user will be prompted to re-enter the required option.

2.3.2.2.1 Option 1 Set A Specific Bit In CSR 00: -

```
-----> Write CSR 00
       ------------

-----> 1  Set Test Counter Clock Disable Bit
-----> 2  Set Sum Channel Force Zero Disable Bit
-----> 3  Set Individual Channel Force Zero Disable Bit
-----> 4  Set Mode Halt Bit
-----> 5  Set Test Counter Clock Enable Bit
-----> 6  Set Sum Channel Force Zero Enable Bit
-----> 7  Set Individual Channel Force Zero Enable Bit
-----> 8  Set Mode Run Bit
-----> 9  Write to CSR 00

-----> 99  Exit Write CSR 00 Menu
```

Enter Menu Option (1 - 9, 99) :

In response to the prompt, select an option by typing the option number located to the left. Menu selection errors will be flagged and the user will be prompted to re-enter the required option.

2.3.2.2.1.1 Option 1 Set Test Counter Clock Disable Bit: -

This option will set the Test Counter Clock Disable Bit and reset the Test Counter Clock Enable Bit of the PostAmp Comparator CSR 00 register. The Write to CSR 00 option must be executed for this option to take effect.

2.3.2.2.1.2 Option 2 Set Sum Channel Force Zero Disable Bit: -

This option will set the Sum Channel Force Zero Disable Bit and reset the Sum Channel Force Zero Enable Bit of the PostAmp Comparator CSR 00 register. The Write to CSR 00 option must be executed for this option to take effect.

2.3.2.2.1.3 Option 3 Set Individual Channel Force Zero Disable Bit: -

This option will set the Individual Channel Force Zero Disable Bit and reset the Individual Channel Force Zero Enable Bit of the PostAmp Comparator CSR 00 register. The Write to CSR 00 option must be executed for this option to take effect.

2.3.2.2.1.4 Option 4 Set Mode Halt Bit: -

This option will set the Mode Halt Bit and reset the Mode Run Bit of the PostAmp Comparator CSR 00 register. The Write to CSR 00 option must be executed for this option to take effect.

2.3.2.2.1.5 Option 5 Set Test Counter Clock Enable Bit: -

This option will set the Test Counter Clock Enable Bit and reset the Test Counter Clock Disable Bit of the PostAmp Comparator CSR 00 register. The Write to CSR 00 option must be executed for this option to take effect.

2.3.2.2.1.6 Option 6 Set Sum Channel Force Zero Enable Bit: -

This option will set the Sum Channel Force Zero Enable Bit and reset the Sum Channel Force Zero Disable Bit of the PostAmp Comparator CSR 00 register. The Write to CSR 00 option must be executed for this option to take effect.

2.3.2.2.1.7 Option 7 Set Individual Channel Force Zero Enable Bit: -

This option will set the Individual Channel Force Zero Enable Bit and reset the Individual Channel Force Zero Disable Bit of the PostAmp Comparator CSR 00 register. The Write to CSR 00 option must be executed for this option to take effect.

2.3.2.2.1.8 Option 8 Set Mode Run Bit: -

This option will set the Mode Run Bit and reset the Mode Halt Bit of the PostAmp Comparator CSR 00 register. The Write to CSR 00 option must be executed for this option to take effect.

2.3.2.2.1.9 Option 9 Write To CSR 00: -

This option will write a user specified value (as specified by the above enable/disable bit manipulations) to the PostAmp Comaparator CSR 00 register. If unsuccessful, the user will be notified.

2.3.2.2.1.10 Option 99 Exit Write CSR 00 Menu: -

This option will return the user to the previous menu.

2.3.2.2.2 Option 2 Read The Status Of A Specific Bit In CSR 00:

-----> Read CSR 00

-----> 1 Read Module ID Bits
-----> 2 Read CLK1 On Bit
-----> 3 Read Test Counter Enable/Disable Bit
-----> 4 Read Sum Channel Enable/Disable Bit
-----> 5 Read Individual Channel Enable/Disable Bit
-----> 6 Read Mode Run/Halt Bit

-----> 99 Exit Read CSR 00 Menu

Enter Menu Option (1 - 6, 99) :

In response to the prompt, select an option by typing the option number located to the left. Menu selection errors will be flagged and the user will be prompted to re-enter the required option.

2.3.2.2.2.1 Option 1 Read Module ID Bits: -

This option will allow the user to view the current Module ID value stored in the PostAmp Comparator CSR 00 register. If unsuccessful, the user will be notified.

2.3.2.2.2.2 Option 2 Read CLK1 On Bit: -

This option will allow the user to view the current status of the CLK1 On Bit stored in the PostAmp Comparator CSR 00 register. If unsuccessful, the user will be notified.

2.3.2.2.2.3 Option 3 Read Test Counter Enable/Disable Bit: -

This option will allow the user to view the current status of the Test Counter Enable/Disable Bit stored in the PostAmp Comparator CSR 00 register. If unsuccessful, the user will be notified.

2.3.2.2.2.4 Option 4 Read Sum Channel Enable/Disable Bit: -

This option will allow the user to view the current status of the Sum Channel Enable/Disable Bit stored in the PostAmp Comparator CSR 00 register. If unsuccessful, the user will be notified.

2.3.2.2.2.5 Option 5 Read Individual Channel Enable/Disable Bit:
This option will allow the user to view the current status of the Individual Channel Enable/Disable Bit stored in the PostAmp Comparator CSR 00 register. If unsuccessful, the user will be notified.

2.3.2.2.2.6 Option 6 Read Mode Run/Halt Bit: -

This option will allow the user to view the current status of the Mode Run/Halt Bit stored in the PostAmp Comparator CSR 00 register. If unsuccessful, the user will be notified.

2.3.2.2.2.7 Option 99 Exit Read CSR 00 Menu: -

This option will return the user to the previous menu.

2.3.2.2.3 Option 3 Read CSR 00: -

This option will allow the user to view the current value stored in the PostAmp Comparator CSR 00 register. If unsuccessful, the user will be notified.

2.3.2.2.4 Option 4 Exercise CSR 00: -

This option will fully exercise the CSR 00 register circuitry of the PostAmp Comparator module utilizing test sequences from the above menu options. The user will be notified of all occurring errors.

2.3.2.2.5 Option 99 Exit CSR 00 Menu: -

This option will return the user to the previous menu.

2.3.2.3 Option 3 Test CSR 01: -

```
-----> Test CSR 01
       -----------

-----> 1  Set Test Counter Byte in CSR 01
-----> 2  Read Test Counter Byte in CSR 01
-----> 3  Read CSR 01
-----> 4  Exercise CSR 01

-----> 99  Exit CSR 01 Menu

 Enter Menu Option (1 - 4, 99) :
```

In response to the prompt, select an option by typing the option number located to the left. Menu selection errors will be flagged and the user will be prompted to re-enter the required option.

2.3.2.3.1 Option 1 Set Test Counter Byte In CSR 01: -

This option will write a user specified value to the PostAmp Comparator CSR 01 register. Valid Test Counter values are displayed for the user. Invalid Test Counter entries will e flagged and the user will be prompted to re-enter a valid Test Counter value. If unsuccessful, the user will be notified.

2.3.2.3.2 Option 2 Read Test Counter Byte In CSR 01: -

This option will allow the user to view the current Test Counter Byte value stored in the PostAmp Comparator CSR 01 register. If unsuccessful, the user will be notified.

2.3.2.3.3 Option 3 Read CSR 01: -

This option will allow the user to view the current value stored in the PostAmp Comparator CSR 01 register. If unsuccessful, the user will be notified.

2.3.2.3.4 Option 4 Exercise CSR 01: -

This option will fully exercise the CSR 01 register circuitry of the PostAmp Comparator module utilizing test sequences from the above menu options. The user will be notified of all occurring errors.

2.3.2.3.5 Option 99 Exit CSR 01 Menu: -

This option will return the user to the previous menu.

2.3.2.4 Option 4 Test CSR 10: -

-----> Test CSR 10

```
-----> 1  Read CSR 10
-----> 2  Exercise CSR 10

-----> 99  Exit CSR 10 Menu

 Enter Menu Option (1 - 2, 99) :
```

In response to the prompt, select an option by typing the option number located to the left. Menu selection errors will be flagged and the user will be prompted to re-enter the required option.

2.3.2.4.1 Option 1 Read CSR 10: -

This option will allow the user to view the current value stored in the PostAmp Comparator CSR 10 register. If unsuccessful, the user will be notified.

2.3.2.4.2 Option 2 Exercise CSR 10: -

This option will fully exercise the CSR 10 register circuitry of the PostAmp Comparator module utilizing test sequences from the above menu options. The user will be notified of all occurring errors.

2.3.2.4.3 Option 99 Exit CSR 10 Menu: -

This option will return the user to the previous menu.

2.3.2.5 Option 5 Test DAC Load And Read-back: -

```
-----> Test DACs/ADCs
       --------------

-----> 1  Load User Selected DAC
-----> 2  Load all DACs to Full Scale
-----> 3  Load all DACs to Zero
-----> 4  Load all DACs to User Selected Value
-----> 5  Load all DACs, Alternating AA and 55 Values
-----> 6  Load all DACs to Their Corressponding Address
-----> 7  Read User Selected ADC
-----> 8  Read all ADCs
-----> 9  Exercise DACs/ADCs
```

-----> 99 Exit Test DACs/ADCs Menu

Enter Menu Option (1 - 9, 99) :

In response to the prompt, select an option by typing the option number located to the left. Menu selection errors will be flagged and the user will be prompted to re-enter the required option.

2.3.2.5.1 Option 1 Load User Selected DAC: -

This option will write a user specified value to a user specified DAC location on the PostAmp Comparator module. The Mode Halt Bit located in CSR 00 is temporarily enabled for this option. Valid DAC locations and acceptable DAC values are displayed for the user. Invalid DAC locations and/or unacceptable DAC values will be flagged and the user will be prompted to re-enter valid DAC locations and/or DAC values. If unsuccessful, the user will be notified.

2.3.2.5.2 Option 2 Load All DACs To Full Scale: -

This option will write a Full Scale value to all available DACs on the PostAmp Comparator module. The Mode Halt Bit located in CSR 00 is temporarily enabled for this option. If unsuccessful, the user will be notified.

2.3.2.5.3 Option 3 Load All DACs To Zero: -

This option will write a Zero value to all available DACs on the PostAmp Comparator module. The Mode Halt Bit located in CSR 00 is temporarily enabled for this option. If unsuccessful, the user will be notified.

2.3.2.5.4 Option 4 Load All DACs To User Selected Value: -

This option will write a user specified value to all available DACs on the PostAmp Comparator module. The Mode Halt Bit located in CSR 00 is temporarily enabled for this option. Acceptable DAC values are displayed for the user. Unacceptable DAC values will be flagged and the user will be prompted to re-enter valid DAC values. If unsuccessful, the user will be notified.

2.3.2.5.5 Option 5 Load All DACs, Alternating AA And 55 Values:
This option will write an alternating pattern of AA and 55 values to all available DACs on the PostAmp Comparator module. The Mode Halt Bit located in CSR 00 is temporarily enabled for this option. If unsuccessful, the user will be notified.

2.3.2.5.6 Option 6 Load All DACs To Their Corresponding Address:
This option will write to a valid DAC, that DAC's corresponding address. As an example, the DAC residing at address 7F will be loaded with the value 7F. The Mode Halt Bit located in CSR 00 is temporarily enabled for this option. If unsuccessful, the user will be notified.

2.3.2.5.7 Option 7 Read User Selected ADC: -

This option will allow the user to view the current value of a user specified ADC location on the PostAmp Comparator module. The Mode Halt Bit located in CSR 00 is temporarily enabled for this option. If unsuccessful, the user will be notified.

2.3.2.5.8 Option 8 Read All ADCs: -

This option will allow the user to determine that all available ADCs return the expected values previously stored in all the available DACs located on the PostAmp Comparator module. The Mode Halt Bit located in CSR 00 is temporarily enabled for this option. All DAC/ADC pairs are compared allowing a tolerance of +- 1 bit and displaying any discrepencies. If unsuccessful, the user will be notified.

2.3.2.5.9 Option 9 Exercise DACs/ADCs: -

This option will fully exercise all circuitry associated with addressing, reading and writing the PostAmp Comparator DACs and ADCs, utilizing test sequences from the above menu options. The user will be notified of all occurring errors.

2.3.2.5.10 Option 99 Exit Test DACs/ADCs Menu: -

This option will return the user to the previous menu.

2.3.2.6 Option 6 Exercise Secondary Addressing: -

This option will fully exercise the secondary address circuitry of the PostAmp Comparator module utilizing test sequences from the above menu options. The user will be notified of all occurring errors.

2.3.2.7 Option 99 Exit Secondary Addressing Menu: -

This option will return the user to the previous menu.

2.3.3 Option 3 Exercise PostAmp Comparator:

This option will fully exercise all FASTBUS accessable circuitry of the PostAmp Comparator module including all geographical address, secondary address, and DAC/ADC circuitry. The user will be notified of all occurring errors.

2.3.4 Conclusion

The purpose of this software package is to effectively and efficiently test and exercise all FASTBUS accessable circuitry of the PostAmp Comparator module. The user is alerted to any and all errors which may occur during execution of the test software, allowing further investigation of problem areas.

Questions and/or comments concerning these PostAmp Comparator tests should be forwarded to :

Garry R. Moore
Fermi National Accelerator Lab
M.S. 222
P.O. Box 500
Batavia, Il 60510

(708)840-4059
FNAL::MOORE
FNSSD1::MOORE

CHAPTER 3

DELAY ENCODER SINGLE BOARD TEST

3.1 INTRODUCTION

The routine DE_TEST is designed to test the Delay Encoder FASTBUS board of the SSD system using the Test Stand Module (TSM). The paramaters are : the primary address of the TSM, the environemnt identification for the FASTBUS standard routines and a flag, which determines in which mode the routine is called. The routine is written in C and uses MCC8K and VAXC for conditional compilations.

3.2 TEST SETUP

The TSM is used either in a system test, in which it replaces a PC or in a single module test, in which only the DE is tested. In the latter case the TSM controls the DE system signals and is used for the readout of the hits. The format of these hits are different from the ones, which are received from the SE in the system test. This program is designed for the single module test of the DE using the TSM.

The Test Setup is as follows :

- Standard FASTBUS crate and an FSCC in slot 0
- DE
- TSM
- Jumper cards for the auxilliary backplane between DE slot and TSM slot
- Terminating card in slot 13
- Jumper cables on the TSM : J1 - J4 (clock) and J2 - J3 (sync)

3.3 PACKAGES USED

The program uses the following software packages:

- o TSMF.C
- o BUFFER.C
- o FASTBUS standard routines
- o Standard C libraries

3.4 INCLUDE FILES USED

- o Standard C : stdlib.h,math.h,stdio.h
- o FASTBUS : fbpars.h,fberrs.h,fbconstants.h
- o FSCC specific : fscc_ad.h
- o SSD specific : menu_intf.h,tsmreg.h,syst.h

3.5 FEATURES TESTED

The routine DE_TEST tests the full functionality of the DE board. Certain kinds of errors will only be detected, if certain hit patterns are used. One option in the manual menu lets you choose specific patterns. By default, the programm wil generate random patterns.

The test relies on a working TSM board. There is the possibilty to verify the stored data in the TSM memory. TSM.

3.6 SOFTWARE ORGANIZATION

The file DE_TEST.C contains three routines. The main entry point DE_TEST(...) and three support routines, DE_MAKE_HIT_LIST(...), DE_COMPARE_HIT_LIST(...) and DE_PRINT_MENU(). All four routines use global variables. These global variables all start with DE_ and are in capital letters. These variables are initialized at program load time. So if you change variables using the menu option, these changes

are permanent and will be used when the automatic test is called. The seed value for the random number generation is one of the global variables.

3.7 USER GUIDE

The routine DE_TEST can be called in the following modes :

- GO_ON_ERR : An internal buffer is filled with random hits (20 hits in even and 60 hits in odd location). The buffer is organized to reflect the ring buffer located in the DE (0..255). The TSM memory is filled and the TSM transfers to the DE started. The readout loops over all trigger addresses, reads the encoded hits via the TSM and compares it with the generated list of hits. The program does not stop if an error occurs. No error messages are printed. The return value is the number of errors detected. The internal loop counter is set to 10. At every iteration, new random hits are generated.

- STOP_ON_ERR : The steps are the same as above, but if an error occurs, the program will return with a non zero return value, after the first attempt to read an event at trigger address 1. No error messages are printed.

- GET_MENU_ON_ERR : Instead of returning with a non zero return value after detecting an error, the program calls the Menu and waits for commands.

- GET_MENU : In this mode, the Menu is called. No actions are taken. So the buffer is still empty and no FASTBUS operations are performed. The user has full control over the sequence of operations. Error messages will be displayed.

3.7.1 Error Messages

Errors can occur during FASTBUS operations, verification of buffer contents and comparing the predicted hit list with the one received from the TSM.

Errors during FASTBUS operations are reported only if the routine is called in the mode GET_MENU. If verifying the TSM contents gives an error, the local buffer is printed together with the data received from the TSM. If an error is detected during readout, the following error information is given :

```
DE_ERROR : 1, ITERATION : 1, DE_NHIT : 86, LOOP : 0
```

```
   byte          F E D C B A 9 8 7 6 5 4 3 2 1 0
   PBA (000) : 5b01000a0a08022000001039400020 00
    TA (001) : 31c8bf1bfffffedf83c01ffa40f8bfe4
| DE  EXP | DE  EXP | DE  EXP | DE  EXP | DE  EXP | DE  EXP
| yif yif | yif yif | yif yif | yif yif | yif yif | yif yif
* 000 020 * 000 050 * 000 060 * 000 070 * 000 100 * 000 110
* 000 120 * 000 130 * 000 140 * 000 151 * 000 170 * 000 230
* 000 240 * 000 250 * 000 260 * 000 270 * 000 361 * 000 401
* 000 410 * 000 431 * 000 441 * 000 451 * 000 460 * 000 470
* 000 500 * 000 510 * 000 520 * 000 530 * 000 541 * 000 660
* 000 670 * 000 700 * 000 710 * 000 770 * 000 800 * 000 810
* 000 820 * 000 830 * 000 840 * 000 851 * 000 860 * 000 870
* 000 911 * 000 920 * 000 930 * 000 940 * 000 950 * 000 960
* 000 970 * 000 a00 * 000 a10 * 000 a20 * 000 a31 * 000 a40
* 000 a50 * 000 a60 * 000 a70 * 000 b00 * 000 b11 * 000 b20
* 000 b31 * 000 b40 * 000 b50 * 000 b60 * 000 b70 * 000 c00
* 000 c11 * 000 c31 * 000 c40 * 000 d00 * 000 d10 * 000 d20
* 000 d30 * 000 d40 * 000 d50 * 000 d70 * 000 e01 * 000 e30
* 000 e60 * 000 e70 * 000 f01 * 000 f11 * 000 f31 * 000 f41
* 000 f50 * 000 f61

PBA : Previous Bucket Address
TA  : Trigger Address
DE  : data received from the TSM
EXP : data expected
f   : RF flag (Previous Bucket Flag)
y   : Byte number (0..f)
i   : Bit number  (0..7)

(The star indicates that this hit was not found in the received list)
```

The current implementation verifies, that all hits predicted are actually present in the received hit list. It will flag extra hits too.

3.7.2 MAIN MENU

```
 ***** WARNING : Buffer not filled ***** : use l) first
----- DE/TSM Test Menu -----
     a) Set Readout delay (0 [msec])
     b) Initialize TSM, fill event memory and verify
     c) Select Trigger Address (1) and read
     d) Loop over all trigger addresses
     e) Random fill and loop over all trigger addresses
     f) Set maximum error count (10)
     g) Set number of hits (even:20,odd:60) and fill buffer
     h) Print this text          i) Read buffer from file
     j) Write buffer to file     k) Accept hits from PBA (T)
     l) Generate new random buffer and do b
```

```
m) Set loop counter for c,d and e (10)
n) Modify buffer                o) Print buffer contents
p) Toggel menu print out        q) FIFO block transfer enable (T)
r) Print Error Buffer           s) Clear Error Buffer
t) Set Trigger Address Correction (0)
u) Clear Write                  v) Verify TSM memory
z) Scramble buffer(S->J)        w) Unscramble buffer (J->S)
x) Exit Test (Last Iteration count : 0, Errors : 0)
Enter command ( HELP : h ) :
```

3.7.2.1 Item A) -

You can introduce a delay between iterations.

3.7.2.2 Item B) -

TSM_FILL in TSMF is called. Reset TSM, copy the event buffer to the TSM and read the buffer back in order to verify it. Only the processor controlled block transfers are used. If the block transfer via the FIFO is enabled, it will be disabled before reading and enabled after reading.

3.7.2.3 Item C) -

You can select a specific trigger address and the encoded hit data will be compared to the predicted list of hits. The number of times this operation is executed can be set in item m). The loop can be interrupted by pressing any key of the kerboard.

3.7.2.4 Item D) -

Similar to c), but the routine loops over all trigger addresses, starting with 1 and ending with 0. The number of times this operation is executed, can be set in item m). The loop can be interrupted by pressing any key of the keyboard.

3.7.2.5 Item E) -

The buffer is filled with a random pattern. The number of hits can be selected in item g). The program loops over all trigger addresses. After each loop, the buffer is filled with new random hits.

3.7.2.6 Item F) -

Set the maximum number of errors. The program will stop if the number of errors reaches this maximum.

3.7.2.7 Item G) -

Set the number of hits for random filling. The number of hits for even addresses and odd addresses seperately will be set, the buffer filled and copied to the TSM.

3.7.2.8 Item H) -

The menu is printed.

3.7.2.9 Item I) -

Instead of filling buffers with random numbers or specific patterns, you can read a prepared buffer from a file. This is only useful, if the program runs on the VAX, communicating with the FSCC via Remote Procedure Execution (RPX). When running on the FSCC, the input will be via the serial line.

3.7.2.10 Item J) -

Writes the current buffer to a file. See Item i)

3.7.2.11 Item K) -

This determins, whether hits in the previous bucket (trigger address - 1) will be accepted for the generation of the hit data. Jumpers on the DE must be set accordingly. By default hits in the previous bucket will be accepted.

3.7.2.12 Item L) -

Generate new random buffer, fill the TSM and verify.

3.7.2.13 Item M) -

Set the outer loop counter for c), d) and e). If a key on the keyboard is pressed, the loop is aborted.

3.7.2.14 Item N) -

This option calls a submenu for the filling of the buffer. This submenu is part of the package BUFFER.C.

```
 0  : fill selected position
 1  : fill with row of pattern
 2  : fill with 5555...
                AAAA...
 3  : fill with FFFF...
                0000...
 4  : fill with PC counter -fixed- pattern
 5  : fill with PC counter -running- pattern
 6  : fill with random pattern
 7  : fill with repeated N raw pattern
                LLLL...
                WWWW...
                KKKK...
 8  : display selected positions
 99 : exit
```

3.7.2.15 Item O) -

Output the contents of the buffer at a certain address.

3.7.2.16 Item P) -

En/disable the printout of the menu.

3.7.2.17 Item Q) -

Change the FASTBUS environment to en/disable the block transfer read via the FSCC fifo. Block transfers via the FIFO are enabled by default.

3.7.2.18 Item R) -

Print the error buffer. A circular buffer accumulates the last 20 data errors.

```
 Error : 1 , Iteration : 1 , NHIT : 86
   byte          F E D C B A 9 8 7 6 5 4 3 2 1 0
   PBA (000) : 5b01000a0a08022000001039400020 00
    TA (001) : 31c8bf1bfffffedf83c01ffa40f8bfe4
-------------------------------------------------------
 Error : 0 , Iteration : 0 , NHIT : 0
   byte          F E D C B A 9 8 7 6 5 4 3 2 1 0
   PBA (000) : 00000000000000000000000000000000
    TA (000) : 00000000000000000000000000000000
```

3.7.2.19 Item S) -

Clear the error buffer.

3.7.2.20 Item T) -

Define an offset from the selected trigger address, which will be subtracted from the trigger address and sent to the TSM. This allows for non synchronized transfers of hits to the DE.

3.7.2.21 Item X) -

Exit the menu and return to the main program.

3.7.2.22 Typical Session -

The following is a typical sequence of events, which is necessary to do the test manually with random hits:

- o Set readout delay a)
- o Set maximum number of errors f)
- o Set maximum loop count m)
- o Set number of hits g)

- Start readout loop e)

The test will be aborted in the following cases:

- The maximum loop count is reached
- The maximum error count is reached
- The user hits a key on the keyboard

CHAPTER 4

SEQUENCER SINGLE BOARD TEST

4.1 INTRODUCTION

The testing of the Sequencer is broken into three separate types of test. The initial low level test are simple peek and poke at FASTBUS registers to fix miswires or broken parts. The second set of test verify that the module works without errors for extended periods, that is test that loop and compare data. The third set of test are high level tests which involve system integration. These last test require that most of the hardware is working and that system test modules are present and working.

The following test are available as low level tests and as tests with looping and compare of data. The final higher level system tests are not described in this document.

4.2 TEST HARDWARE SETUP

The Sequencer stand alone test which are described here require that a sequencer test module be present and working. The Sequencer test module is composed of two parts.

4.2.1 Sequencer Fastbus Test Card

This card resides in a Fastbus slot and emulates the MTC interface for the sequencer and also provides control switches to control the auxiliary cards emulation of the Delay encoder.

Switches are present which select the start and stop strip pattern for generation of events. 4 switches to select a start pattern number and 4 switches to select a stop pattern number. There are also 12 switches which enable/disable which delay encoder channels will have hits. There is a set of 6 switches to select the event size. The last set of 8 switches selects the event address.

There are buttons to generate sync errors, reset, manual or automatic address valid, write enable, sync, and a reset of the test card.

There are led for

1. ERR_TO_MTC. indicates that the sequencer has set the error signal.

2. WR_ENBL. indicates that the sequencer has received and sent the write enable.

3. SYNC. indicates that the sequencer has received and sent the sync signal.

4. Reset. indicates that the sequencer has received and sent the reset signal.

5. ADR_VAL. indicates that the sequencer had received and sent the address valid signal.

6. +5 power and -5 power. indicate presence of power to test card.

7. Error count. is a count of errors detected by the auto test hardware.

8. TEST count. is a count of transfers completed with or without errors.

4.2.2 Sequencer Auxiliary Test Card

This card resides on the SSD aux backplane. It received power from the backplane as well as from a cable to the fastbus backplane. This card emulates 12 delay encoders and also test the auxiliary interface. (See attachment for more detailed description.)

4.2.3 Cables

There are 5 cables which must be connected before testing can begin.

1. CLK 53MHZ. This is generated by the FASTBUS test card and a cable must be connected between the 53MCLK lemo of the test card and the 53MCLK lemo of the Sequencer.

2. SYNC. This signal is generated by the FASTBUS test card and a cable must be connected between the SYNC lemo of the test card and the SYNC lemo of the Sequencer.

3. MTC interface 34pin ribbon. This cable must connect between the FASTBUS test card and the Sequencer. It has all the signals which are normally generated by the MTC. (ie. ev_adr(0:7), Write enable, address valid, reset, ...)

4. Test card interface. This 40 pin ribbon connects control signals and status between the FASTBUS test card and the Auxiliary test card.

5. Aux power. This cable provides additional power for the auxiliary test card for the standard fastbus backplane.

4.3 TEST PROGRAM FILES

The Sequencer test program is integrated into the SSD single board test program. The Sequencer test are part of a program SE_TEST.

There are a number of include files which must be present in SSD_DIAG libraries. The ones specific to the sequencer are:

4.3.1 SEERS.H

This is a definition of the error bit fields which get set during the test program operation. They are used to display status but are mainly for debugging difficult to understand errors.

4.3.2 SEREG.H

This is a definition of the sequencer register bit fields. Some of the fields are specific for building event data for checking errors. The bit field definitions simplify the test programs setting and clearing of bits and display of status.

4.3.3 SEQPAT.H

This is a definition of an array which contains fixed patterns which are present in the sequencer auxiliary test card

for generation of delay encoder events.

4.3.4 CURCON.H

This is a definition of cursor control bit strings which simplify the cursor control commands.

4.3.5 CURCON.C

This is a set of basic cursor control commands. (i.e. clear screen and set cursor position.)

4.4 SEQUENCER ACCES THROUGH SINGLE BOARD TEST MENU

Before starting the tests the Sequencer should be located in slot 13 (D)

of the crate. Use the SET PRIMARY ADDRESS option of the Single Board TEST (SBT) menu to confirm the address value.

The sequencer test can be executed in automatic mode or in manual test mode. To enter automatic mode the test card switches must first be set to enable all encoder channels and generate 2 word events with start and stop pattern 0 and 1. Select the SBT option for Default test Menu to enter

auto mode. Set desired mode (ie. stop on error) then execute test. If errors occur the test program will return an error and stop or go to a test menu or continue.

To operate in manual mode select the SBT menu option for Sequencer menu.

You will receive the sequencer test menu.

4.5 SEQUENCER MENU

```
----- Sequencer Main Menu -----
a)  Single Transfer menu
b) Block Transfer menu
y) Sequencer Auto test
g) Sequencer menu
i) Set loop counter ,se_nbr_loops
m) Set Primary Address (PAD) (hex value: *se_pad)
```

```
n) SAD (hex value: *se_sad)
r) display read buffer
s) load write buffer
w) Map FASTBUS crate
u) Show or reset Status Summary, TOTAL errors= se_ERS_total
x) Exit Program

Enter se_command:
```

In response to the prompt select an option by typing the letter on the left. The response is not case sensitive. In some cases another submenu will be displayed.

4.5.1 I) Set Loop Counter ,se_nbr_loops

In the main menu display option I shows the value of se_nbr_loops which is the number of times a test will be executed. Select option I and you can change this value. The following prompt will be displayed.

```
Enter loop value (dec) :
```

If you type a value of 0 the tests will loop until a key is pressed to stop them.

4.5.2 M) Set Primary Address (PAD) (hex Value: *se_pad)

Main menu option M shows se_pad, which is the value of the FASTBUS primary address which will be used for the tests. The following prompt will be displayed.

```
Enter PAD (in hex):
```

4.5.3 N) SAD (hex Value: *se_sad)

Option N shows the value of the FASTBUS secondary address which will be used or was used. The following prompt will be displayed.

```
Enter SAD (in hex):
```

4.5.4 A) Single Transfer Menu

In response to the selection of option A the subprogram process_se_single_transfer(se_pad,envid,action) will be called

and the following menu will b displayed.

```
----- Single Transfer Menu -----
a) FRD
b) FWD
c) FRC
d) FWC
n) Print read data after every loop ,print_data
o) Inc SAD after each loop ,inc_st_sad
p) Inc write data after each loop ,inc_st_data
x) Exit this menu
Enter se_command:
```

4.5.4.1 A) Single_transfer_read_data -

The FSCC will do a FASTBUS read of data space using the previously selected FASTBUS primary address (se_pad), secondary address (se_sad) and will loop for se_nbr_loop times. The data value read will be displayed each time. The option of incrementing data or secondary address toggled by option P or O respectively.

If errors are detected in the FASTBUS return status the following message will be printed if in GET_MENU mode.

FASTBUS error at PAD: se_pad SAD: se_sad

The following message will be printed with each FRD loop.

SAD (in hex): se_sad Value (in hex): se_read_data

4.5.4.2 B) Single_transfer_write_data -

Selecting the FWD option will provide the following prompt.

Enter data value (in hex):

The FSCC will do a FASTBUS write to data space using the previously selected FASTBUS primary address (se_pad), secondary address (se_sad) and will loop for se_nbr_loop times. The data value you select will be written each time with the option of incrementing data or secondary address selected by option P or O respectively.

If errors are detected in the FASTBUS return status the following message will be printed if in GET_MENU mode.

FASTBUS error at PAD: se_pad SAD: se_sad

4.5.4.3 C) Single_transfer_read_control -

The FSCC will do a FASTBUS read of control space using the previously selected FASTBUS primary address (se_pad), secondary address (se_sad) and will loop for se_nbr_loop times. The data value read will be displayed each time. The option of incrementing data or secondary address toggled by option P or 0 respectively.

If errors are detected in the FASTBUS return status the following message will be printed if in GET_MENU mode.

FASTBUS error at PAD: se_pad SAD: se_sad

The following message will be printed with each FRD loop.

SAD (in hex): se_sad Value (in hex): se_read_data

4.5.4.4 B) Single_transfer_write_control -

Selecting the FWC option will provide the following prompt.

Enter data value (in hex):

The FSCC will do a FASTBUS write to control space using the previously selected FASTBUS primary address (se_pad), secondary address (se_sad) and will loop for se_nbr_loop times. The data value you select will be written each time with the option of incrementing data or secondary address selected by option P or 0 respectively.

If errors are detected in the FASTBUS return status the following message will be printed if in GET_MENU mode.

FASTBUS error at PAD: se_pad SAD: se_sad

4.6 U) SHOW STATUS DISPLAY

Option U in main menu shows se_ERS_total, which is the total errors which have already occurred. If selected STA_SUM(action,se_pad,envid) will be called and the following status display will be shown.

Summary Screen
PAD: se_pad SAD: se_sad
The Executing Test is in Reverse graphic
se_CSR000_tst errors : se_ERS_000
se_CSRC00_tst errors : se_ERS_C00
Block Word counter tests, se_CSRC10_tst errors :se_ERS_C10

```
Clock delays tests, se_CSRC11_tst errors :se_ERS_C11
Overflow status register, se_CSRC12_tst errors : se_ERS_C12
Encoder event fifo, se_ECF_tst  errors : se_ERS_ecf

read: %x write: %x\n",se_read_data,se_write_data);
se_data_rec: # se_ignore_rec: # se_rf_rec: # se_ovf_rec: #
se_strip_rec: # se_id_rec: #
Wd cnt exp: # rec: # Blk cnt exp: # rec: #
errors this  test:  #  SubTotal  tests  :  #       Total data errors: %x
Total errors : se_ERS_total Total tests : se_TOT_tst
on last data error read: se_read_data write: se_write_data


        Do you wish to reset the status? (Y) or (N):
```

4.7 W) MAP THE FASTBUS CRATE

Selecting this option will call the following program. map_crate(action,se_pad,envid). The FSCC will sequentially address each slot at CSR space with secondary address 0. Only response with FASTBUS status FENORM will display the following:

```
        Module with ID: se_read_data responds to PAD:  *se_pad
```

When complete the FSCC will prompt with:

```
        Press Q to return to menu
```

To this you must type a key followed by return to return to the main menu.

4.8 R) SE_DISPLAY_READ_BUFFER()

This option will display the contents of the read buffer. The following will be displayed

```
        Last buff_loc used was : se_buff_loc
        Enter starting offset (in hex):
```

The se_buff_loc is the last used read buffer location. You select a buff location value to display. Then the following will be printed.

```
        %08lx     %08lx  %08lx  %08lx  %08lx
```

This is a block of values found at the selected buffer locations followed by the following prompt.

```
        Continue?
```

If you type Y the display will again show more values starting at the next location. A N response will abort the display and return to the main menu.

4.9 S) SE_LOAD_WRITE_BUFFER()

Selection of option S will display the following menu.

```
----- Display Menu -----
a) Clear write buffer
b) Load write buffer with bit pattern
c) Load write buffer with ascending integers
d) Load write buffer with long words
e) Load write buffer with '55555555,AAAAAAAA'
f) Copy read buffer to write buffer
w) Display write buffer
y) Load write buffer with datasel[i] pattern
h) Load datasel[i] with write buffer pattern
x) exit
Enter se_command:
```

4.9.1 A) Clear Write Buffer

This option if selected will clear the write buffer from location 0 to se_buff_length as set by the global at beginning of the program.

4.9.2 B) Load With Bit Pattern

This option will produce the following prompt

```
Enter bit pattern (in hex):
```

The program will then fill the write buffer from location 0 to se_buff_length with the selected pattern.

4.9.3 C) Load Write Buffer With Ascending Integers

This option will load the write buffer with ascending values starting with location 0 with the value 0 until se_buff_length is loaded with value se_buff_length.

4.9.4 D) Load Write Buffer With Long Words

This option will prompt with the following.

Starting offset (in hex):

You select the buff location then the following is displayed.

i : se_wrbuff(i) New value (exit with '.') :

Where i is the buff location and you select a value or type . to terminate this option. If a value other than . is selected the next location will be displayed and a new value can be entered.

4.9.5 E) Load Write Buffer With '55555555,AAAAAAAA'

Loads the write buffer with the pattern 55555555 to location 0 then AAAAAAAA to location 1 then repeats the sequence until se_buff_length is reached.

4.9.6 F) Copy Read Buffer To Write Buffer

Copies the contents of the read buffer to the write buffer.

4.9.7 Y) Load Write Buffer With Datasel[i] Pattern

Loads the write buffer with the datasel pattern which was previously set.

4.9.8 H) Load Datasel[i] With Write Buffer Pattern

Loads the datasel buffer with the contents of the write buffer. The datasel buffer is only a 16 location array which is used when checking for expected events. This is meant to duplicate the patterns which might be set by hardware test module sequences.

4.9.9 W) Display Write Buffer

To display the contents of the write buffer select option W. The following will be displayed

Last buff_loc used was : se_buff_loc

Starting offset (in hex):

To this you select the buffer location which you wish to display and the following will be displayed

```
se_wrbuff[loc],se_wrbuff[loc+1],se_wrbuff[loc+2],se_wrbuff[loc+3]
```

After the array of data has been displayed the following prompt will be given.

Continue?

To this if you type Y more data will be displayed but if you type N the routine will be terminated and the program will return to the main routine.

4.10 Y) SEQUENCER AUTO TEST

This option in the main menu will call the following routine.

```
process_seqauto(action,se_pad,envid)
```

This routine will in turn call the following routines.

```
sqt_000(action,se_pad,envid);      /* Process CSR0 tests */
sqt_C10(action,se_pad,envid);      /* Process CSR C0000010 tests */
sqt_C11(action,se_pad,envid);      /* Process CSR C0000011 tests */
sqt_C12(action,se_pad,envid);      /* Process CSR C0000012 tests */
sqt_ecf(action,se_pad,envid);      /* Process Encoder fifo tests */
```

4.11 G) SEQUENCER MANUAL TEST

This option in the main menu will call the following subroutine.

```
process_seqtst(action,se_pad,envid)
```

The following sub menu will be displayed.

```
----- Sequencer Manual test Menu -----
a) Geographic addressing
b) NTA load and increment
c) Set start and stop pattern datasel[i],se_start_pat,se_stop_pat
d) Set Valid Delay encoder channels dvs_number
f) Set Event data size  se_ev_data_size maxbytes= se_maxbytes
h) CSR0 tests
i) CSR C0000000-C000000B Ram tests
j) CSR C0000010 tests
```

```
k) CSR C0000011 tests
l) CSR C0000012 tests
m) Encoder fifo tests
o) Dump registers
p) Set ignore error flag %x \n",se_ignore_err
q) Display error in compare read and write buffers
x) Exit this menu
Enter se_command:
```

4.11.1 A) Geographic Addressing

Selection of this option will cause geographic data space addressing of the sequencer to be tested using the address selected by se_pad. The FASTBUS operation executed will be a primitive address cycle without a secondary address or a data cycle followed by a address cycle release if an AK was returned. The test will loop depending upon se_nbr_loop. The following messages will be printed if there are fastbus errors.

FASTBUS error at PAD: se_pad

If there is an error in the release of the FASTBUS the following message will be printed.

FASTBUS address release error at PAD: se_pad

4.11.2 B) NTA Load And Increment

This test will address the module at se_pad and perform a secondary address write followed by a secondary address read. When errors are found the following message will be printed.

FASTBUS error at PAD: se_pad SAD: se_sad

If the value read is not the value written to the NTA then the following message will be printed.

Bad NTA error at PAD: se_pad SAD: se_sad Read: se_bufspec

4.11.3 C) Set Start And Stop Pattern Dataset[i],se_start_pat,se_stop_pat

This option will allow you to change the value of the start and stop test pattern for the strip field when checking for events in the ECF test routine. The following messages will be printed.

Starting pattern address (in hex):

Stop pattern address (in hex):

The normal response to these prompts is the switch setting of the Sequencer FASTBUS test card for the respective settings of start and stop pattern.

4.11.4 D) Set Valid Delay Encoder Channels Dvs_number

If selected this option will prompt for which encoder channels are enabled. This data will be used when checking incoming events for the correct data. The normal response to the prompt is the setting of the Sequencer FASTBUS test card setting for the individual channel enables 12 switches or values 0 to 4096. Each bit position is a separate channel enable. The prompt received is as follows.

Set encoded Valid Delay encoder channels:

4.11.5 F) Set Event Data Size Se_ev_data_size Maxbytes= Se_maxbytes

To set the size of the expected event select option F. The following prompt will be displayed.

Set Delay encoder event size:

The response should be the same setting as is on the Sequencer FASTBUS test card switch for event size. It is a byte count of each of the expected Delay encoder channels. Each channel will have the same byte count. Only one value will be expected. Total event size and byte count will be calculate using this number with the number of channels enabled and ignore words.

4.11.6 H) CSR0 Tests

When option H is selected the following sub test will be executed. int sqt_000(action,se_pad,envid) This test will read and write patterns to the valid bits of CSR0 and check if errors occur. If an error occurs the int se_erinc(action,se_pad,envid) subprogram will be called to increment the error count and then the following display programs will be called.

int se_display_subtest(action,se_pad,envid) Which will print the following message to the screen at a fixed screen line.

errors this test: # SubTotal tests se_sub_TOT_tst: #

int se_display(action,se_pad,envid) Will then be called and the status summary of STA_SUM(action,se_pad,envid) will be displayed. The test will then terminate by calling int se_endtest(action,se_pad,envid) and prompting withe the following:

Press return to exit:

To this prompt a key must be pressed followed by a return and the program will then return to the sequencer manual test sub menu.

4.11.7 I) CSR C0000000-C000000B Ram Tests

This test will execute as above in test H with the exception that tests will be performed on the CSRC locations of the Sequencer. By calling the int sqt_C00(action,se_pad,envid) test program.

4.11.8 J) CSR C0000010 Tests

This test will execute as above in test H with the exception that tests will be performed on the CSRC-10 locations of the Sequencer. By calling the int sqt_C10(action,se_pad,envid) test program.

4.11.9 K) CSR C0000011 Tests

This test will execute as above in test H with the exception that tests will be performed on the CSRC-11 locations of the Sequencer. By calling the int sqt_C11(action,se_pad,envid) test program.

4.11.10 L) CSR C0000012 Tests

This test will execute as above in test H with the exception that tests will be performed on the CSRC-12 locations of the Sequencer. By calling the int sqt_C12(action,se_pad,envid) test program.

4.11.11 M) Encoder Fifo Tests

This test will execute as above in test H with the following exceptions. The tests will be performed on the data space 0 locations of the Sequencer. The program called will be int sqt_ecf(action,se_pad,envid). This routine must be set up if not in Stop_on_err mode or Get_menu_on_err mode. The setup information comes from responses to prompts in this section of the program and those previously selected for configuring the event. The Sequencer FASTBUS test card and the Sequencer Aux test cards must be attached or these test might be confusing. The following prompt will be typed.

Do you want to reset the Sequencer (Y) :

If Y is selected routine int sqt_rst(action,se_pad,envid) will be called and the bit 30 (reset) will be set and written to CSR0 of the sequencer at se_pad. If not selected Y then be careful about how the previous values of CSR registers were set.

Then the following will be printed.

Read event with single word transfers (Y) :

The program can read the Sequencer event data in block transfer mode or single transfers based upon this response.

After the above responses have been made the program will then call the sqt_ecf_buff(action,se_pad,envid) routine which will prompt for the following.

Do you wish to load ID ram with incrementing pattern? (Y) or (N):

This will load the ID ram of the Sequencer if Y is selected.

Do you wish to set the RF flag? (Y) or (N):

This will set the RF flag in the expected event if Y is selected.

Do you wish to disable the aux fifo? (Y) or (N):

Do you wish to disable the FASTBUS fifo? (Y) or (N):

The enable bits of the Sequencer buffers will be set depending on the response to the above prompts.

Events will now be read, when ready press G for go :

When ready to start taking events press G return. The test card should at this time be armed and data sent to the Sequencer. Pressing G will cause the program to read the Sequencer for the expected events and compare the data calling normal routines as in the previous routine H. That is status will be displayed with

error checking.

4.11.12 0) Dump Registers

An additional status display is present which shows the registers. The option causes int sqt_dr(action,se_pad,envid) to be called which will read each register and display the status in the following format.

```
----- Sequencer Register dump -----


CSR0 errs: wr: # rd: # cmp: #
CSRC00 errs: wr: # rd: # cmp: #
CSRC10 errs: wr: # rd: # cmp: #
CSRC11 errs: wr: # rd: # cmp: #
CSRC12 errs: wr: # rd: # cmp: #

CSR0 : ID: # OT: #  FEF: #  AIF: #

CSR10 : Blockcount: # Wordcount: #

CSR11 : Clk1 delay: # Clk2 delay: #

CSR12 : Sync err: #  Encoder fifo overflow: #

Valid Delay encoder channels #

start and stop pattern datasel[i] # , #

Event data size: # maxbytes: # maxwds: #
Fastbus status:
```

The following prompt will then be given to select more event data display.

```
Display event info Y ?
```

If Y is the response then the following will be typed.

```
se_ev_idbuf:
      se_ev_idbuf[i],se_ev_idbuf[i+1],se_ev_idbuf[i+2],se_ev_idbuf[i+3]);

se_event_data.bit: FLAG: se_event_data.bit.flag
SE_EVENT_DATA.bit: ID: #    , strip: #    , RF: #
se_event_data.word: se_event_data.word
se_event_last.bit.blockcount: se_event_last.bit.blockcount

press key to return
```

In response to the prompt a key press followed by return must be

pressed.

4.11.13 P) Set Ignore Error Flag ,se_ignore_err

Will prompt with the following.

Do you wish to ignore all errors?

If you answer Y the ignore errors will be selected and much display information will not be displayed errors will not be checked in sub programs. If you answer N then error checking will again be enabled.

4.11.14 Q) Display Error In Compare Read And Write Buffers

This routine will compare the read and write buffers for differences and display those differences. It will prompt of the starting location in the buffers to compare.

Starting offset (in hex):

Then it will display the data in the following format.

loc: i read: se_rdbuff[i] write: se_wrbuff[i]

After displaying a group of differences it will prompt

Continue?

A response of Y return will cause more compares to be done until all the buffer has be compared. Then the only valid response will be a N return.

4.12 B) BLOCK TRANSFERS

Selection of this main menu option will cause the process_se_block_transfer(se_pad,envid,action) subprogram to be called and the following submenu to be displayed.

```
----- Block Transfer Menu -----
a) FRDB
b) FWDB
g) FIFO read enabled (%c) COPYEN (%c)\n",fifo_read,coen
h) PIPElined read enabled (%c) : %d nsec\n",pipe_read,pipe_time
i) Get number of bytes transferred (%d) \n",number_of_bytes
j) Reset FIFOs
l) Reset FIFOs in each loop (%c)\n",fifo_reset
```

```
o) EOBINT enabled (%c)\n",eoen
p) Number of long words to transfer (dec value: %d)\n",se_maxwds
n) Print read data after every loop (%c)\n",print_data
x) Exit this menu

 Enter se_command:
```

4.12.1 A) FRDB

Will initiate a block transfer loop which will repeat se_nbr_loop times. if errors occur they will be displayed as follows.

FASTBUS error at PAD: se_pad SAD: se_sad

Return data will be displayed if enabled by option N. with format:

PAD : se_pad SAD : se_sad

loc: i read: se_rdbuff[i]

When the test loops are complete the following prompt will be displayed.

Continue?

The only valid response is a key press followed by a return.

4.12.2 I) Get Number Of Bytes Transferred

Will display the number of bytes transfered in the last command.

Number of Bytes transferred : number_of_bytes

4.12.3 P) Process "set Number Of Words To Transfer" Commands

Sets the number of words to transfer in the block transfer command.

Enter maximum number of long words to transfer (in decimal):

4.12.4 G) Process FIFO Command

Will select use of FIFO in FSCC and set copy flag for data compare.

Copy enabled (T/F) :

4.12.5 H) Process PIPE Command

Will select the pipeline transfer option for block transfers and set the speed parameter.

pipelined read (100/200/400) nsec:

For further information, please contact:

M. Larwill, B. Demaat

CHAPTER 5

MTC SINGLE BOARD TEST

5.1 TEST DEFINITION

5.1.1 Modes Of Operation (Ref. MTC HARDWARE DISCRIPTION , October 19, 1990)

1. Test : External trigger requests are disregarded , triggers are generated by software (FB triggers). Write counter is shut down and the read address is controlled by the trigger offset (0 - trig. offset) Wait is always set in this mode.

2. Calibration : Each external trigger generates N (switch setable) consecutive read addresses. Address valid is generated independent of the presence of any error .

3. Run : Each external trigger generates a read address, which is stored in the FIFO if D/E READY is not present. The FIFO depth is programmable and WAIT is set if the number of FIFO stages is greater equal than that. If more triggers occur then FIFO OVERFLOW ERROR will be generated. If the read address (obtained by adding the offset determined by the calibration procedure to the reference D/E write counter of the MTC) is close by a programmable amount to the current write address in the D/Es a MEMORY OVERWRITE ERROR will be issued. Any of the above errors , or TRIGGER PHASE and DE errors will stop external triggers and FIFO reads (addr_valid) until it is cleared

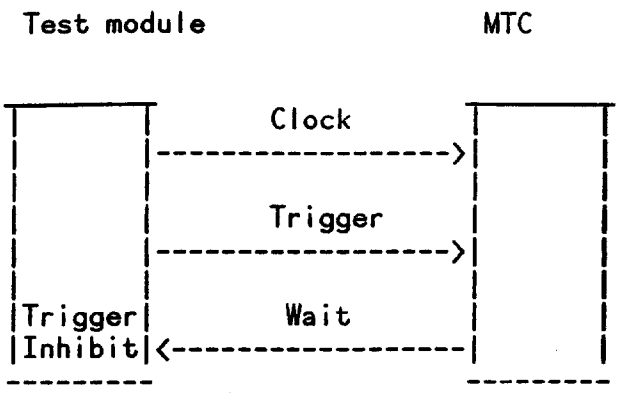
5.2 FEATURES OF THE MODULE THAT ARE TESTED :

1. Fastbus interface
2. Fifo read address
3. Fifo overflow
4. Clock phase adjustment
5. Memory overwrite
6. Wait
7. Calibration

5.3 HARDWARE REQUIREMENTS

A standard Fastbus crate for the MTC and a standard NIM crate for the MTC test module. The test module provides the clock input and the external trigger input for the MTC. Since there are no external readings durring the test , the module will issue a Memory Overwrite Error after (256 - Memory window set) clock cycles. The external trigger rate from the test module was choosen to be 1/6 clock cycles in order to fire enough triggers to test the whole FIFO depth before the error is issued. The triggers from the test module are inhibited with the WAIT output of the MTC , to prevent FIFO overflow error. The phase of the trigger must be adjusted to be synchronized with respect to te clock . This is done in the following way : the front pannel provides a trigger monitor and a trigger window signals (NIM). The possition of the leading edge of the trigger signal must be set in the middle of the trigger window delaying the external trigger with respect to the clock. Also, the window width can be adjusted from the front pannel switch . A working module should not issue a trigger phase error (front pannel led) when the write enable is set if the phase is properly adjusted (the write enable must be set to see a trigger phase error).

No more front panel connections are required and the presence of an ENCODER READY from the ribbon cable connector will cause the test to fail.



5.4 TEST DESCRIPTION.

According to a) , b) , c) the test procedures are defined :

1. Fastbus interface : Write and read to (from) CSR10 - CSR13 set and reset flags (bits) and read statuses of CSR0.

2. Fifo read address :

```
Reset
Error flag reset
Test mode
Write enable
Trigger address offset to 256 - N
FB  trigger
FB  encoder ready
Read CSR10
Compare
Reset
```

3. Fifo overflow :

```
Reset
Error flag reset
Test mode
Write enable
Set trigger pipeline depth to 7
While not overflow
  Set trigger address offset
  FB trigger
End While
Check number of triggers generated
Loop on trigger number
   ad back with Encoder Ready and
   eck address
```

```
End Loop
Reset
```

4. Clock phase adjustment : Adjust clock phase and delay the trigger over a range of values , checking for trigger phase error (only in manual test , for automatic tests make sure that trigger remains in sync)

5. Memory Overwrite :

```
Reset
Error flag Reset
Test mode
Write Enable
Loop on the 255 possible read addresses
 Error flag  Reset
 Set trigger address offset
 Assert FB trigger
 Check Memory Overflow
 If not when address agrees with the Dip Switch
                report it
 Assert  FB trigger
 Read and compare address
End Loop
Reset
```

Here you have to set the switch to the hardcoded value 8 (see last MTC TESTS section) or you can change the default during the initiallization time , to agree with your set value.

6. Wait :

```
Reset
Error flag reset
Loop over fifo depth
  Set fifo depth
  Set write enable
  Loop until wait or error or timeout
  When wait or error triggers(external) are inhibited
  Set test mode
  Assert Fb encoder ready until fifo is empty
  Compare with depth or if error report
End Loop
Reset
```

7. Calibrate :

```
Reset
Error flag reset
Set trigger pipeline to 1 so wait will inhibit all
 but the first external trigger
Set calibration
Set Write enable
```

```
Loop until Wait
Set Test mode
Loop
 Assert Fb encoder ready
 Fill array
 Break if fifo empty
End Loop
Compare and check array, report errors
Reset
```

Test mode is required when reading back external triggers in Run mode in order to prevent the blocking of addr_valid due to Memory overwite error. For the same reason Reset is required after every test completion. For every FB write the corresponding read is performed in order to check the FB interface. During the initiallization the parameters that correspond to dip switches (memory overwrite window , number of calibration triggers) must be matched with the hardware values, from the main menu (option d).

Also depending on the mode of operation (interactive or auto) you can (or cannot in auto) alter the values of the parameter of each test (i.e trigger offset , pipeline depth) , to run on your prefered values.

5.5 SOFTWARE DESCRIPTION

MTC_TEST.C program compiles ,links and runs with the standard SSD procedures , under the control of SSD_MAIN_MENU , either on the VAX or on the 68020 of the FSCC (FNSSD1 node). The interface with SSD_MENU is via a call to the function : mtc_test(*primary_address,*envid,*action) where action is defined according to the SSD standard (section 1.3.11) , so you can either call the full test or the main menu of the test. The function returns the longword MTC_er_log.word , which is defined in the include file ssd_diag$library:mtcdefn.h (== PASS when everything is fine)

```
MTC_er_log;                    /* Used to pack error status */
```

Bits 0 - 11

Error Flag
Write Enable
Calibration Mode
Test Mode
Fifo Pipeline Depth
Trigger Address Offset
Clock
Fifo Read Address procedure

Fifo Overflow procedure
Calibration procedure
Wait procedure
Memory Overwrite procedure

5.5.1 Program Structure

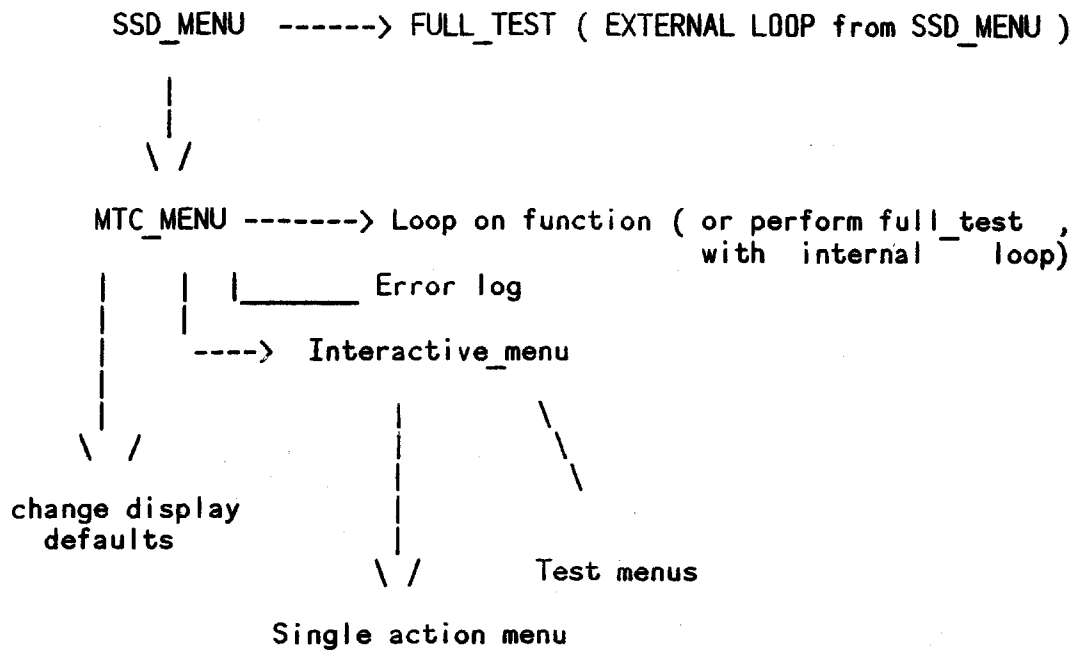
There are 4 different types of functions according to the task they perform :

1. Single action
2. Test procedure
3. Menu
4. Display and error log.

The single action functions perform all the operations that are provided from the Fastbus Interface . In each of them succesfull completion is checked and a status is returned. In the test procedure functions the previous category functions are combined to perform the different module tests . The full test calls sequentially the test procedures and does the error loging.

The menus (see Appendix),give the capabillity to perform each test individually , if you enter mtc_test in menu driven mode , either interactively (step by step) or in a loop using the standard features of SSD_DIAG , stop or continue on error and log the errors.

The error log (and the corresponding display function) provides information in two levels :There is an error word bitfield that has a bit set corresponding to every error that occures in each single action and a bit set for each test procedure error , then there are counters for each register and for every test procedure that give the multiplicity of each error .



5.5.2 Mode Description

The differnt modes of operation in the menu driven run of the test are ===>

1. LOOP mode :

   You can choose to run the full test or any single task test in an infinite loop (until CR is pressed) or specifying the number of repetitions , aborting or continuing on error. Also for checking the output signals exists an infinite loop of setting and resetting particular bits on each register.

2. Interactive mode :

   You can run each test step by step specifying the conditions by answering the on screen questions and having the status of the module after each action on screen. If an error occures the control is passed to the FASTBUS interface menu.There is also the option of selecting a single action to activate , which gives the possibility to combine them to perform a test different from the defaults.

   In each of the interactive submenus reset module ,reset

error flag and display register values items are provided.

5.6 SUMMARY - DEFAULT SETTINGS

A users application can call mtc_test via the standard Single Board Test action flags either to run the full test and decide pass or fail from the return value of the function or to enter it in the menu driven mode .

There are two parameters that must be matched in the software with the corresponding hardware values (switch setable) :

1) The memory window (MEMSW) -- default value 8

2) Number of triggers from calibration -- default 3 .

This values can be changed via the main menu item d) or one can set them on the module to the hardcoded software values via the dip switces.

APPENDIX A

TYPICAL SESSION WITH MTC_TEST PROGRAM IN MENU DRIVEN MODE

After setting the MTC primary address in the top level menu and calling MTC__TEST in the mode GET__MENU, a typical test sesson could look like the following :

```
----- Main Menue -----
     e) Expert actions
     d) Display and change defaults
     a) Full test (1 time )
     l) Loop on  function
     s) Check signals
     i) Display error summary
     c) Clear error summary and reset
     g) Registers display
     r) Reset module
     t) Reset error flags
     x) Exit Program
Enter command: d

default MEMSW is : 8
 do you want to change it (1) :
default total fifo counts are : 9
 do you want to change it (1) :
default wait set offs is : 1
 do you want to change it (1) :
default extra trig is : 0
 do you want to change it (1) :
default calibration length is : 3
 do you want to change it (1) :


----- Main Menue -----
     e) Expert actions
     d) Display and change defaults
     a) Full test (1 time )
     l) Loop on  function
     s) Check signals
     i) Display error summary
```

```
     c) Clear error summary and reset
     g) Registers display
     r) Reset module
     t) Reset error flags
     x) Exit Program
Enter command: l

 ERRORS are not cleared by default

----- Loop Menue -----
    AVAILABLE FUNCTIONS ARE
     f0) Full_TEST , burning  test
     f1) Fbi0_test
     f2) Fbi11_test
     f3) Fbi12_test
     f4) Fbi13_test
     f5) RA_test
     f6) Fifo_test
     f7) Calibrate_test
     f8) Wait_test
     f9) Mem_Ov_test
     x) Exit Menue
Enter command: f0

 Infinite loop (0) or not (anything else) :0

 Stop on error(0) or continue test(anything else) :0

----- Loop Menue -----
    AVAILABLE FUNCTIONS ARE
     f0) Full_TEST , burning  test
     f1) Fbi0_test
     f2) Fbi11_test
     f3) Fbi12_test
     f4) Fbi13_test
     f5) RA_test
     f6) Fifo_test
     f7) Calibrate_test
     f8) Wait_test
     f9) Mem_Ov_test
     x) Exit Menue
Enter command: x

 TEST COMPLETED , NO ERRORS DETECTED

----- Main Menue -----
     e) Expert actions
     d) Display and change defaults
     a) Full test (1 time )
     l) Loop on  function
     s) Check signals
     i) Display error summary
     c) Clear error summary and reset
```

```
     g) Registers display
     r) Reset module
     t) Reset error flags
     x) Exit Program
Enter command: e

----- Main interactive Menue -----
     s) Set Primary Address (mtc_pa) (hex value)
     i) Fastbus Interface test menue
     f) Fifo test menue
     c) Callibrate test menue
     p) Clock Phase adjastment menue
     w) Wait test menu
     m) Memory overflow test menue
     a) Individuall action menue
     d) Display register
     r) Reset module
     t) Reset error flag
     x) Exit Menue
Enter command: r


 Reseting MTC .......
 CSR0 is set to : 1a20080
 CSR10 is set to : 0
 CSR11 is set to : 11
 CSR12 is set to : 1
 CSR13 is set to : 1f


----- Main interactive Menue -----
     s) Set Primary Address (mtc_pa) (hex value)
     i) Fastbus Interface test menue
     f) Fifo test menue
     c) Callibrate test menue
     p) Clock Phase adjastment menue
     w) Wait test menu
     m) Memory overflow test menue
     a) Individuall action menue
     d) Display register
     r) Reset module
     t) Reset error flag
     x) Exit Menue
Enter command: c



 Set Calibration mode
 CSR0 is set to : 1a200c0
 and so the Calibration mode is : 1
 Set Write Enable
Write enable : 1
 Hopefully you are providing an external Trigger input
```

```
  you expect to get back 5 consecutive read addresses
 Read ERROR flag
 CSR0 is set to : 1a208c5
 and so the ERROR flag is : 1
 Wait bit is : 1
 Test mode set
 TEST mode is : 1
 CSR10 is set to : 9
 and so the FIFO read addres is : 9
Generate Fastbus Encoder Ready
 CSR10 is set to : a
 and so the FIFO read addres is : a
Generate Fastbus Encoder Ready
 CSR10 is set to : b
 and so the FIFO read addres is : b
Generate Fastbus Encoder Ready
 CSR10 is set to : c
 and so the FIFO read addres is : c
 Reset Calibration mode
 CSR0 is set to : 1a20985
 and so the Calibration mode is : 0


----- Main interactive Menue -----
     s) Set Primary Address (mtc_pa) (hex value)
     i) Fastbus Interface test menue
     f) Fifo test menue
     c) Callibrate test menue
     p) Clock Phase adjastment menue
     w) Wait test menu
     m) Memory overflow test menue
     a) Individuall action menue
     d) Display register
     r) Reset module
     t) Reset error flag
     x) Exit Menue
Enter command: w

  Setting trigger address offset to 1 ...

----- Wait Test  Menue -----
     l) Loop over preselected values
     i) Manual test
     d) Display register
     r) Reset module
     t) Reset error flag
     x) Exit Menue
Enter command: r


 Reseting MTC .......
 CSR0 is set to : 1a20080
 CSR10 is set to : c
```

```
 CSR11 is set to : 14
 CSR12 is set to : 1
 CSR13 is set to : 1f


----- Wait Test  Menue -----
     l) Loop over preselected values
     i) Manual test
     d) Display register
     r) Reset module
     t) Reset error flag
     x) Exit Menue
Enter command: i

Enter Trigger Pipeline Depth (hex 3 bits ) :  1

 Triger Pipeline depth is : 1
 Set Write Enable
Write enable : 1
 Read ERROR flag
 CSR0 is set to : 1a20885
 and so the ERROR flag is : 1
 Wait bit is : 1
 Test mode set
 Assert first FB encoder ready, check for Fifo Status
Generate Fastbus Encoder Ready
 Fifo status bit is : 1
 Enc ready eq Pipeline Depth+1 Fifo  must be empty
 Enter 1 to assert another enc ready  0 to stop :  1
Generate Fastbus Encoder Ready
 Fifo status bit is : 0
 Enc ready eq Pipeline Depth+1 Fifo  must be empty
 Enter 1 to assert another enc ready  0 to stop :  0

 Test completed

----- Wait Test  Menue -----
     l) Loop over preselected values
     i) Manual test
     d) Display register
     r) Reset module
     t) Reset error flag
     x) Exit Menue
Enter command: x

----- Main interactive Menue -----
     s) Set Primary Address (mtc_pa) (hex value)
     i) Fastbus Interface test menue
     f) Fifo test menue
     c) Callibrate test menue
     p) Clock Phase adjastment menue
     w) Wait test menu
     m) Memory overflow test menue
```

```
     a) Individuall action menue
     d) Display register
     r) Reset module
     t) Reset error flag
     x) Exit Menue
Enter command: i

----- Fastbus Interface Menue -----
     l) Loop over preselected values
     i) Read Write register
     x) Exit Menue
Enter command: i

     i) Read module's ID
     c) Check flag setting reseting
     w) Write register (one of CSR10 11 12 13)
     r) Read register (one of CSR10 11 12 13)
     d) Display registers
     s) Reset MTC
     x) Exit Menue
Enter command: c

     e) Check error flag
     w) Check write enable
     c) Check Calibration mode set reset
     t) Check Test mode set reset
     r) Reset module
     f) Write Read FIFO read addres
     p) Write Read Triger Pipeline depth
     a) Write Read Triger addres offset
     k) Check Clock phase adjustement
     d) Display registers
     x) Exit Menue
Enter command: r

 Reseting MTC .......
 CSR0 is set to : 1a20080
 CSR10 is set to : 16
 CSR11 is set to : 11
 CSR12 is set to : 1
 CSR13 is set to : 1f

     e) Check error flag
     w) Check write enable
     c) Check Calibration mode set reset
     t) Check Test mode set reset
     r) Reset module
     f) Write Read FIFO read addres
     p) Write Read Triger Pipeline depth
     a) Write Read Triger addres offset
     k) Check Clock phase adjustement
     d) Display registers
     x) Exit Menue
```

```
Enter command: c

 Set Calibration mode
 CSR0 is set to : 1a200c0
 and so the Calibration mode is : 1
 Reset Calibration mode
 CSR0 is set to : 1a20080
 and so the Calibration mode is : 0

     e) Check error flag
     w) Check write enable
     c) Check Calibration mode set reset
     t) Check Test mode set reset
     r) Reset module
     f) Write Read FIFO read addres
     p) Write Read Triger Pipeline depth
     a) Write Read Triger addres offset
     k) Check Clock phase adjustement
     d) Display registers
     x) Exit Menue
Enter command: x

     i) Read module's ID
     c) Check flag setting reseting
     w) Write register (one of CSR10 11 12 13)
     r) Read register (one of CSR10 11 12 13)
     d) Display registers
     s) Reset MTC
     x) Exit Menue
Enter command: x

----- Fastbus Interface Menue -----
     l) Loop over preselected values
     i) Read Write register
     x) Exit Menue
Enter command: x

----- Main interactive Menue -----
     s) Set Primary Address (mtc_pa) (hex value)
     i) Fastbus Interface test menue
     f) Fifo test menue
     c) Callibrate test menue
     p) Clock Phase adjastment menue
     w) Wait test menu
     m) Memory overflow test menue
     a) Individuall action menue
     d) Display register
     r) Reset module
     t) Reset error flag
     x) Exit Menue
Enter command: a

----- Action Menue -----
```

```
     d) Display register
     e) Error Flag
     w) Write Enable
     c) Calibration mode
     s) Start Calibration
     t) Test mode
     f) Fifo address
     p) Trigger pipeline
     a) Trigger address ofset
     l) Clock phase
     i) Read wait
     k) Read Fifo status
     b) Assert FB trigger
     y) Assert FB encoder ready
     r) Reset module
     x) Exit Program
Enter command: t

 Enter 1 to set anything to reset :
 Test mode reset

----- Action Menue -----
     d) Display register
     e) Error Flag
     w) Write Enable
     c) Calibration mode
     s) Start Calibration
     t) Test mode
     f) Fifo address
     p) Trigger pipeline
     a) Trigger address ofset
     l) Clock phase
     i) Read wait
     k) Read Fifo status
     b) Assert FB trigger
     y) Assert FB encoder ready
     r) Reset module
     x) Exit Program
Enter command: x

----- Main interactive Menue -----
     s) Set Primary Address (mtc_pa) (hex value)
     i) Fastbus Interface test menue
     f) Fifo test menue
     c) Callibrate test menue
     p) Clock Phase adjastment menue
     w) Wait test menu
     m) Memory overflow test menue
     a) Individuall action menue
     d) Display register
     r) Reset module
     t) Reset error flag
     x) Exit Menue
```

```
Enter command: x

----- Main Menue -----
     e) Expert actions
     d) Display and change defaults
     a) Full test (1 time )
     l) Loop on  function
     s) Check signals
     i) Display error summary
     c) Clear error summary and reset
     g) Registers display
     r) Reset module
     t) Reset error flags
     x) Exit Program
Enter command: x

For further information, please contact:

       Panagiotis    Spentzouris    E771 , UOA.

       FNAL::SPENTZ
             x4250

       JNET%"SPA350GRATHUN1" ,  UOA.
```

A.1 ABBREVIATIONS

DE	Delay Encoder
FSCC	FASTBUS Smart Crate Controller
FSR	FASTBUS Standard Routines
MTC	Master Timing Controller
PC	Postamplifier/Comparator
RPX	Remote Procedure Execution
SE	Sequencer
SSD	Silicon Strip Detector
TSM	Test Stand Module

PN 436
Version : 1
15 April 1991

Silicon Strip Detector (SSD)
System Test Software Guide

Wolfgang Kowald
Duke University, E771

Panagiotis Spentzouris
UOA, E771

Dave Slimmer
Online Support Department
Fermilab Computing Division

This is the reference guide for the SSD system diagnostic part of the SSD product.

KEYWORDS: Diagnostic, Fastbus, SSD, Silicon Strip Detector

Systems Supported: pSOS, VMS V5.3
Software Version: V1.1

Acknowledgments

We would like to thank Gene Oleynik, Jonathan Streets, and Gary Sergey for their contribution in getting the diagnostic to run under VMS with remote procedure calls to the FSCC.

CONTENTS

CHAPTER 1

OVERVIEW

The primary goals of the SSD system diagnostic tests are to exercise as of much of the SSD hardware as possible and to exercise that hardware in a manner similar to the way it will be used in the experimental environment. The hardware includes a FASTBUS crate with SSD backplane, Postamp Comparator module (PC), Delay Encoder (DE), Sequencer (SE), and Master Timing Controller (MTC). A FSCC is used as the system control and data handling device as well as the processor running the diagnostic code.

Various hardware configurations involving test modules are used at the front end of the system in the system tests. There are three different front end test modules. Each of these provides an input to, or substitutes for, the PC module. FASTBUS test hardware modules include a Test Stand Module (TSM), and a Level Shifter Module (LSM). The TSM provides output patterns for the DE using the SSD backplane. Alternately, the TSM output can be used as input for the LSM module. The LSM then converts the TSM output to a level compatible with the front panel inputs of the PC. Note that tests using the TSM-LSM test module configuration produce results dependent on the TSM-LSM signals, which vary slightly from channel to channel in time, slope, and amplitude. An external test device, the Dave Christian's Card (DCC), can provide test pattern output directly to the PC inputs. A description of the hardware configurations is given in Hardware Note 102 (HN102).

The system test software is organized in a menu driven format with a tree structure. Top level menu items are generally concerned with FASTBUS environment settings. Lower level menus are concerned with either making special test initializations or selecting specific tests. Default values for all FASTBUS and test parameters are initialized immediately after the diagnostic is called to allow certain tests to be executed with minimal menu interaction. There is a default test provided as a "button test" which runs most of the individually selectable tests with default parameters.

Tests described by this document are meant to verify subsets of the total possible SSD system configuration. It is assumed that the appropriate single board tests for individual SSD modules have been successfully completed before these tests are attempted. Although the system tests provide error messages, problems should first be isolated by module substitution, and later by using single board tests to isolate a hardware fault. Because some of the single board tests do not test module I/O paths, the system tests may be needed to trouble-shoot an individual SSD module.

CHAPTER 2

ABBREVIATIONS SUMMARY

- add - address
- bkt - bucket
- cntr - counter
- corr - correction
- DCC - Dave Christian's Card
- DE - Delay Encoder
- dec - decimal
- encdr - encoder
- [F] - False
- FSCC - FASTBUS Smart Crate Controller
- FSR - FASTBUS Standard Routines
- horiz - horizontal
- IC - (PC) Individual Channel
- LSM - Level Shifter Module
- MTC - Master Timing Controller
- ovflw - overflow
- PAD - FASTBUS Primary address
- PC - Postamplifier/Comparator

- pln - plane
- prev - previous
- RPC - Resistive Plate Counter module (variant of PC)
- RPX - Remote Procedure Execution
- SC - (PC) Sum Channel
- SE/SEQ - Sequencer
- SSD - Silicon Strip Detector
- stats - statistics
- [T] - True
- trig - trigger
- TSM - Test Stand Module
- vert - vertical

CHAPTER 3

SYSTEM TEST OPERATION FORMATS

The diagnostic can be run in a variety of ways, but the method with the fastest execution speed and most convenience is using the ROM version of the code. The ROM version of the diagnostic can be installed in FSCC ROM bank 2. (Copies can be made of the latest version in release from Computing Department master prom set.) If FSCC ROM bank 2 is not available, the diagnostic can be downloaded from the SSD_DIAG$tests product directory. (The logical SSD_DIAG$tests pointing to the SSD tests directory is defined by setting up the SSD product with the command $ SETUP SSD_DIAG). Both the ROM version and downloaded version of the diagnostic run in FSCC RAM, so execution speed is the same. Downloading the diagnostic can take up to 10 minutes over a serial line, making this method far less convenient than using the ROM version. The diagnostic may also be run using remote procedure calls, using either the Ethernet or serial front panel ports on the FSCC. Although the Ethernet version is slightly faster, both methods have relatively slow execution speed due to the overhead associated with high numbers of small data transfers. Details of using the ROMed version follows. Please note that the commands are case insensitive, but are shown in capital letters for demonstration purposes.

3.1 ROM

The following sequence of commands copies the diagnostic code from FSCC ROM bank 2 to FSCC RAM, and starts the diagnostic.

{reset the FSCC manually with the front panel reset button}

```
$ SET HOST/DTE ttnn:          (establish serial communications with FSCC)
pROBE> go 40000               (run EPROM to RAM copy program)
pROBE> gs                     (start pSOS and do other board initializations)
pROBE> go                     (start diagnostic)
```

3.2 SERIAL DOWNLOAD

This command sequence downloads the diagnostic from the SSD_DIAG$tests project area directly to FSCC RAM using the PORT_MGR product, and starts the diagnostic.

```
$ SETUP PORT_MGR                    (if not done previously)

$ DOWNLOAD SSD_DIAG$root:[tests]ssd_diag.abs ttnn:
$ PTALK ttnn:   (or alternately)    $ SET HOST/DTE ttnn:
pROBE> gs                           (start pSOS and do other board initializations)
pROBE> go                           (start diagnostic)
```

3.3 RPX ETHERNET

This sequence of commands downloads the server code to the FSCC through the serial port and starts the "boss" on a VAX. The diagnostic runs as if it were executing on the VAX, and uses the Ethernet connection to the FSCC for communications.

```
$ SETUP SSD_DIAG                    (setup appropriate products)
$ SETUP PORT_MGR
$ SETUP TRMB0

$ SET DEFAULT SSD_DIAG$TESTS        (diagnostic and command procedure directory)
$ @defrpx                           (defines logicals - must be tailored to
                                      environment)
$ @boot_fscc_eth                    (download server code and start boss - starts
                                      boss with call to boss_eth.com. Both
                                      must be tailored to environment)
$ RUN/NODEBUG SSD_DIAG              (start diagnostic)
```

3.4 RPX SERIAL

This sequence of commands also downloads the server code to the FSCC through the serial port and starts the "boss" on a VAX. The diagnostic then runs as if it were executing on the VAX, using the serial connection to the FSCC for communications.

```
$ SETUP SSD_DIAG                    (setup appropriate products)
$ SETUP PORT_MGR
$ SETUP TRMB0

$ SET DEFAULT SSD_DIAG$TESTS        (diagnostic and command procedure directory)
$ @defrpx                           (defines logicals - must be tailored to
                                      environment)
$ @boot_fscc                        (download server code and start boss - starts
```

```
                                    boss with call to boss.com. Both
                                    must be tailored to environment)
$ RUN/NODEBUG SSD_DIAG              (start diagnostic)
```

CHAPTER 4

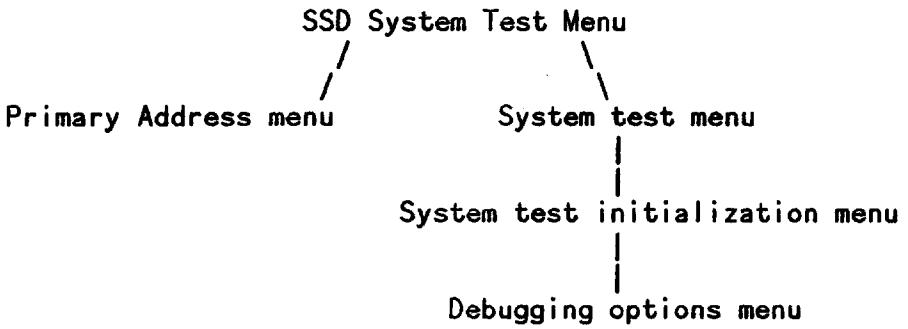
SYSTEM TEST SOFTWARE HARDWARE CONFIGURATION

There are four general test configurations used in the system tests. They are the PC/s injecting test bit patterns, the TSM substituting for the PC, the TSM driving the PC through the LSM, and the DCC generating fixed pattern input to the PC. The only test allowing for multiple PC/DE pairs is the PC injecting bit pattern test. Other tests will typically be run with one of each type of SSD module required, appropriate test module/s, and the FSCC, all in the same FASTBUS/SSD crate. Reference HN102 for the appropriate hardware test configuration.

CHAPTER 5

SYSTEM TEST SOFTWARE DESCRIPTION

5.1 MENU STRUCTURE OVERVIEW



5.2 MAIN MENU

The top level menu provides information on the FASTBUS environment as well as options to tailor the FASTBUS environment. In addition, it provides access to lower level menus, and a means to end the test program. Several initializations occur automatically as the test is started to reduce the amount of menu interaction required. Initializations include those done by menu option Initialize FASTBUS, and initialization of the long timer. Primary addresses are set assuming one of each type of SSD system module. Note that it is not necessary for the system tests to have the FSCC arbitrate for bus mastership. Generally, the only menu interaction required is to check the Primary address initializations and to select the system test submenu. The main menu:

```
          SSD System Test Menu
          ====================
       1) Initialize FASTBUS
       2) Arbitrate (status = unknown)
       3) Release Bus
       4) Set Long Timer (297 usec)
       5) Set Short Timer (1500 usec)
       6) Set Primary Addresses
       7) System test menu
      99) Exit to pROBE
Enter Command:
```

5.2.1 Initialize FASTBUS

When the SSD System diagnostic is started, the FASTBUS environment is automatically initialized by the FASTBUS routines called by this menu option. Each FASTBUS Standard Routine that is part of this initialization is displayed in a banner above the top level menu. The initialization creates a FASTBUS environment ID that is passed to the module tests, so an error will occur if this menu option is called more than once before calling menu option Release Bus. The FASTBUS initialization routines are:

- GPMINI - initialize interrupts and internal database
- FBOPEN - open FASTBUS routines for use
- FCIENV - allocate an environment ID and associated storage
- FNPALL - allocate FASTBUS port for use
- FBPSET(EG) - set environment parameter to enable geograhic addressing
- FBPSET(COEN) - set environment parameter to enable FSCC data FIFO to processor FIFO copy
- FBPSET(EOBI) - set environment parameter to enable end of block interrupt

5.2.2 Arbitrate

The menu option shows the current arbitor status of the FSCC, one of either "Master", "Slave", or "unknown". If this option is chosen, the FSCC tries to arbitrate for bus mastership using the

selected arbitration level. It is not necessary for the FSCC to be bus master to run these tests.

5.2.3 Release Bus

This option will release the bus and delete the current FASTBUS environment ID using the following routines:

- o FRLENV - releases system resources, environment ID becomes invalid
- o FBCLOS - deasserts any asserted FASTBUS port lines

5.2.4 Set Long Timer

The menu option displays the current environment value for the long timer. Selecting this option allows the long timer to be set to a new value. After selecting menu item Initialize FASTBUS, the long timer will always be reset to 5000000 usec.

5.2.5 Set Short Timer

Same as above, but the short timer can only be enabled or disabled. Selecting 1500 enables the timer and selecting 0 disables the timer.

5.2.6 Set Primary Address Menu

Set the primary addresses (slot number) for the SSD modules. The submenu displays the current values of the PADs.

```
                                    PAD (dec)
        a) Postamp Comparator         21
        b) TSM                        17
        c) Sequencer                  13
        d) Master Timing Controller   2
        x) Go back to main menu
    Select module:
```

The DE module has no FASTBUS interface, and therefore no PAD.

5.2.7 System Test Menu

In the process of bringing up the system test menu, an initialization is performed to allow the user to bypass the system test initialization submenu. The PC PAD displayed in the Primary address menu is written out to the SE as the PC's plane ID. All other PC plane IDs in the SE are set to zero. See the System test menu section for description of tests.

5.2.8 Exit To PROBE

In the ROM version, this option will terminate the SSD System tests with a pROBE break.

5.3 SYSTEM TEST MENU

A full description of each System test menu option follows.

a) System test initialization menu
b) TSM as PC test
c) PC test counter test
d) General readout loop
f) CLK1/DAC scan
g) PC channel quick tests
h) PC channel characterization tests
j) PC crosstalk test
k) Default system tests
l) Trigger address scan
x) Go back to main menu

Select test:

5.3.1 System Test Initialization Menu

All tests, with the exception of the General readout loop, are setup to use some default values for test parameters. The purpose of System test initialization menu options is to allow system tests to be run in non-default ways. Although each of these menu options are given default values when the diagnostic first starts, subsequent changes will remain in effect. Each test description section contains a section named "Initialization Menu Options" that should be checked prior to running a test. The menu is logically divided so that menu options are associated with their respective SSD module. Additionally, there are several options at the end of the menu that are not specifically associated with one module.

Several FASTBUS operations occur before the system test initialization menu scrolls onto the screen. The operations determine the current state (as displayed by the menu) of several selectable options for the SE and MTC. A detailed description of initialization menu options follows.

System test initialization menu with default states and options shown:

```
PC:     1) PC PADs/pln encdr IDs      2) Define PC PADs
        3) Running Test Cntr [F]      4) Rolling Test Cntr [T]
        5) Set Test Cntr (FE hex)     6) PC latch mode [F]
        7) Set DAC buffer (100 dec)   8) IC Enable [F]
        9) DAC scan (230,  0, -1, F) 10) SC Enable [F]

DE:    20) Accept prev bkt hit [T]   21) Loop on trig add (0 dec) [F]
       22) Set Trig Add Cor (-1 dec) 23) Trig add range (0 to 255 dec

SE:    30) Reset Sequencer           31) Define pln encdr IDs
       32) FASTBUS FIFO Enabled [F]  33) Aux FIFO Enabled [F]
       34) Set CLK1 (00), CLK2 (00)  35) Ovflw trunc Enabled [F]
       36) CLK1 scan ( 0,63, 1, F)

MTC:   40) Reset MTC                 41) Set trig pipe depth (0 dec)
       42) Write Enable on [F]

       50) Fill Buffer               51) Use RPC [F]
       52) One ch CLK1/DAC scan [F]  53) Print stats buf horiz
       54) FSCC Data FIFO enable [F] 55) Print stats buf vert
       66) Debugging options menu
       99) Exit menu
Enter Command:
```

- o PC PADs/pln encdr IDs

  Displays a list of PC PADs and their associated SE plane IDs. Each non-zero PC PAD in the list indicates a PC used during subsequent tests. Note that only the PC test counter test can handle multiple PC's running simultaneously.

- o Define PC PADs

  Allows creation of the PC PAD list. All previous PADs in the PAD list are set to zero. Each addition to the list is displayed. SE plane IDs should be redefined with the Define pln encdr IDs menu option.

- o Running Test Cntr [F]

  Sets a flag. Pertains only to the PC test counter test. See the test description for more detail.

- Rolling Test Cntr [T]

  Sets a flag. Applies only to the PC test counter test. See the test description for more detail.

- Set Test Cntr (FE hex)

  Sets a hex value in the range of 00 to FF (hex) which is passed to the PC test counter test. Pertains only to the PC test counter test. See the test description for more detail.

- PC latch mode [T]

  Sets a flag used by an internal PC initialization routine. In some tests this request may be overridden to establish the proper test conditions. The tests which automatically set this flag to the state shown in parenthesis include: PC test counter test (F), CLK1/DAC scan (T), PC channel characterization tests (T), PC channel quick tests (T), and PC crosstalk (T). This mode must be set to the desired state for the General readout loop test.

- Set DAC buffer (100 dec)

  Sets a 256 byte buffer with a single DAC value to be used by the PC. This value is used by tests which may not scan thru or initialize DAC values as part of the test. These tests include the General readout loop, PC channel characterization tests (CLK1 vs channel mode), and PC channel quick tests.

- IC Enable [T]

  Toggles a flag used by the PC initialization routine which enables or disables the PC Individual channel logic. This flag is automatically initialized to the appropriate state by the PC test counter test, the CLK1/DAC scan, the PC crosstalk test, the PC channel characterization tests, and the PC channel quick tests. This mode should always be set to the desired state for the General readout loop.

- DAC scan (230, 0, -1, F)

  Sets the DAC scan range for the following tests REGARDLESS OF THE INDICATED STATE OF THE FLAG: CLK1/DAC scan, PC channel characterization tests (in the DAC vs channel mode), and PC crosstalk test. The General readout loop uses the indicated DAC scan range only if the flag shows the scan function to be enabled. If the scan range is not enabled, the General readout loop uses

the PC DAC value as set by the Set DAC buffer (ddd dec) menu option where "ddd" is a value between 0 and 255.

Because there are a fixed number of PC CSR bits (8), a DAC value request greater than 255 will always be truncated to 255. Tests using the DAC scan range end when the terminal value is reached. If the DAC increment or decrement is set so that the terminal value never occurs, the test will go into an infinite loop.

- SC Enable [F]

  Toggles a flag used by the PC initialization routine which enables or disables the PC Sum channel logic. This flag is automatically initialized to the appropriate state by the PC test counter test, the CLK1/DAC scan, the PC crosstalk test, the PC channel characterization tests, and the PC channel quick tests. This mode should always be set to the desired state for the General readout loop.

- Accept prev bkt hit [T]

  Sets a flag. If set, internal data checking routines will accept a hit occurring in the previous bucket that may or may not be present in the current bucket.

- Loop on trig add (0 dec) [F]

  If enabled, the selected test continues to loop on the beginning trigger address as set by the Trig add range menu option until a keyboard return is entered. This switch applies to the PC test counter test (fixed test counter only), TSM as PC test, and General readout loop.

- Set Trig Add Corr (-1 dec)

  Sets up a positive or negative value in the range of -255 to +255 which is used in the DE trigger address offset calculations in the following tests: PC test counter test, TSM as PC test, CLK1/DAC scan, PC crosstalk, PC channel characterization tests, General readout loop, and PC channel quick tests. Normally this value should be set to zero except for the PC test counter test. See the System test description section for more information.

- Trig add range (0 to 255 dec)

  Sets the range of DE trigger addresses to be read out. Used by the following tests: PC test counter test, TSM as PC test, and General readout loop. The beginning address only is used by the DAC/CLK1 scan, the PC

crosstalk test, the PC channel characterization tests and the PC channel quick tests.

- Reset Sequencer

  Performs a FASTBUS write to SE CSR 0 to set the SE reset bit.

- Define pln encdr IDs

  Allows definition of plane IDs in SE RAM. Selecting this option will result in a query for all PC plane IDs. After the last PC plane ID is entered, the plane IDs are written to the SE. Then, the active PC PAD list and their corresponding plane IDs are displayed.

- FASTBUS FIFO Enabled [T]

  Switches state of SE FASTBUS FIFO by writing to SE CSR 0 and then checks if the bit is set. The flag following the menu option always reflects the enable state of the FIFO. The FASTBUS FIFO should be enabled for all system tests.

- Aux FIFO Enabled [F]

  Switches state of SE Auxiliary FIFO by writing to SE CSR 0 and then checks if the bit is set. The flag following the menu option always reflects the enable state of the FIFO. Having this FIFO enabled will cause overflow errors if there is no readout device attached to the SE auxiliary output port.

- Set CLK1 (20), CLK2 (24)

  Sets SE CLK1 with FASTBUS write to SE CSR space. The current value of CLK1 and CLK2 is read back and displayed by the menu. CLK2 is set by dip switches which are located on the SE module. The CLK1 value is important to tests that may not scan CLK1 values, such as the PC test counter test, the TSM as PC test, the General readout loop, the PC channel characterization tests, the PC channel quick tests, and the PC crosstalk test.

  Because the CLK1 value is always read back after it is set; and, there are a fixed number of bits (6) allocated in the internal data structure for the CLK1 value, a CLK1 value request greated than 63 will always be truncated to 63.

- Ovflw trunc Enabled [F]

  Switches the state of the SE overflow truncation function. When enabled, the SE truncates events after the SE FASTBUS FIFO is half full. This function should be disabled if more than one PC is used in the PC test counter test due to the amount of data produced by the PCs.

- CLK1 scan (0,63, 1, F)

  Sets the CLK1 scan range for the CLK1/DAC scan and the PC channel characterization tests REGARDLESS OF THE INDICATED STATE OF THE FLAG. The General readout loop is a special case and uses the indicated CLK1 scan range only if the flag shows the scan function to be enabled. If the scan range is not enabled, the General readout loop uses the SE CLK1 value indicated by the Set CLK1 (dd), CLK2 (dd) menu option where "dd" has a value from 0 to 63 and each unit is equal to 0.5 nsec.

  Because the CLK1 value is always read back after it is set; and, there are a fixed number of bits (6) allocated in the internal data structure for the CLK1 value, a CLK1 value request greater than 63 will always be truncated to 63. Tests using the CLK1 scan range end when the terminal value is reached. If the CLK1 increment or decrement is set so that the terminal value never occurs, the test will go into an infinite loop - perhaps a useful feature in trouble-shooting.

- Reset MTC

  Performs a FASTBUS write to MTC CSR 0 to set the MTC reset bit. This reset also resets the other SSD modules. The SE auxiliary FIFO enable state, FASTBUS FIFO enable state, and overflow truncation enable state may need to be reinitialized afer a MTC reset operation.

- Set trig pipe depth (7 dec)

  Sets the MTC trigger pipeline depth (0 to 7) to the requested value by a write to MTC CSR space. The current pipe depth is displayed by the menu option.

- Write Enable on [T]

  Switches state of MTC write enable, which when on allows DEs to accept data. The current state of write enable is displayed.

o Fill Buffer

Calls submenu that gives options for filling the data buffer loaded to the TSM memory. (Data loaded into the TSM memory becomes the TSM output.) This option applies to the TSM as PC test and the General readout loop which do not initialize the buffer used by the TSM.

Other tests which do initialize the TSM input buffer, but may be overridden, include the CLK1/DAC scan, the PC channel characterization tests, and the PC channel quick tests. Entering "111" at the System initialization menu "Enter Command" prompt will toggle a flag that allows the user to change the default buffer initialization for these three tests. When menu responds to entering "111" with "use default hit_buffer [F]", the tests will use the buffer as initialized by the Fill Buffer menu option. When the menu responds to entering "111" with "use default hit_buffer [T]", the tests will use default buffer initializations.

The menu:

```
 0  : fill selected position
 1  : fill with row of pattern
 2  : fill with 5555...
                 AAAA...
 3  : fill with FFFF...
                 0000...
 4  : fill with PC counter -fixed- pattern
 5  : fill with PC counter -running- pattern
 6  : fill with random pattern
 7  : fill with repeated N raw pattern
                 LLLL...
                 WWWW...
                 KKKK...
 8  : display selected positions
 99 : exit

 Selection :
```

o Use RPC [F]

Sets a flag used by the TSM initialization routine. If set, this flag enables a variation on the scrambling routine used to setup the TSM data buffer which is specific to the RPC version of the PC. Assuming that only PCs are tested, none of the system tests require this flag to be set.

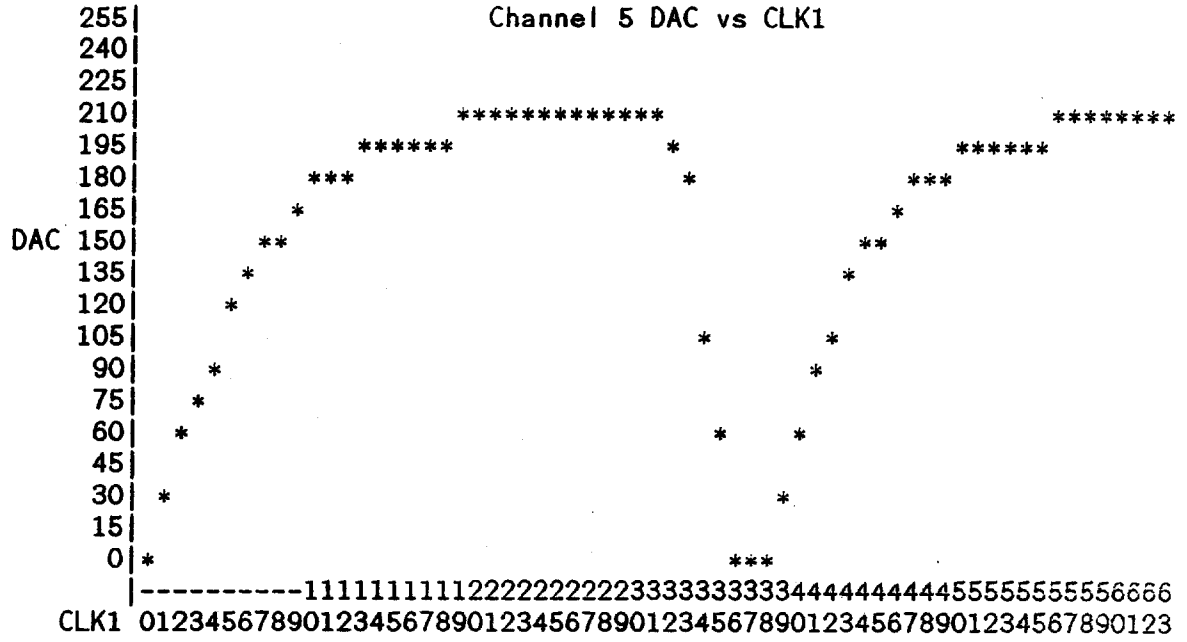
- One ch CLK1/DAC scan [F]

  This option, which only applies to the CLK1/DAC scan, allows one selected channel to be scanned, verses all 128 channels.

- Print stats buf horiz

  This menu option is for examining the CLK1/DAC scan data on individual channels. It allows selection of a single channel or range of channels for graphic display. If using the RPX version of the system tests, the graphs may be printed to an output file by enabling the Log errors to file switch in the Debugging options menu.

  Example output:



  The DAC values shown on the vertical axis represent the programmed DAC value. Actual DAC output is approximately equal to the programmed value multiplied by 5mV. CLK1 values on the horizontal axis also represent programmed value. The actual CLK1 value is equal to the programmed value multiplied by 0.5 nsec. Note that the graph displays the actual hit (the first pulse) and an extra hit, (the second pulse) which is provided by the DE. The extra hit provided by the DE is enabled by jumpers on the DE board.

- FIFO block transfer enable [F]

  This option allows use of the FSCC data FIFO. If enabled, data is also copied to the FSCC processor FIFO

for checking. Enabling this option may slow test execution speed, and in some cases the FSCC data FIFO may overflow. This option, however, will allow data to be read out by an auxiliary device attached to the FSCC.

- Print stats buf vert

  This option allows a finer examination of the CLK1/DAC scan channel data. The display for each selected channel is distributed accross several terminal screens.

- Debugging options menu (default state on switches shown)

```
a) Stop on error switch [F]
b) Readout loop delay (0 dec) [F]
c) Debugging mode switch [F]
d) Stepping mode switch [F]
e) Log errors to file switch (rpx only) [F]
f) Log errors to banner switch [F]
g) Print errors to terminal [F]
h) Display Status
x) Exit debugging options menu
Select option:
```

- Stop on error switch [F] - Sets a flag. Applies to PC test counter test and TSM as PC tests only.

- Readout loop delay (0 dec) [F] - Sets a flag and a delay value if enabled. Applies to PC test counter test and TSM as PC tests only. A delay count value of 3,000,000 produces a delay of about 1 second.

- Debugging mode switch [F] - If enabled, sets a flag with test specific meanings as follows.

  CLK1/DAC scan - display TSM input buffer, each CLK1 value used, each DAC value used
  PC crosstalk test - display first 5 TSM input buffer locations
  PC channel characterization tests - display TSM input buffer, display internally buffered data for graphi display
  PC channel quick tests - display TSM input buffer

- Stepping mode switch [F] - Sets a flag. Internal routines for dynamic memory allocation, deallocation, PC initialization, MTC initialization, and SE initialization respond with a print statement indicating the next action to be taken, which must be initiated with a keyboard <Return>. The PC test counter test and TSM as PC test also use this flag in the same manner to indicate internal test

actions.

- Log errors to file switch (rpx only) [F] - This option tries to open a file with the name "syst.log" in the current directory. All tables and graphs output to the terminal are also written to this log file. Menus and other prompts do not appear in this file. The log file is always opened with the append option, so all data logged will go to the same file.

- Log errors to banner switch [F] - Sets a flag. If enabled, the PC test counter test and TSM as PC test will have a banner displayed in inverse video along the bottem of the terminal screen indicating the current PC plane ID, DE address being read, total number of data errors, total number of word count errors, and total test iterations. Because of the frequent screen updates, use of this option will significantly reduce test execution speed.

- Print errors to terminal [F] - If enabled, errors detected after each DE address read will be displayed in tabular form. An example:

```
 ================> PC 21 <===============
|-------------------------------------------------------------------------
|  0 **  |  1 *  P|  2 **  |  3 *  P|  4 **  |  5 *  P|  6 **  |  7 *
|  8 **  |  9 *  P| 10 **  | 11 *  P| 12 **  | 13 *  P| 14 **  | 15 *
| 16 **  | 17 *  P| 18 **  | 19 *  P| 20 **  | 21 *  P| 22 **  | 23 *
| 24 **  | 25 *  P| 26 **  | 27 *  P| 28 **  | 29 *  P| 30 **  | 31 *
| 32     M| 33     M| 34     M| 35     M| 36     M| 37     M| 38     M| 39
| 40     M| 41     M| 42     M| 43     M| 44     M| 45     M| 46     M| 47
| 48     M| 49     M| 50     M| 51     M| 52     M| 53     M| 54     M| 55
| 56     M| 57     M| 58     M| 59     M| 60     M| 61     M| 62     M| 63
| 64     M| 65     M| 66     M| 67     M| 68     M| 69     M| 70     M| 71
| 72     M| 73     M| 74     M| 75     M| 76     M| 77     M| 78     M| 79
| 80     M| 81     M| 82     M| 83     M| 84     M| 85     M| 86     M| 87
| 88     M| 89     M| 90     M| 91     M| 92     M| 93     M| 94     M| 95
| 96     M| 97     M| 98     M| 99     M|100     M|101     M|102     M|103
|104     M|105     M|106     M|107     M|108     M|109     M|110     M|111
|112     M|113     M|114     M|115     M|116     M|117     M|118     M|119
|120     M|121     M|122     M|123     M|124     M|125     M|126     M|127
|-------------------------------------------------------------------------
  Extra Ch. (E) :   0, Missing Ch. (M) :  96, Prev. Bucket Error (P) :
  TA    :   0, CLK1 :  20      Buffer [254] : FFFFFFFFFFFFFFFFFFFFFFFFAA
  CLK2  :  24, DAC  :  50             [255] : FFFFFFFFFFFFFFFFFFFFFFFFFF
  Rec.  :  32, Exp. : 128       ==>   [  0] : 00000000000000000000000000
                                      [  1] : 00000000000000000000000055
```

The first "star" to the right of a channel number indicates that an expected channel was received. The second "star" to the right of a channel number indicates the expected previous hit flag was

received. Single capital letters following channel numbers indicate errors as shown by the key at the bottom of the table.

- Display Status - Selecting this option will cause a table of information to be printed to the screen which represents the current content of the SSD module CSR registers. An example display:

```
==========================================================================
MTC CSR0=01a20985 CSR10=00000100 CSR11=00000010 CSR12=000000fc CSR13=00

    TRIG_PHASE_ERR=0 FIFO_OVFL_ERR   =0  MEM_OVWR_ERR=1 DE_SEQ_ERR
    CLOCK_MISSING =0 TEST_MODE       =1  ENC_READY   =1 CALIBRATION_MOD
    WRITE_ENABLE  =1 FIFO_RA         =00 PRDPTH      =0 TOFFSET
    CKPHASE       =00 ERROR_FLAG      =1 TRIG_WAIT   =1 FIFO_NOT_EMPTY
--------------------------------------------------------------------------
SE  CSR0=01a30040 CSRCx10=00000000 CSRCx11=040c0600 CSRCx12=00000000

    OVERFLOW_TRUNC   =0  LOAD_AUXILIARY  =0  LOAD_FASTBUS    =1 BLOCK_C
    WORD_COUNT       =00 CLK2_DELAY      =18 CLK1_DELAY      =00
    ENCODER_ERROR_STATUS  =000      SYNC_ERROR_FLAG =0
    PLANE_ENCODER_RAM : 00 00 00 00 00 00 00 00 00 00 00 00
--------------------------------------------------------------------------
PC  SLOT=15 CSR0=00000000 CSR1=00000000

    1MHz_CLOCK    =X CLK1_ENABLE=0 SUM_CHANNEL =0 SINGLE_CHANNEL=0
    MODE_RUN      =0 MOD_STATUS =X MODE1_STATUS=X MODE2_STATUS  =X
    TEST_COUNTER =00
```

5.4 SYSTEM TESTS DESCRIPTION

The system tests can be broadly categorized as those using the PC front panel input and those that do not. The "PC test counter test" and the "TSM as PC test" are the two tests that do not use the PC front panel input path. All other tests in the menu do use the PC front panel input path. A full description of each test follows.

5.4.1 TSM As PC Test

In this test the TSM generates test patterns and outputs them via the backplane to the DE. The DE is read out by the FSCC from the SE FIFO. Bit patterns are verified by the FSCC. System triggers are generated by the MTC. Errors may be displayed while the test is running as with the PC test counter test.

5.4.1.1 Test Pattern - The TSM input data buffer must be initialized before running the TSM as PC test with the Fill buffer menu option. There is no default initialization for the data buffer for this test.

5.4.1.2 Initialization Menu Options - (default state shown)

- o Loop on trig add (0 dec) [F]
- o Accept prev bkt hit [T]
- o Trigger add range (0 to 255 dec)
- o Aux FIFO Enabled [F]
- o FASTBUS FIFO Enabled [T]
- o Ovflw trunc Enabled [F]
- o Set CLK1 (20), CLK2 (24)
- o Set Trig Add Corr (-1 dec)

5.4.1.3 Debugging Menu Options - The following options can be used for the TSM as PC test. The default state is shown.

- o Stop on error switch [F]
- o Readout loop delay (0 dec) [F]
- o Stepping mode switch [F]
- o Log errors to file switch (rpx only) [F]
- o Log errors to banner switch [F]
- o Print errors to terminal [F]

5.4.2 PC Test Counter Test

In this test the PC generates test patterns which are latched by the DE and read out by the FSCC from the SE. The PC test patterns are verified by the FSCC. System triggers are generated by the MTC. Errors may be reported during the test by enabling one of the debugging submenu options. An error summary is given

at the end of the test. See appendix for standard error summary. NOTE: the TSM and LSM modules should be removed from the crate during this test or spurious errors will occur. In addition, the trigger address correction should be set to 1 with the system test initialization menu option Set Trig Add Corr (d dec). This test will loop indefinately until a keyboard <Return> is entered.

5.4.2.1 Initialization Menu Options -

5.4.2.1.1 Test Pattern -

1. Running Test Cntr [T], Rolling Test Cntr [F]

The DE is filled with patterns resulting from incrementing the PC test counter by one, 255 times. (See appendix for patterns.) Each input buffer pattern always corresponds to the same DE trigger address, i.e. first test counter pattern to DE trigger address 0, second test counter pattern to DE trigger address 1, etc. The initial test counter pattern is set with option Set Test Cntr (xx hex).

2. Running Test Cntr [F], Rolling Test Cntr [T]

The DE is filled with patterns resulting from incrementing the PC test counter by one, 255 times. After all PC test counter patterns are read out from the DE, the PC test counter patterns are rewritten to the DE, incrementing the test patterns relationship to DE trigger addresses by one. The PC test counter patterns are "rotated" through DE trigger addresses in this manner so that each test pattern is eventually read out from each possible DE trigger address. The initial test counter pattern is set with option Set Test Cntr (xx hex). This is the default mode for the PC injecting bit pattern test.

3. Running Test Cntr [F], Rolling Test Cntr [F]

With both options false, the entire DE memory is filled with the pattern resulting from the test counter value input with option Set Test Cntr (xx hex).

5.4.2.1.2 Other Options Affecting Test - (default state shown)

- o Loop on trig add (0 dec) [F]
- o Accept prev bkt hit [T]
- o Trigger add range (0 to 255 dec)
- o Aux FIFO Enabled [F]
- o FASTBUS FIFO Enabled [T]
- o Ovflw trunc Enabled [F]
- o Set CLK1 (20), CLK2 (24)
- o Set Trig Add Corr (-1 dec)

5.4.2.1.3 Options Set Automatically By Test - (to state shown)

- o PC latch mode [F]
- o IC Enable [F]
- o SC Enable [F]

5.4.2.2 Debugging Menu Options - The following options can be used for the PC test counter test. The default state is shown.

- o Stop on error switch [F]
- o Readout loop delay (0 dec) [F]
- o Stepping mode switch [F]
- o Log errors to file switch (rpx only) [F]
- o Log errors to banner switch [F]
- o Print errors to terminal [F]

5.4.3 General Readout Loop

This is a general readout routine which allows maximum flexiblity in a test configuration using test modules like the TSM-LSM pair or the DCC to provide input for a single PC/DE pair of SSD

modules. This flexibilty is provided to allow creation of special purpose trouble-shooting loops. Data taken from this test is not saved, but the standard error summay is displayed when the General readout loop is stopped by keyboard <Return> or ends. There are no default parameters supplied by the test which requires that several parameters be setup in the System test initialization menu prior to executing the test. These parameters control DAC, CLK1 and trigger address loops. The input data buffer for the TSM must also be setup in the System initialization menu with the Fill Buffer menu option.

5.4.3.1 Required Initializations - (default state shown)

- o CLK1 scan (0,63, 1, F) - If the scan range is not enabled as indicated by the flag, the test will use the CLK1 value displayed by the Set CLK1 (dd), CLK2 (dd) menu option.
- o Set CLK1 (20), CLK2 (24) - Used if CLK1 scan function not enabled.
- o Fill Buffer
- o PC latch mode [T]
- o DAC scan (230, 0, -1, F) - If the scan range is not enabled as indicated by the flag, the test will use the DAC value displayed by the Set DAC buffer (ddd dec) menu option.
- o Set DAC buffer (100 dec) - Used if DAC scan function not enabled.
- o IC Enable [T]
- o SC Enable [F]

5.4.3.2 Optional Initializations - (default state shown)

- o Loop on trig add (0 dec) [F]
- o Accept prev bkt hit [T]
- o Trigger add range (0 to 255 dec)
- o Aux FIFO Enabled [F]

- o FASTBUS FIFO Enabled [T]
- o Ovflw trunc Enabled [F]
- o Set Trig Add Corr (-1 dec)

5.4.3.3 Debugging Menu Options -

- o Debugging mode switch [F]
- o Log errors to file switch (rpx only) [F]
- o Print errors to terminal [F]

5.4.4 CLK1/DAC Scan

This test produces a data base which contains the highest PC DAC value at which a channel is on for each SE CLK1 delay. The test is designed to take data on (1) or (128) channels through user selected ranges of DAC values and CLK1 delays. The channels may be read at a user selected DE trigger address. A table is provided at the end of the test which indicates the CLK1 value at the highest DAC value found for each channel. The criteria for finding a "higher" DAC value is that the DAC value must be 6 counts higher than the previous highest DAC value. This condition is imposed to try to find the highest DAC value in the actual hit instead of in the extra hit given by the DE. (In the graph displayed by the Print stats buf horiz menu option, the actual hit corresponds to the first pulse, and the extra hit given by the DE corresponds to the second pulse.) The average CLK1 value is a simple average of all CLK1 values displayed in the table below.

ch:	CLK1	ch:	CLK1	ch:	CLK1	ch:	CLK1	ch:	CLK1	ch:	CLK1	ch:	CLK1	ch:	CLK1
0:	23	1:	20	2:	16	3:	23	4:	18	5:	21	6:	22	7:	25
8:	15	9:	19	10:	15	11:	23	12:	15	13:	21	14:	14	15:	21
16:	58	17:	24	18:	17	19:	20	20:	15	21:	17	22:	18	23:	26
24:	14	25:	20	26:	16	27:	18	28:	17	29:	18	30:	13	31:	20
32:	15	33:	22	34:	16	35:	19	36:	18	37:	19	38:	14	39:	19
40:	16	41:	18	42:	10	43:	19	44:	15	45:	16	46:	10	47:	22
48:	13	49:	17	50:	13	51:	21	52:	18	53:	18	54:	10	55:	22
56:	15	57:	16	58:	8	59:	18	60:	18	61:	14	62:	12	63:	19
64:	21	65:	16	66:	12	67:	21	68:	17	69:	18	70:	16	71:	24
72:	16	73:	14	74:	15	75:	19	76:	22	77:	14	78:	19	79:	21
80:	18	81:	14	82:	14	83:	21	84:	16	85:	12	86:	15	87:	24
88:	19	89:	11	90:	14	91:	20	92:	16	93:	14	94:	15	95:	21
96:	0	97:	16	98:	10	99:	16	100:	16	101:	20	102:	14	103:	18
104:	20	105:	16	106:	10	107:	21	108:	18	109:	18	110:	16	111:	19
112:	19	113:	14	114:	8	115:	19	116:	19	117:	14	118:	15	119:	14
120:	18	121:	11	122:	9	123:	14	124:	22	125:	11	126:	16	127:	14

average CLK1 : 17
max CLK1 : 58
min CLK1 : 0

A error summary is also provided at the end of test. The system initialization menu options Print stats buf horiz and Print stats buf vert provide the means to examine individual channel data from the data base produced by the test. The data base contains valid data after the CLK1/DAC scan has completed, and retains that data until the CLK1/DAC scan is re-run, causing old data to be overwritten.

5.4.4.1 Test Pattern Input - The input data buffer for the TSM consists of (hex) A's and 5's. This data will produce a 50% duty cycle at the input of the PC. The buffer appears as follows:

```
[  0] AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
[  1] 55555555555555555555555555555555
[  2] AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
[  3] 55555555555555555555555555555555
   .  .
   .  .
   .  .
[255] 55555555555555555555555555555555
```

5.4.4.2 Initialization Menu Options - (default state shown)

- o Accept prev bkt hit [T]
- o Trigger add range (0 to 255 dec) - Uses beginng trigger address only.
- o Aux FIFO Enabled [F]
- o FASTBUS FIFO Enabled [T]
- o Ovflw trunc Enabled [F]
- o CLK1 scan (0,63, 1, F)
- o Set Trig Add Corr (-1 dec)
- o DAC scan (230, 0, -1, F)
- o One ch CLK1/DAC scan [F]
- o Print stats buf horiz
- o Print stats buf vert

5.4.4.3 Options Set Automatically By Test - (to state shown)

- o PC latch mode [T]
- o IC Enable [T]
- o SC Enable [F]

5.4.4.4 Debugging Menu Options -

- o Debugging mode switch [F]
- o Log errors to file switch (rpx only) [F]
- o Print errors to terminal [F]

5.4.5 PC Channel Quick Tests

The "quick" tests for the PC front end use two TSM-LSM generated data patterns. The first pattern presented to the PC input is all 0's and should not generate PC output. If channels do

appear, they are considered "hot" channels. The second pattern should turn all channels on. If any channel does not appear given the second pattern, it is considered a "dead" channel. This test provides very limited information about a PC module, and is meant to be used only as a cursory examination of a PC to locate potential problems. Different variations of the second pattern are used to test the PC individual and sum channel logic. After the test executes, the SSD modules remain active, with the TSM-LSM continuously presenting the second data pattern to the PC inputs. This provides an opportunity to trace signals through the PC.

Output is displayed graphically as follows:

"0" = chan 0 - 63
"+" = chan 64 - 127
"p" = previous hit flag error
"e" = extra hit flag error
"*" = previous hit flag and extra hit flag error

symbol plotted along top of grid for hot channel
symbol plotted along bottom of grid for dead channel

Sample quick test output testing individual channel logic.

```
          |                  Individual Channel Quick Test              DAC=50
  Hot     |                                                             CLK1=17
Channels  |
          |
          |
 Good     |
Channels  |0000000000000000000000000000000000000000000000000000000000000000
0 to 63   |
          |
          |
          |
 Good     |
Channels  |++++++++++++++++++++++++++++++++ ++++++++++++++++++++++++++++++++
64 to 127 |
          |
          |
 Dead     |
Channels  |                          +
          |----------1111111111222222222233333333334444444444555555555566666
          0123456789012345678901234567890123456789012345678901234567890123
 Dead channels
   96
```

Note that the horizontal scale indicates channel numbers for channels 0 to 63 only. Channels in the range from 64 to 127 may be identified by adding 64 to the appropriate horizontal scale number, or by using the "Dead channel" list

summary displayed at the lower left side of the graph.

5.4.5.1 Test Pattern Input - The input data buffer for the TSM consists of patterns selected via submenu to test either the PC individual or sum channel logic. When testing the individual channel logic, the sum channel logic is disabled and visa versa. The tests and corresponding input patterns are:

Whole buffer 0's, followed by:

Individual channel
single 0's, F's
Sum channel (3's)
single 0's, 3's
Sum channel (6's)
single 0's, 6's
Sum channel (C's)
single 0's, C's
Sum channel (1's)
single 0's, 1's
Sum channel (2's)
single 0's, 2's
Sum channel (4's)
single 0's, 4's
Sum channel (8's)
single 0's, 8's

A representative buffer for the Individual channel test:

```
[  0] 00000000000000000000000000000000
[  1] FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
[  2] 00000000000000000000000000000000
[  3] 00000000000000000000000000000000
   .  .
   .  .
   .  .
[255] 00000000000000000000000000000000
```

5.4.5.2 Initialization Menu Options -

- o Accept prev bkt hit [T]
- o Trig add range (0 to 255 dec) - Only the beginning trigger address used.

- o Aux FIFO Enabled [F]
- o FASTBUS FIFO Enabled [T]
- o Ovflw trunc Enabled [F]
- o Set CLK1 (20), CLK2 (24)
- o Set DAC buffer (100 dec) - Depending on the combination of DAC value and input threshold used, adjacent sum channel responses indicated by "e"s on the graph may be present.
- o Set Trig Add Corr (-1 dec)

5.4.5.3 Automatic Initializatons -

- o PC latch mode [T]
- o IC Enable [T] - State depends on selected test.
- o SC Enable [F] - State depends on selected test.

5.4.5.4 Debugging Menu Options -

- o Debugging mode switch [F]
- o Log errors to file switch (rpx only) [F]
- o Print errors to terminal [F]

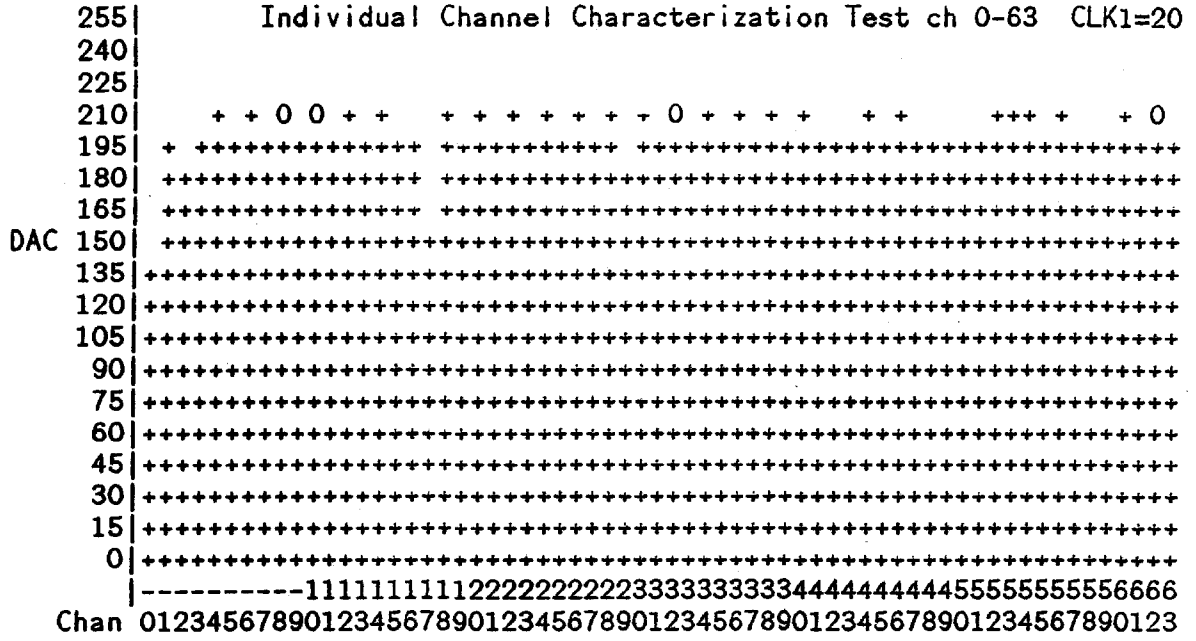
5.4.6 PC Channel Characterization Tests

Depending on the selected submenu option, this test does individual or sum channel characterization for the Postamp Comparator front end. The default test checks individual or sum channel response by scanning through a selected DAC range and recording where the channels respond less than 8 out of 8 reads and also where the channels respond 8 out of 8 reads. The fixed CLK1 value is user selected. The alternate test scans thru a selected CLK1 range using a selected, fixed DAC value. Selecting the DAC vs channel submenu option will toggle the test mode to CLK1 vs channel and visa versa.

Output is displayed graphically as follows:

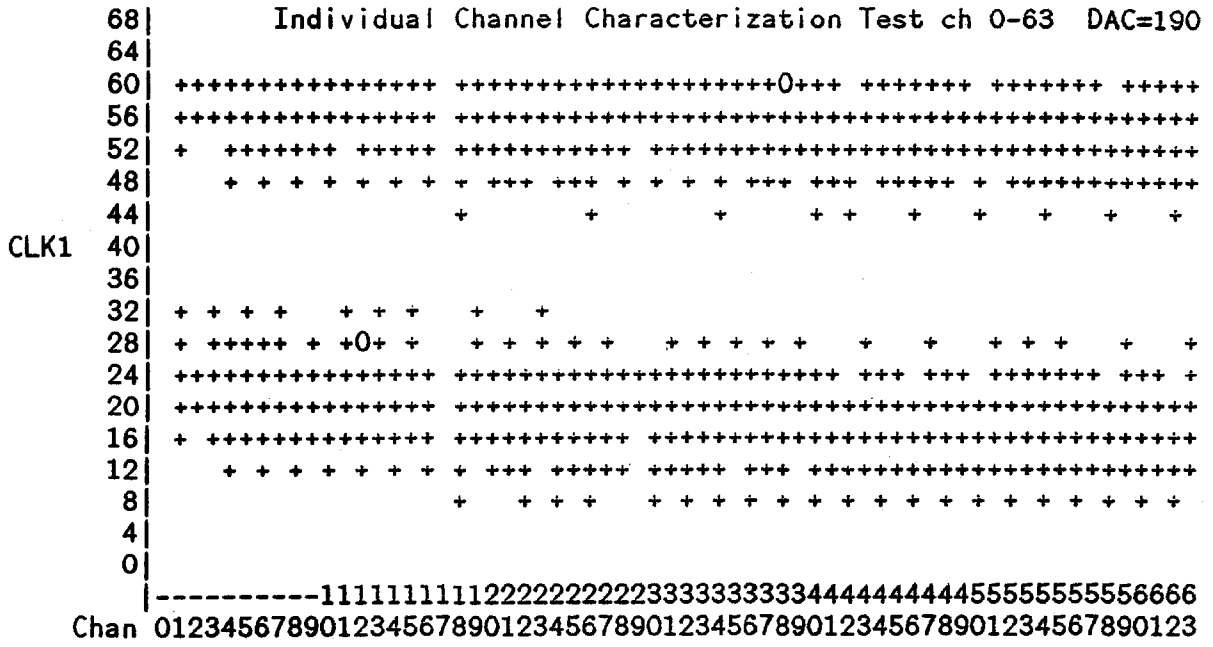
no hits	= blank space
< 8 out of 8 hits	= '0'
8 out of 8 hits	= '+'

Sample output for the DAC vs channel mode:



Output for this test is displayed in two terminal screens. The first screen displays channels 0 to 63, and the second screen displays channels 64 to 127.

Sample output for the CLK1 vs channel mode:



Output is also displayed in two screens as with the DAC vs channel mode.

5.4.6.1 Test Pattern Input - The input data buffer for the TSM consists of patterns selected via submenu to test either the PC individual or sum channel logic. When testing the individual channel logic, the sum channel logic is disabled and visa versa. The tests and corresponding patterns are:

```
Individual channel
                whole buffer F's, 0's
Sum channel (3's)
                whole buffer 3's, 0's
Sum channel (6's)
                whole buffer 6's, 0's
Sum channel (C's)
                whole buffer C's, 0's
Sum channel (1's)
                whole buffer 1's, 0's
Sum channel (2's)
                whole buffer 2's, 0's
Sum channel (4's)
                whole buffer 4's, 0's
Sum channel (8's)
                whole buffer 8's, 0's
```

A representative buffer for the Individual channel test:

```
[  0] FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
[  1] 00000000000000000000000000000000
[  2] FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
   .  .
   .  .
   .  .
[255] 00000000000000000000000000000000
```

5.4.6.2 Initialization Menu Options - (defaults shown)

- o Accept prev bkt hit [T]
- o Trig add range (0 to 255 dec) - The beginning trigger address only used.
- o Aux FIFO Enabled [F]
- o FASTBUS FIFO Enabled [T]
- o Ovflw trunc Enabled [F]
- o CLK1 scan (0,63, 1, F) - Used for the CLK1 vs channel mode.

- Set CLK1 (20), CLK2 (24) - Used for the DAC vs channel mode.
- Set DAC buffer (100 dec) - Used for the CLK1 vs channel mode.
- Set Trig Add Corr (-1 dec)
- DAC scan (230, 0, -1, F) - Used for the DAC vs channel mode.

5.4.6.3 Automatic Initializatons -

- PC latch mode [T]
- IC Enable [T] - State depends on selected test.
- SC Enable [F] - State depends on selected test.

5.4.6.4 Debugging Menu Options -

- Debugging mode switch [F]
- Log errors to file switch (rpx only) [F]
- Print errors to terminal [F]

5.4.7 PC Crosstalk Test

Crosstalk test for the Postamp Comparator front end. This test uses the TSM-LSM configuration to generate a single hit, and then checks all channels for the expected hit and any extra hits. The test cycles through all 128 channels placing a hit on each channel. Each channel is checked through a selected PC DAC range. Up to four channels showing crosstalk are recorded for each channel expected to produce an output. No channel is reported more than once, even though it may be detected as a crosstalk channel at several DAC values. Errors are printed in a table at the end of the test.

Channels under the label "Ich" are the channels with the input or the ones expected to produce an output. Channel numbers appearing under the label "ch" produced an unexpected output at the DAC value shown to the right of the channel number. Sample crosstalk error output with standard error summary:

```
ERROR                Crosstalk results
 Ich    ch; DAC        ch; DAC        ch; DAC        ch; DAC

   2     6;  24         6;  23         6;  22         6;  21
   6     2;  25         2;  24         2;  23         2;  22
  29    33;  52        33;  51        33;  50        33;  49
  96    92;  43        92;  42        92;  41        92;  40
 125   121;  24       121;  23       121;  22       121;  21

 Total events read:        27008        Data verify errors:       3007
 Events with errors:        2979        Prev. bucket errors:       245
 Sync errrors:                 0        Missing hit errors:       2686
 Word count errors:         2796        Extra hit errors:           76
```

If no error is found, the test will print a "no crosstalk errors found" message before the standard error summary.

5.4.7.1 Test Pattern Input - Input data buffer for the TSM consists of a single channel on in the entire buffer. The representative buffer displayed below is used for testing channel 0.

```
[  0] 00000000000000000000000000000001
[  1] 00000000000000000000000000000000
[  2] 00000000000000000000000000000000
   .  .
   .  .
   .  .
[255] 00000000000000000000000000000000
```

5.4.7.2 Initialization Options -

- o Trig add range (0 to 255 dec) - Accepts beginning trigger address only.
- o Aux FIFO Enabled [F]
- o FASTBUS FIFO Enabled [T]
- o Ovflw trunc Enabled [F]
- o Set CLK1 (20), CLK2 (24)
- o Set Trig Add Corr (-1 dec)
- o DAC scan (230, 0, -1, F)

5.4.7.3 Automatic Initializatons - (to state shown)

- o PC latch mode [T]
- o IC Enable [T]
- o SC Enable [F]

5.4.7.4 Debugging Menu Options -

- o Debugging mode switch [F]
- o Log errors to file switch (rpx only) [F]
- o Print errors to terminal [F]

5.4.8 Default System Tests

This test is a shell used to run several of the system tests. The system tests run with default parameters which are in effect only during the execution of this test. These tests include:

1. CLK1/DAC scan - Default parameters include a DAC scan range of 250 to 0 by steps of -1, and a CLK1 scan range of 0 to 63 by steps of 1.

2. PC channel quick tests - This test is run with the (8) patterns which correspond to PC individual and sum channel modes.

3. PC channel characterization tests - The test is run in the DAC versus channel mode and CLK1 versus channel mode with PC individual logic enabled only.

4. PC crosstalk test - The default parameters for the DAC scan range are 230 to 20 by steps of -1.

5.4.9 Trigger Address Scan

This option is not a test, but uses the TSM-LSM pair in conjunction with a DE-PC pair to determine the DE address where TSM generated data should appear. The result suggests a value for the trigger address correction that should then be entered in the system test initialization menu. This routine scans through both CLK1 values and DE trigger addresses to find a TSM input pattern of 128 channels enabled. The CLK1 scan range and steps may be controlled by system intializaion menu option, but the routine will always scan DE addresses 0 through 255. MTC module Trigger address offset calculation information and scan results are presented following the completion of the scan.

5.4.9.1 Initialization Menu Options -

- Accept prev bkt hit [T]
- Aux FIFO Enabled [F]
- FASTBUS FIFO Enabled [T]
- Ovflw trunc Enabled [F]
- CLK1 scan (0,63, 1, F)

5.4.9.2 Debugging Menu Options -

- Debugging mode switch [F]
- Log errors to file switch (rpx only) [F]

- Print errors to terminal [F]

CHAPTER 6

REFERENCE DOCUMENTS

The following is a brief listing of supporting hardware and software product documents. Document HN102 contains a complete listing of SSD hardware documentation.

- PORT_MGR - PN406
- FSCC - HN96
- RPX - PN384
- FSR_FSCC (FASTBUS Standard Routines) - PN416

CHAPTER 7

RELEASE NOTES

7.1 SYSTEM TESTS V1.0

7.1.1 Hardware Versions

- PC - Ver 2
- DE - Rev 2
- SE - Rev A
- MTC - Rev C
- TSM - July 1990
- LS - June 1990
- FSCC - PC2, PC3 with board revisions to date
- SSD Backplane - Rev A

7.1.2 Executables

- SYST.ABS - System Tests described by this document, PN436. The target for this executable is the FSCC.
- SYST_RPX.EXE - Same as above, but built for execution on the VAX using Remote Procedure Calls over a serial or Ethernet connection to the FSCC.
- SYST_RPX_DBG.EXE - Same as SYST_RPX.EXE, but built to enable VAX debugging.

7.1.3 Remote Procedure Call Executables For FSCC

- o FSR.ABS - Serial communication version of FASTBUS Standard Routine server.

- o FSR_ETH.ABS - Ethernet communication verson of FASTBUS Standard Routine server.

7.2 SYSTEM TESTS V1.1

7.2.1 Bug Fixes

- o Stop on error switch

  This switch, which applies to the PC test counter tests and TSM as PC tests, caused test execution to stop after every trigger in V1.0. Initialization corrections allow this option to function as described in the section labeled "Debugging options menu" above.

- o TSM as PC tests

  In version V1.0 TSM input data pattern scrambling was enabled. The scrambling function is disabled for version V1.1.

- o General readout loop

  Several problems were corrected that caused this routine to loop incorrectly over CLK1 and DAC scan ranges.

- o Print errors to terminal

  An initialization of an internal error counter was moved so that previous hit flag errors are now displayed if the Print errors to terminal Debugging menu option is enabled.

- o Default system tests

  Trigger address correction is no longer set in this routine. The value used for the trigger address correction is the value displayed by the System initialization menu.

- o SYST_RPX

  Several tests required additional calls to the

SWITCH_CRATE routine to cover error and no error conditions.

7.2.2 New Features

- o Tigger address scan

  This feature was added to facilitate determination of the correct trigger address correction for the TSM-LSM tests using the new SSD backplane. The trigger address correction for the PC test counter tests has been found to have the following relationship to the TSM-LSM tests trigger address correction value:

  TAC = trigger address correction value

  PC test counter tests TAC = ((TSM-LSM tests TAC) + 2)

- o PC test counter tests

  This routine was modified so that the SE word counter is written prior to each MTC trigger with the two's compliment of the expected word count. This modification causes the SE word count to be zero under normal circumstances. A non-zero word count provides a real time trigger for SE module debugging purposes. This modification handles the case of multiple PC-DE pairs, but is not designed to handle the case of SE overflow truncation enabled.

7.2.3 Changes In Behavior

1. FSCC pre verify event

   This routine will only print the message "defaulting to fscc_vfy_event" if the Debugging menu option Print errors to terminal is enabled.

2. Trigger address correction

   This value was initialized to 0 in V1.0. It is now intialized to -1 to reflect timing differences between the current and previous versions of the SSD backplane.

3. PC test counter - Stop on error

   The print statements were removed that duplicate the display of the test counter value and the previous and current bucket PC output patterns.

4. Default System tests

   The average CLK1 value determined by the CLK1/DAC scan is no longer passed to the PC Quick tests. The CLK1 value for the entire set of tests is taken from the value given by the System test initialization menu.

5. Set Primary Addresses

   The TSM PAD is now initialized to slot 17 (dec).

6. Set DAC buffer

   This System test initialization option for changing the PC DAC value shows the new initialization value of 100.

APPENDIX A

PC TEST COUNTER PATTERNS

PC tp bits	resultant PC output	#bits set in output
[0]	00000000 00000000 00000000 00000000	(0)
[1]	00000000 00000000 00000000 55555555	(16)
[2]	00000000 00000000 00000000 AAAAAAAA	(16)
[3]	00000000 00000000 00000000 FFFFFFFF	(32)
[4]	00000000 00000000 55555555 00000000	(16)
[5]	00000000 00000000 55555555 55555555	(32)
[6]	00000000 00000000 55555555 AAAAAAAA	(32)
[7]	00000000 00000000 55555555 FFFFFFFF	(48)
[8]	00000000 00000000 AAAAAAAA 00000000	(16)
[9]	00000000 00000000 AAAAAAAA 55555555	(32)
[10]	00000000 00000000 AAAAAAAA AAAAAAAA	(32)
[11]	00000000 00000000 AAAAAAAA FFFFFFFF	(48)
[12]	00000000 00000000 FFFFFFFF 00000000	(32)
[13]	00000000 00000000 FFFFFFFF 55555555	(48)
[14]	00000000 00000000 FFFFFFFF AAAAAAAA	(48)
[15]	00000000 00000000 FFFFFFFF FFFFFFFF	(64)
[16]	00000000 55555555 00000000 00000000	(16)
[17]	00000000 55555555 00000000 55555555	(32)
[18]	00000000 55555555 00000000 AAAAAAAA	(32)
[19]	00000000 55555555 00000000 FFFFFFFF	(48)
[20]	00000000 55555555 55555555 00000000	(32)
[21]	00000000 55555555 55555555 55555555	(48)
[22]	00000000 55555555 55555555 AAAAAAAA	(48)
[23]	00000000 55555555 55555555 FFFFFFFF	(64)
[24]	00000000 55555555 AAAAAAAA 00000000	(32)
[25]	00000000 55555555 AAAAAAAA 55555555	(48)
[26]	00000000 55555555 AAAAAAAA AAAAAAAA	(48)
[27]	00000000 55555555 AAAAAAAA FFFFFFFF	(64)
[28]	00000000 55555555 FFFFFFFF 00000000	(48)
[29]	00000000 55555555 FFFFFFFF 55555555	(64)
[30]	00000000 55555555 FFFFFFFF AAAAAAAA	(64)
[31]	00000000 55555555 FFFFFFFF FFFFFFFF	(80)
[32]	00000000 AAAAAAAA 00000000 00000000	(16)
[33]	00000000 AAAAAAAA 00000000 55555555	(32)

[34]	00000000	AAAAAAAA	00000000	AAAAAAAA	(32)
[35]	00000000	AAAAAAAA	00000000	FFFFFFFF	(48)
[36]	00000000	AAAAAAAA	55555555	00000000	(32)
[37]	00000000	AAAAAAAA	55555555	55555555	(48)
[38]	00000000	AAAAAAAA	55555555	AAAAAAAA	(48)
[39]	00000000	AAAAAAAA	55555555	FFFFFFFF	(64)
[40]	00000000	AAAAAAAA	AAAAAAAA	00000000	(32)
[41]	00000000	AAAAAAAA	AAAAAAAA	55555555	(48)
[42]	00000000	AAAAAAAA	AAAAAAAA	AAAAAAAA	(48)
[43]	00000000	AAAAAAAA	AAAAAAAA	FFFFFFFF	(64)
[44]	00000000	AAAAAAAA	FFFFFFFF	00000000	(48)
[45]	00000000	AAAAAAAA	FFFFFFFF	55555555	(64)
[46]	00000000	AAAAAAAA	FFFFFFFF	AAAAAAAA	(64)
[47]	00000000	AAAAAAAA	FFFFFFFF	FFFFFFFF	(80)
[48]	00000000	FFFFFFFF	00000000	00000000	(32)
[49]	00000000	FFFFFFFF	00000000	55555555	(48)
[50]	00000000	FFFFFFFF	00000000	AAAAAAAA	(48)
[51]	00000000	FFFFFFFF	00000000	FFFFFFFF	(64)
[52]	00000000	FFFFFFFF	55555555	00000000	(48)
[53]	00000000	FFFFFFFF	55555555	55555555	(64)
[54]	00000000	FFFFFFFF	55555555	AAAAAAAA	(64)
[55]	00000000	FFFFFFFF	55555555	FFFFFFFF	(80)
[56]	00000000	FFFFFFFF	AAAAAAAA	00000000	(48)
[57]	00000000	FFFFFFFF	AAAAAAAA	55555555	(64)
[58]	00000000	FFFFFFFF	AAAAAAAA	AAAAAAAA	(64)
[59]	00000000	FFFFFFFF	AAAAAAAA	FFFFFFFF	(80)
[60]	00000000	FFFFFFFF	FFFFFFFF	00000000	(64)
[61]	00000000	FFFFFFFF	FFFFFFFF	55555555	(80)
[62]	00000000	FFFFFFFF	FFFFFFFF	AAAAAAAA	(80)
[63]	00000000	FFFFFFFF	FFFFFFFF	FFFFFFFF	(96)
[64]	55555555	00000000	00000000	00000000	(16)
[65]	55555555	00000000	00000000	55555555	(32)
[66]	55555555	00000000	00000000	AAAAAAAA	(32)
[67]	55555555	00000000	00000000	FFFFFFFF	(48)
[68]	55555555	00000000	55555555	00000000	(32)
[69]	55555555	00000000	55555555	55555555	(48)
[70]	55555555	00000000	55555555	AAAAAAAA	(48)
[71]	55555555	00000000	55555555	FFFFFFFF	(64)
[72]	55555555	00000000	AAAAAAAA	00000000	(32)
[73]	55555555	00000000	AAAAAAAA	55555555	(48)
[74]	55555555	00000000	AAAAAAAA	AAAAAAAA	(48)
[75]	55555555	00000000	AAAAAAAA	FFFFFFFF	(64)
[76]	55555555	00000000	FFFFFFFF	00000000	(48)
[77]	55555555	00000000	FFFFFFFF	55555555	(64)
[78]	55555555	00000000	FFFFFFFF	AAAAAAAA	(64)
[79]	55555555	00000000	FFFFFFFF	FFFFFFFF	(80)
[80]	55555555	55555555	00000000	00000000	(32)
[81]	55555555	55555555	00000000	55555555	(48)
[82]	55555555	55555555	00000000	AAAAAAAA	(48)
[83]	55555555	55555555	00000000	FFFFFFFF	(64)
[84]	55555555	55555555	55555555	00000000	(48)
[85]	55555555	55555555	55555555	55555555	(64)
[86]	55555555	55555555	55555555	AAAAAAAA	(64)

PC TEST COUNTER PATTERNS

[87]	55555555	55555555	55555555	FFFFFFFF	(80)
[88]	55555555	55555555	AAAAAAAA	00000000	(48)
[89]	55555555	55555555	AAAAAAAA	55555555	(64)
[90]	55555555	55555555	AAAAAAAA	AAAAAAAA	(64)
[91]	55555555	55555555	AAAAAAAA	FFFFFFFF	(80)
[92]	55555555	55555555	FFFFFFFF	00000000	(64)
[93]	55555555	55555555	FFFFFFFF	55555555	(80)
[94]	55555555	55555555	FFFFFFFF	AAAAAAAA	(80)
[95]	55555555	55555555	FFFFFFFF	FFFFFFFF	(96)
[96]	55555555	AAAAAAAA	00000000	00000000	(32)
[97]	55555555	AAAAAAAA	00000000	55555555	(48)
[98]	55555555	AAAAAAAA	00000000	AAAAAAAA	(48)
[99]	55555555	AAAAAAAA	00000000	FFFFFFFF	(64)
[100]	55555555	AAAAAAAA	55555555	00000000	(48)
[101]	55555555	AAAAAAAA	55555555	55555555	(64)
[102]	55555555	AAAAAAAA	55555555	AAAAAAAA	(64)
[103]	55555555	AAAAAAAA	55555555	FFFFFFFF	(80)
[104]	55555555	AAAAAAAA	AAAAAAAA	00000000	(48)
[105]	55555555	AAAAAAAA	AAAAAAAA	55555555	(64)
[106]	55555555	AAAAAAAA	AAAAAAAA	AAAAAAAA	(64)
[107]	55555555	AAAAAAAA	AAAAAAAA	FFFFFFFF	(80)
[108]	55555555	AAAAAAAA	FFFFFFFF	00000000	(64)
[109]	55555555	AAAAAAAA	FFFFFFFF	55555555	(80)
[110]	55555555	AAAAAAAA	FFFFFFFF	AAAAAAAA	(80)
[111]	55555555	AAAAAAAA	FFFFFFFF	FFFFFFFF	(96)
[112]	55555555	FFFFFFFF	00000000	00000000	(48)
[113]	55555555	FFFFFFFF	00000000	55555555	(64)
[114]	55555555	FFFFFFFF	00000000	AAAAAAAA	(64)
[115]	55555555	FFFFFFFF	00000000	FFFFFFFF	(80)
[116]	55555555	FFFFFFFF	55555555	00000000	(64)
[117]	55555555	FFFFFFFF	55555555	55555555	(80)
[118]	55555555	FFFFFFFF	55555555	AAAAAAAA	(80)
[119]	55555555	FFFFFFFF	55555555	FFFFFFFF	(96)
[120]	55555555	FFFFFFFF	AAAAAAAA	00000000	(64)
[121]	55555555	FFFFFFFF	AAAAAAAA	55555555	(80)
[122]	55555555	FFFFFFFF	AAAAAAAA	AAAAAAAA	(80)
[123]	55555555	FFFFFFFF	AAAAAAAA	FFFFFFFF	(96)
[124]	55555555	FFFFFFFF	FFFFFFFF	00000000	(80)
[125]	55555555	FFFFFFFF	FFFFFFFF	55555555	(96)
[126]	55555555	FFFFFFFF	FFFFFFFF	AAAAAAAA	(96)
[127]	55555555	FFFFFFFF	FFFFFFFF	FFFFFFFF	(112)
[128]	AAAAAAAA	00000000	00000000	00000000	(16)
[129]	AAAAAAAA	00000000	00000000	55555555	(32)
[130]	AAAAAAAA	00000000	00000000	AAAAAAAA	(32)
[131]	AAAAAAAA	00000000	00000000	FFFFFFFF	(48)
[132]	AAAAAAAA	00000000	55555555	00000000	(32)
[133]	AAAAAAAA	00000000	55555555	55555555	(48)
[134]	AAAAAAAA	00000000	55555555	AAAAAAAA	(48)
[135]	AAAAAAAA	00000000	55555555	FFFFFFFF	(64)
[136]	AAAAAAAA	00000000	AAAAAAAA	00000000	(32)
[137]	AAAAAAAA	00000000	AAAAAAAA	55555555	(48)
[138]	AAAAAAAA	00000000	AAAAAAAA	AAAAAAAA	(48)
[139]	AAAAAAAA	00000000	AAAAAAAA	FFFFFFFF	(64)

[140]	AAAAAAAA	00000000	FFFFFFFF	00000000	(48)
[141]	AAAAAAAA	00000000	FFFFFFFF	55555555	(64)
[142]	AAAAAAAA	00000000	FFFFFFFF	AAAAAAAA	(64)
[143]	AAAAAAAA	00000000	FFFFFFFF	FFFFFFFF	(80)
[144]	AAAAAAAA	55555555	00000000	00000000	(32)
[145]	AAAAAAAA	55555555	00000000	55555555	(48)
[146]	AAAAAAAA	55555555	00000000	AAAAAAAA	(48)
[147]	AAAAAAAA	55555555	00000000	FFFFFFFF	(64)
[148]	AAAAAAAA	55555555	55555555	00000000	(48)
[149]	AAAAAAAA	55555555	55555555	55555555	(64)
[150]	AAAAAAAA	55555555	55555555	AAAAAAAA	(64)
[151]	AAAAAAAA	55555555	55555555	FFFFFFFF	(80)
[152]	AAAAAAAA	55555555	AAAAAAAA	00000000	(48)
[153]	AAAAAAAA	55555555	AAAAAAAA	55555555	(64)
[154]	AAAAAAAA	55555555	AAAAAAAA	AAAAAAAA	(64)
[155]	AAAAAAAA	55555555	AAAAAAAA	FFFFFFFF	(80)
[156]	AAAAAAAA	55555555	FFFFFFFF	00000000	(64)
[157]	AAAAAAAA	55555555	FFFFFFFF	55555555	(80)
[158]	AAAAAAAA	55555555	FFFFFFFF	AAAAAAAA	(80)
[159]	AAAAAAAA	55555555	FFFFFFFF	FFFFFFFF	(96)
[160]	AAAAAAAA	AAAAAAAA	00000000	00000000	(32)
[161]	AAAAAAAA	AAAAAAAA	00000000	55555555	(48)
[162]	AAAAAAAA	AAAAAAAA	00000000	AAAAAAAA	(48)
[163]	AAAAAAAA	AAAAAAAA	00000000	FFFFFFFF	(64)
[164]	AAAAAAAA	AAAAAAAA	55555555	00000000	(48)
[165]	AAAAAAAA	AAAAAAAA	55555555	55555555	(64)
[166]	AAAAAAAA	AAAAAAAA	55555555	AAAAAAAA	(64)
[167]	AAAAAAAA	AAAAAAAA	55555555	FFFFFFFF	(80)
[168]	AAAAAAAA	AAAAAAAA	AAAAAAAA	00000000	(48)
[169]	AAAAAAAA	AAAAAAAA	AAAAAAAA	55555555	(64)
[170]	AAAAAAAA	AAAAAAAA	AAAAAAAA	AAAAAAAA	(64)
[171]	AAAAAAAA	AAAAAAAA	AAAAAAAA	FFFFFFFF	(80)
[172]	AAAAAAAA	AAAAAAAA	FFFFFFFF	00000000	(64)
[173]	AAAAAAAA	AAAAAAAA	FFFFFFFF	55555555	(80)
[174]	AAAAAAAA	AAAAAAAA	FFFFFFFF	AAAAAAAA	(80)
[175]	AAAAAAAA	AAAAAAAA	FFFFFFFF	FFFFFFFF	(96)
[176]	AAAAAAAA	FFFFFFFF	00000000	00000000	(48)
[177]	AAAAAAAA	FFFFFFFF	00000000	55555555	(64)
[178]	AAAAAAAA	FFFFFFFF	00000000	AAAAAAAA	(64)
[179]	AAAAAAAA	FFFFFFFF	00000000	FFFFFFFF	(80)
[180]	AAAAAAAA	FFFFFFFF	55555555	00000000	(64)
[181]	AAAAAAAA	FFFFFFFF	55555555	55555555	(80)
[182]	AAAAAAAA	FFFFFFFF	55555555	AAAAAAAA	(80)
[183]	AAAAAAAA	FFFFFFFF	55555555	FFFFFFFF	(96)
[184]	AAAAAAAA	FFFFFFFF	AAAAAAAA	00000000	(64)
[185]	AAAAAAAA	FFFFFFFF	AAAAAAAA	55555555	(80)
[186]	AAAAAAAA	FFFFFFFF	AAAAAAAA	AAAAAAAA	(80)
[187]	AAAAAAAA	FFFFFFFF	AAAAAAAA	FFFFFFFF	(96)
[188]	AAAAAAAA	FFFFFFFF	FFFFFFFF	00000000	(80)
[189]	AAAAAAAA	FFFFFFFF	FFFFFFFF	55555555	(96)
[190]	AAAAAAAA	FFFFFFFF	FFFFFFFF	AAAAAAAA	(96)
[191]	AAAAAAAA	FFFFFFFF	FFFFFFFF	FFFFFFFF	(112)
[192]	FFFFFFFF	00000000	00000000	00000000	(32)

APPENDIX B

STANDARD ERROR SUMMARY

Total events read:	2	Data verify errors:	128
Events with errors:	2	Prev. bucket errors:	16
Sync errrors:	0	Missing hit errors:	96
Word count errors:	2	Extra hit errors:	16