AL-TR-1992-0160

ARMSTRONG LABORATORY

INFLIGHT EVALUATION OF AN ACOUSTIC ORIENTATION INSTRUMENT

Dan D. Fulgham
John L. Orr
Brian Mikiten

MacAulay Brown, Incorporated
3915 Germany Lane
Dayton, OH 45431

Southeastern Center for Electrical Engineering Education
1101 Massachusetts Avenue
St. Cloud, FL 34769

CREW SYSTEMS DIRECTORATE
2504 D Drive, Suite 1
Brooks Air Force Base, TX 78235-5104

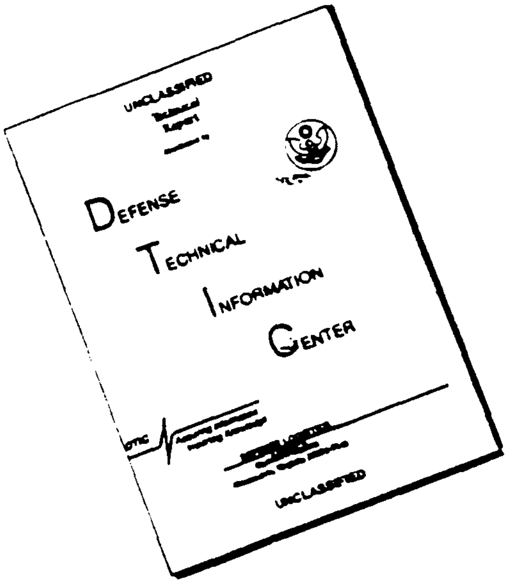
December 1992

Interim Technical Report for Period 16 November 1987 - 30 April 1991

Approved for public release; distribution is unlimited.

AIR FORCE MATERIEL COMMAND
BROOKS AIR FORCE BASE, TEXAS

DISCLAIMER NOTICE



UNCLASSIFIED

Technical Report

DEFENSE TECHNICAL INFORMATION CENTER

DTIC

UNCLASSIFIED

THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

NOTICES

This technical report is published as received and has not been edited by the technical editing staff of the Armstrong Laboratory.

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Office of Public Affairs has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This report has been reviewed and is approved for publication.

Kent K. Gillingham

KENT K. GILLINGHAM, M.D., Ph.D. Project Scientist	RONALD C. HILL, Lt Col, USAF, BSC Chief, Flight Motion Effects Branch

RICHARD L. MILLER, Ph.D.
Chief, Crew Technology Division

REPORT DOCUMENTATION PAGE

Form Approved

[illegible]

1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE
December 1992

3. REPORT TYPE AND DATES COVERED
Interim - 16 November 1987 – 30 April 1991

4. TITLE AND SUBTITLE

Inflight Evaluation of an Acoustic Orientation Instrument

5. FUNDING NUMBERS
C - F33615-87-C-0534
(Task 0002)
C - F33615-87-D-0609
(Task 0014)
PE - 61101F, 62202F
PR - 7930
TA - 14
WU - 8E

6. AUTHOR(S)

Dan D. Fulgham
John L. Orr
Brian Mikiten

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

MacAulay Brown, Inc.
3915 Germany Lane
Dayton, OH 45431

Southeastern Center for Electrical
Engineering Education (SCEEE)
1101 Massachusetts Avenue
St. Cloud, FL 34769

8. PERFORMING ORGANIZATION REPORT NUMBER

SwRI Project No. 12-2301
Subcontract No. 1107-02-08SWA

9. SPONSORING

Armstrong Laboratory
Crew Systems Directorate
2504 D Drive, Suite 1
Brooks Air Force Base, TX 78235-5104

AL-TR-1992-0160

11. SUPPLEMENTARY NOTES

Armstrong Laboratory Technical Monitor: Dr. Kent K. Gillingham, (210) 536-3521

12a. DISTRIBUTION

Approved for public release; distribution is unlimited.

Subcontractor (Southwest Research Institute) provides engineering data and system software description for the Acoustic Orientation Instrument (AOI), which they developed for inflight testing. The overall scheme of operation of the system involves inputting a frame of scaled flight data (airspeed, bank angle, vertical velocity, etc.) from the Flight Instrument Package (FIP) over an RS232 serial link to the AOI, conversion of the flight data to corresponding output voltage waveforms by the AOI, and delivery of those outputs to stereo headphones to generate an auditory display of the flight parameters of interest. This report constitutes an operation and maintenance manual for the AOI, including system programming in the FORTH computer language.

14. SUBJECT TERMS

Acoustic orientation
Aural displays
Flight instrumentation

400

17. SECURITY CLASSIFICATION OF REPORT

Unclassified	Unclassified	Unclassified	UL

TABLE OF CONTENTS

Appendices:

INTRODUCTION

Selected flight parameters of the aircraft are monitored by the Flight Information Package (FIP). Upon a request from the Acoustic Orientation Instrument (AOI), the FIP sends a frame of scaled data corresponding to the current flight parameters over a RS232 serial link to the AOI. Figure 1 shows a diagram of the system.

The audio control circuitry of the AOI provides direct microprocessor control over six channels of waveform generation. Scaled parameters from the FIP are mapped to an output waveform heard by the pilot through stereo earphones. A simple example follows: Heading deviation (the deviation of actual heading relative to a set heading) is used to help the pilot stay on a pre-determined course. This heading deviation information is used to control the position of the acoustic signal across the pilot's head by adjusting the signal to the channels of the pilot's stereo headset. A course deviation of 10 degrees or greater results in a monophonic signal in the ear opposite the deviation . As the pilot corrects the error, the signals become more centrally located between the left and right ears, until finally there is an equal amplitude between the pilot's left and right ears and the auditory image is centered.

Other aircraft parameters which may be presented as transformations mapped to acoustic signals include airspeed, angle of attack, vertical velocity, altitude, and pitch and roll angle. Input parameters may be represented as changes in waveform frequency, amplitude, modulation, duration, attack, and decay times. The system software is written in the FORTH computer language and is designed for flexibility over parameter minimums, maximums, activation values, and types of signals used for stimulus. This system allows changes in control values to remap the acoustic signals with little knowledge of the theory of the system or programming.

SOFTWARE

Acoustic Orientation Aid Software

The software for the AOI has been provided in three formats: 1) source code listing in Appendix H 2) MS-DOS "non-document" text files on 1.2 Mbyte 5 1/4 inch floppy disk and 3) 720 Kbyte 3 1/2 inch disks.

Software Operations Overview

A MS-DOS laptop computer with a communications program such as CrossTalk XVI or Mirror III should be used to load the software into the AOI and monitor its operation.

A batch file for Mirror is included which SENDS modules to the AOI. If another communications program is used, it should be set to require character echo and wait for a CR/LF before sending the next line.

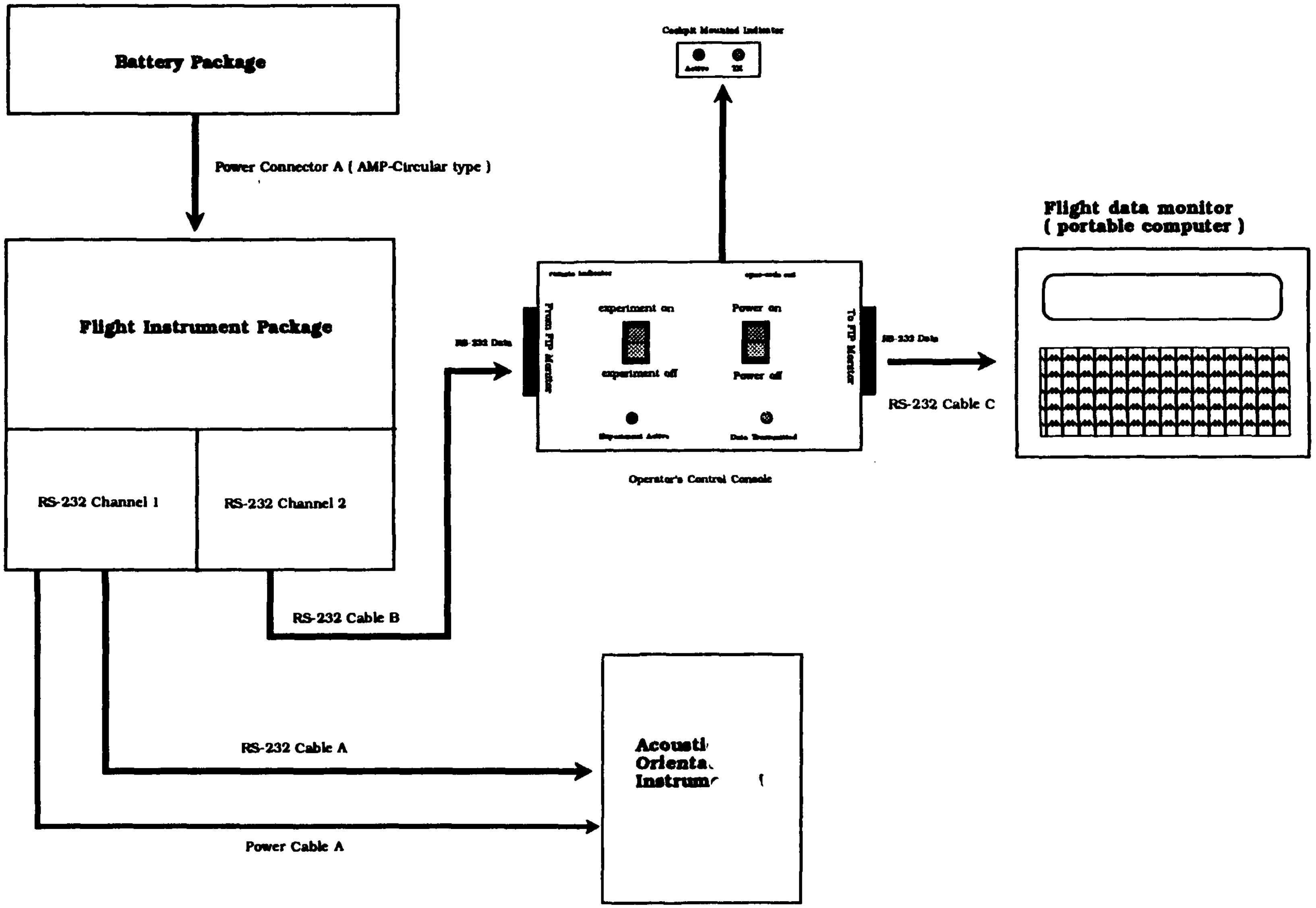


FIGURE 1. AOI/FIP SYSTEM CONFIGURATION.

Once the AOI software has been loaded, the laptop functions as a terminal to the AOI. The delivered DEMO module can stream the values read from the FIP to the screen.

Each of the AOI functional components - for example DO.AIRSPEED, which translates the data from the airspeed channel of the FIP into tone frequency - has several variables associated with it. Two of the airspeed related variables are AIRSPEED.MAX.FREQ. and AIRSPEED.MIN.FREQ. These variables correspond to the maximum and minimum frequencies generated by the airspeed channel. The characteristics of the AOI may be modified by simply placing a new value in the variable and starting the program. The functional components and their associated variables are described in detail in the section Acoustic Transformations.

Software integrity is only maintained by avoiding modifications of the source code. Changes in the variable settings should be made in the file DEMO.DO using a text editor.

Operating Sequence

The FIP and AOI computer programs must be loaded into the respective systems before a flight occurs. First, secure the battery package to the aircraft seat channels. Next, mount the FIP to the top of the battery package and engage the holding straps. This places the FIP in the correct configuration for flight. Following the steps outlined below will run the system.

- Turn the FIP control switches located on the to panel of the FIP to the off position.

- Connect the battery pack power connector labeled Power Cable A to the FIP and lock in place

- Connect RS-232 Cable A to the AOI but not the FIP

- Connect RS-232 Cable B to the Inflight contoller

- Connect RS-232 Cable C from the Inflight controller to the Flight data monitor

- Connect the cockpit mounted remote indicator

- Connect the AOI Power Cable from the FIP to the AOI

- Turn on the Inflight controller power and verify battery power

- Turn on the FIP main power switch

- Turn on the Gyro power

- Turn on the FIP computer power

- Connect the downloading computer to port A of the FIP

- Turn on the downloading computer and start MIRROR

- Confirm the <OK> from the FIP computer *

- Download the FIP computer program

- Hook up the AOI RS-232 Cable A to the FIP Port A

- Hook up the downloading computer RS-232 cable to the AOI terminal port.

- Confirm the OK from the AOI **

- Download the code from the computer to the AOI (see below)

- Start the AOI program

- Turn the experiment-on switch to the on position

- Confirm data streaming and the TX light at a rate of approximately 1Hz. ***

- Start the recording system and the flight

* If an <ok> is not present, the connections are faulty or the FIP computer is not working. If the green light on the FIP computer is on, the connections are faulty.

** If an <ok> is not present, check the power connection and press the red reset button on the front panel of the AOI. If this does not correct the problem, the AOI or FIP power system may require service.

*** If data is not streaming or Tx light is not at 1 Hz, the FIP may not be executing its program. Check the download program and the status of the FIP computer system.

AOI Code loading instructions:

1. Start the communications program. The settings are 9600 baud, 8 bits 1 stop hit, no parity, character echo.

2. Line Wait = CTRL J.

3. From the MIRROR command line type: DO STARTUP and press the ENTER key.

4. The program modules will then load into the AOI.

5. Normal operation of the AOI is indicated by an OK prompt.

NOTE: BE CERTAIN THAT THE CAPS KEY ON THE LAPTOP IS DEPRESSED. ALL COMMANDS TO THE AOI MUST BE IN UPPER CASE!

6. Type AOI and press ENTER. In a few seconds, data values from the FIP will start streaming to the laptop display.

7. Pressing any key will stop the AOI program. Typing AOI and pressing ENTER will partially re-initialize the system and start the program. Variables which have been changed at the OK prompt will now be in effect until changed or until the program is reloaded.

Values Displayed When AOI Operates

The values displayed are scaled data values from the FIP. The figures in the section Acoustic Transformations show the relationships between the readings on the cockpit instruments and the scaled data.

Reading and Setting Variables

To see the value of a variable you must do three things: 1) obtain the location of the variable by typing its name, 2) fetch the value of the variable by typing @, and 3) display the variable by typing a period. The OK prompt means the AOI is ready to accept a command.

For example:

```
OK AIRSPEED.MIN.FREQ @ .

40

OK
```

shows that the current value of the variable AIRSPEED.MIN.FREQ. is 40.

To temporarily change the value of a variable, three steps are again required: 1) type the new value and a space, 2) type the name of the variable and a space, and 3) type ! and Enter. Each of these entries must be separated with a blank space.

For example to make the lowest fre·quency for airspeed 200 Hz:

```
OK 200  HZ  AIRSPEED.MIN.FREQ  !

OK
```

and the value is changed. HZ is a word which translates frequency to period.

Changing the Value of a Variable in DEMO

The same commands may be used in DEMO to change the values of a variable. Use a text editor in the laptop to edit the file DEMO to contain the command.

As in the airspeed example above, type:

```
200  HZ AIRSPEED.MIN.FREQ  !
```

then save the modified DEMO file. There is no OK here because we are editing a text file, not interacting with the interpreter. When DEMO is reloaded, the new value will be in effect. The advantage of this approach is that the program itself is not changed, it is easy to keep track of where changes have been made, and a complete re-loading of the program is not required.

Software Vector Hooks

DEMO contains examples of the use of the Forth MAKE construct. As AOI was written, portions of the program were constructed as changeable hooks, also known as forward references or execution vectors. The MAKE command is used to plug a particular section of code into the hook. Initially, all hooks are connected to a routine called NOP (no operation), which does nothing. Inspection of the source code in Appendix H for the routine AOI in the module AOI will show that it is just an infinite loop with a series of named hooks. The declaration of one of these hooks is by the word DOER. The declaration of the execution vectors for the routine AOI in the module AOI, is just before the AOI routine itself.

If AOI was loaded, but not DEMO, and AOI typed to start it, the AOI microprocessor would be an infinite loop doing nothing. It would be necessary to either remove power from the system and reload the software or press the reset button on the AOI in order to stop the execution.

In DEMO the phrase:

MAKE HOOK4 GET.OUT

changes the operation of HOOK3 from NOP to the routine GET.OUT which checks for depression of any key and exits the loop if any key has been pressed.

Similarly, the phrase:

MAKE N.STARTUP SETUP

Causes the routine SETUP to be inserted into the hook N.STARTUP.

Hooks can be turned off by setting them to execute the routine NOP as follows:

MAKE HOOK2 NOP

This phrase will stop the display of the scaled data values on the laptop.

In general, the vectors can be hooked and unhooked as desired, with one exception. The routines for airspeed (DO.AIRSPEED) and for vertical velocity (DO.VERTICALVELOCITY) are related since the vertical velocity routine controls the amplitude of the airspeed tone.

Acoustic Transformations

The software modules for the various functions work in a consistent manner. There is a range of scaled data values defined by endpoints and a range of AY-8930 control parameters for an acoustic dimension defined by endpoints.

The software linearly scales the data value to the AY-8930 parameter value and loads the appropriate AY-8930 register to set the corresponding acoustic dimension. The rest of the code in the function routines is primarily logic to prepare for the appropriate scaling.

The following discusses the relationships between the scaled data from the FIP and parameters for the acoustic signals. The specific values described are those originally delivered with AOI-II. As described above, it is intended that these parameters be changed by modifying the file DEMO. It is important to understand that the encoding relationships described here are only and example of the set of possible encoding schemes. The source code containing FIP data to AOI scaled data is found in Appendix H. One must carefully read the source code to understand the details of the encoding examples. In figures 2-7, the upper line indicates the range of scaled data from the FIP and the corresponding physical parameter values, while the lower line displays the range of acoustic parameters.

AIRSPEED

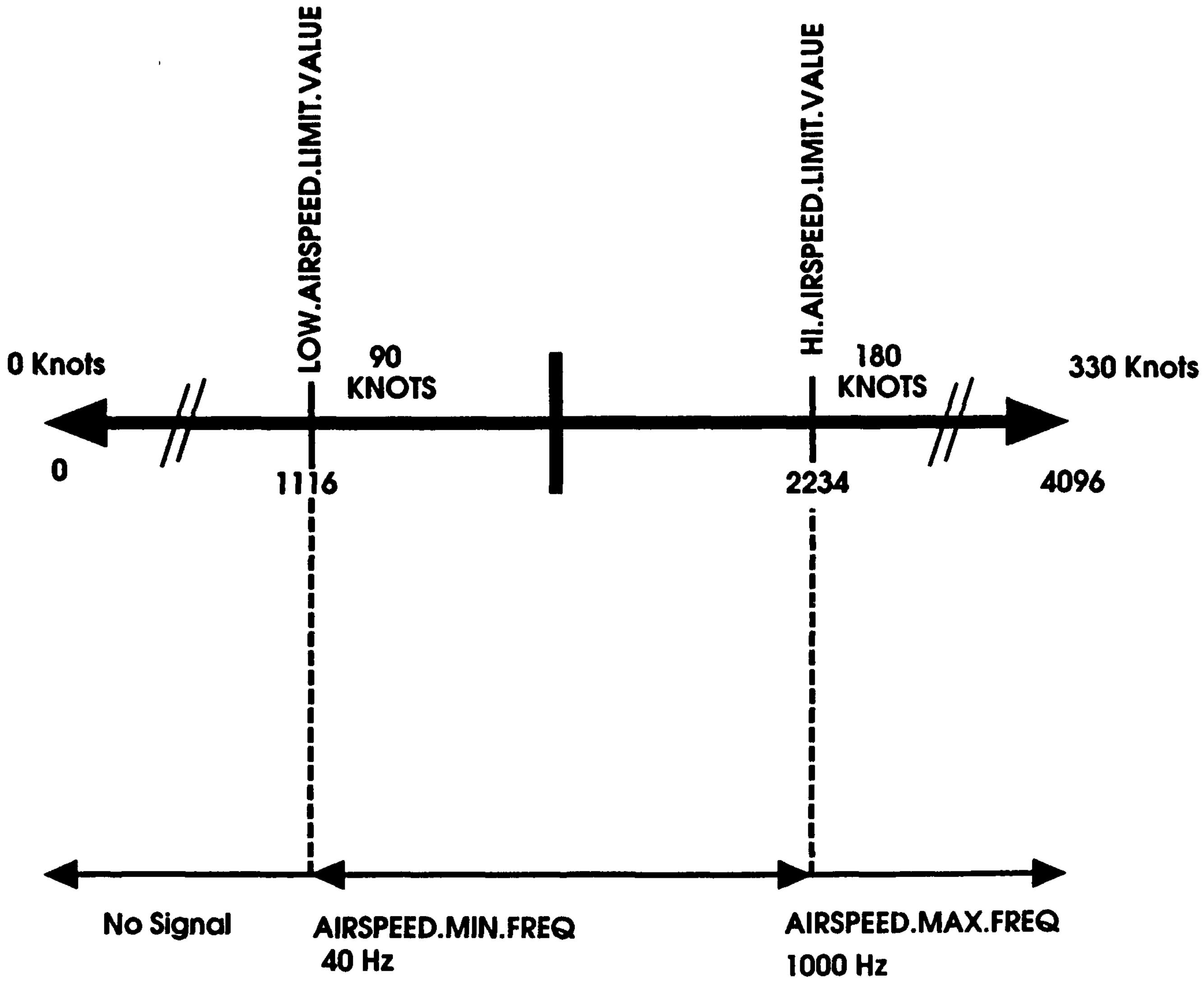


FIGURE 2. AIRSPEED INPUTS, SCALED DATA VALUES AND THEIR ACOUSTIC TRANSFORMATION.

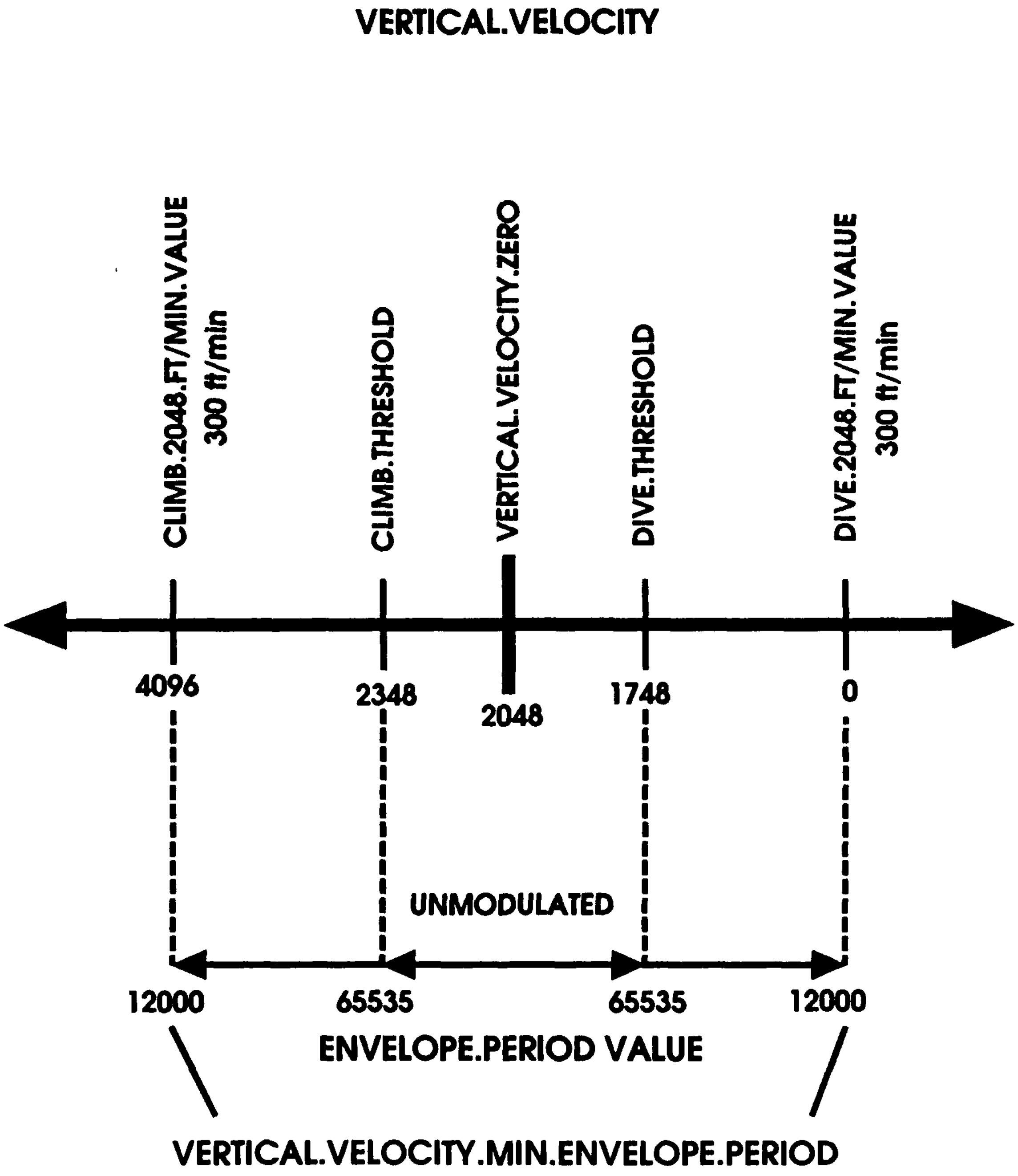


FIGURE 3. VERTICAL VELOCITY INPUTS, SCALED DATA VALUES, AND THEIR ACOUSTIC TRANSFORMATION.

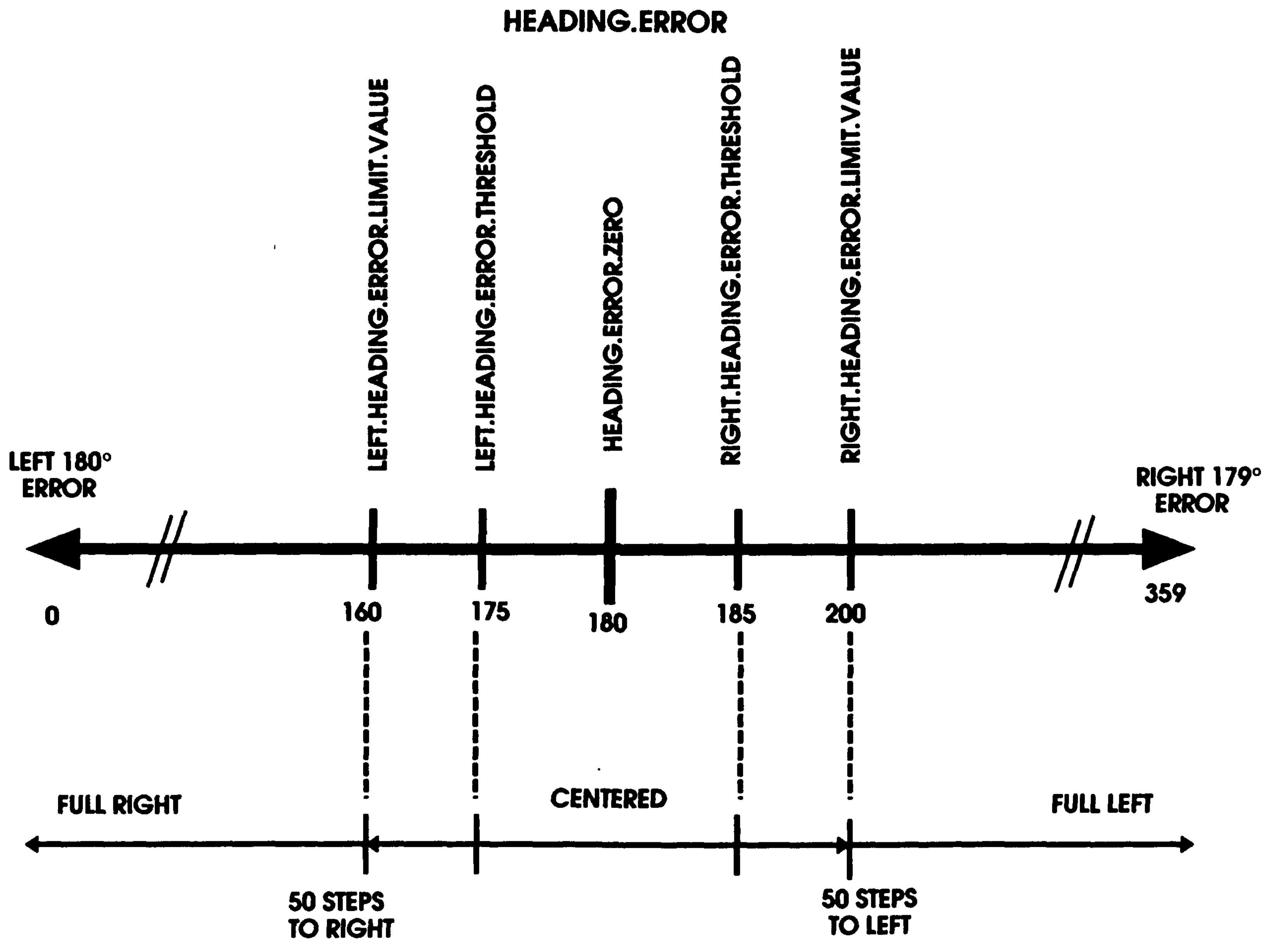


FIGURE 4. HEADING ERROR, SCALED DATA VALUES AND THEIR ACOUSTIC TRANSFORMATION.

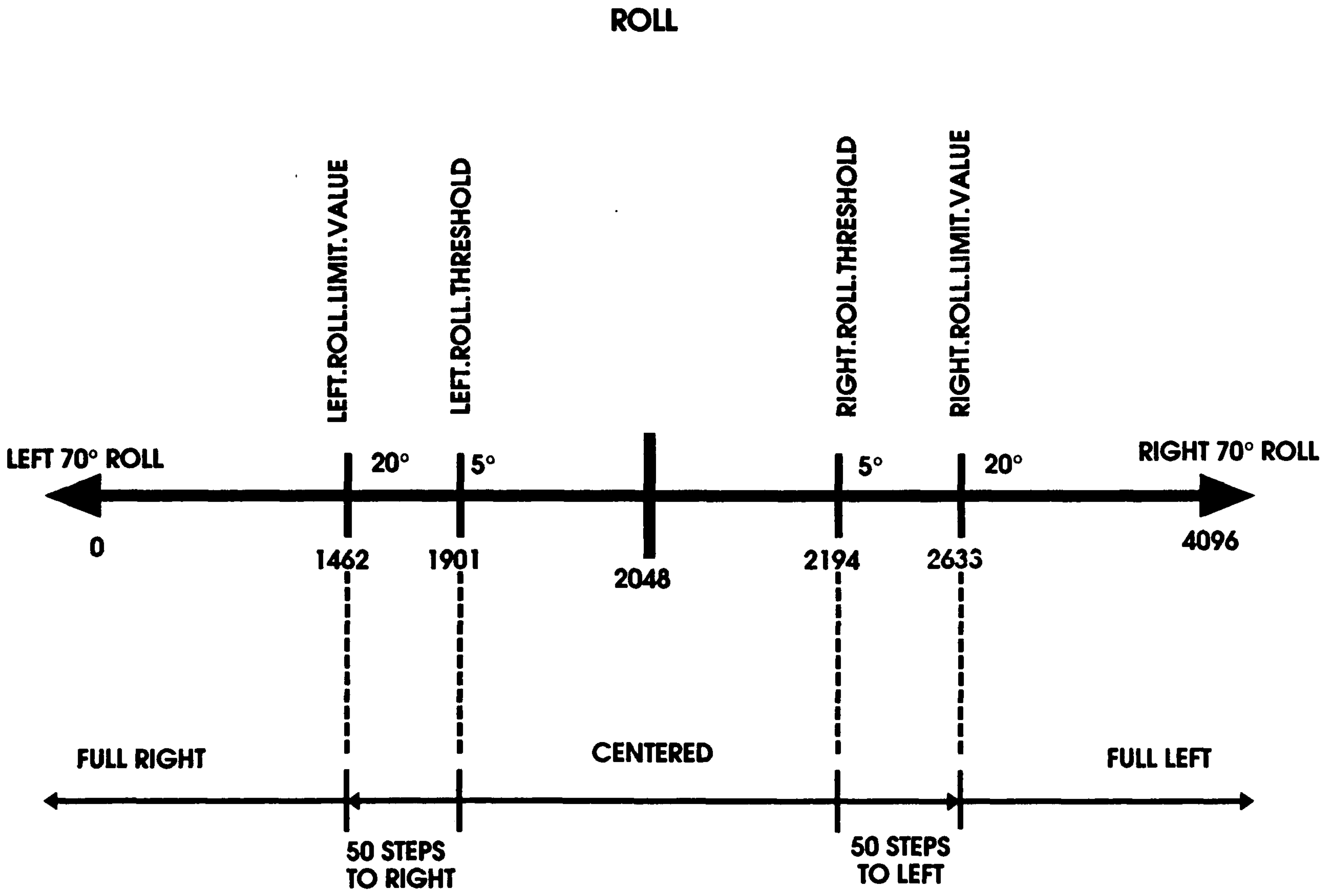


FIGURE 5. ROLL INPUTS, SCALED DATA VALUES, AND THEIR ACOUSTIC TRANSFORMATION.

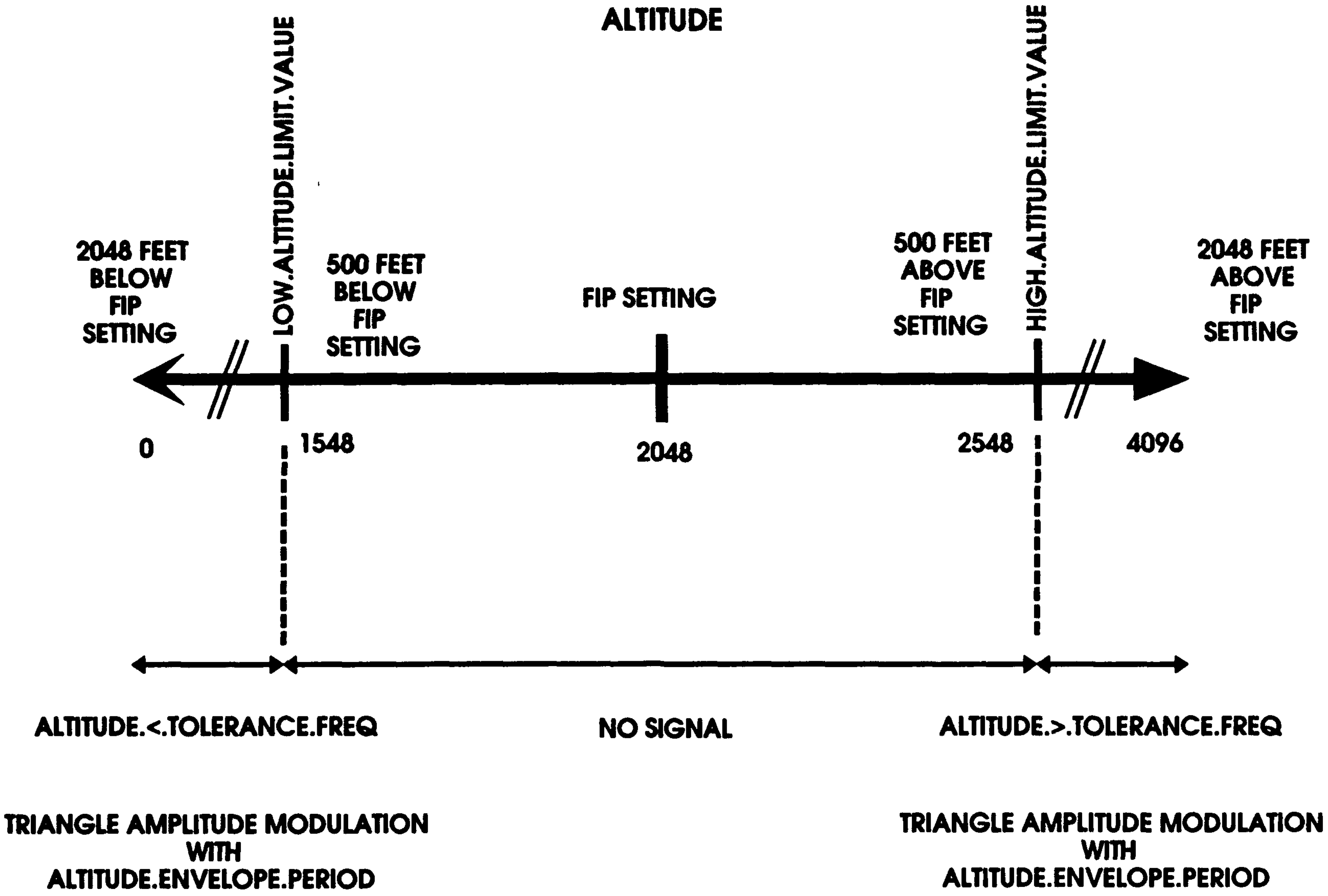

ALTITUDE
2048 FEET BELOW FIP SETTING
LOW.ALTITUDE.LIMIT.VALUE
500 FEET BELOW FIP SETTING
FIP SETTING
500 FEET ABOVE FIP SETTING
HIGH.ALTITUDE.LIMIT.VALUE
2048 FEET ABOVE FIP SETTING
0
1548
2048
2548
4096
ALTITUDE.<.TOLERANCE.FREQ
NO SIGNAL
ALTITUDE.>.TOLERANCE.FREQ
TRIANGLE AMPLITUDE MODULATION WITH ALTITUDE.ENVELOPE.PERIOD
TRIANGLE AMPLITUDE MODULATION WITH ALTITUDE.ENVELOPE.PERIOD

FIGURE 6. ALTITUDE INPUTS, SCALED DATA VALUES, AND THEIR ACOUSTIC TRANSFORMATION.

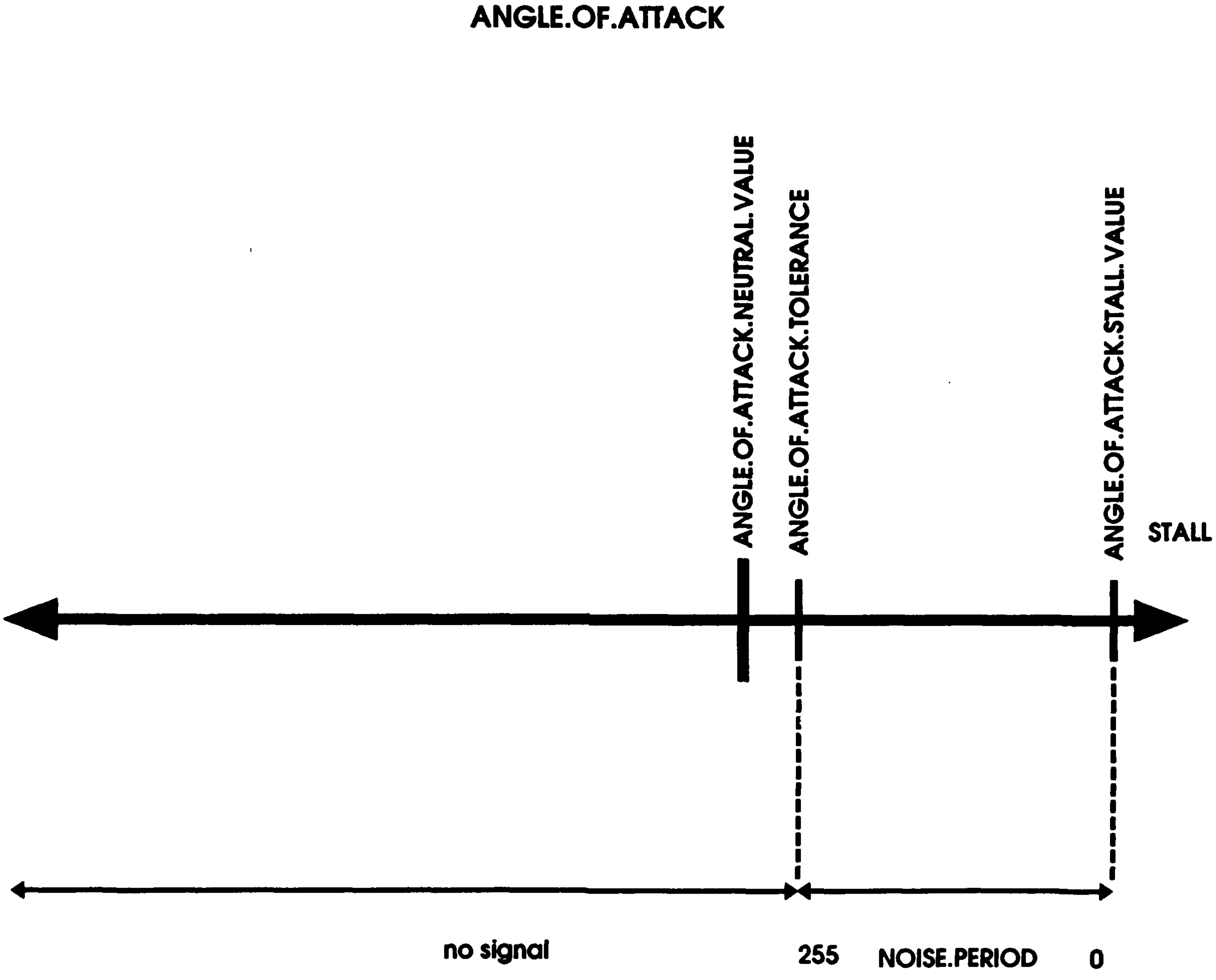

ANGLE.OF.ATTACK
ANGLE.OF.ATTACK.NEUTRAL.VALUE
ANGLE.OF.ATTACK.TOLERANCE
ANGLE.OF.ATTACK.STALL.VALUE
STALL
no signal
255
NOISE.PERIOD
0

FIGURE 7. ANGLE OF ATTACK INPUTS AND THEIR ACOUSTIC TRANSFORMATION.

Airspeed

Airspeed is encoded into the frequency of a square wave output which increases in frequency as airspeed increases. The upper line in Figure 2 shows that the range of scaled airspeed values is from 0 to 330 knots, corresponding to scaled data values from 0 to 4096. The lower line shows that the AIRSPEED.MIN.FREQ is 40 Hz and the AIRSPEED.MAX.FREQ is 1000 Hz. This range of frequencies is mapped linearly between the scaled data values corresponding to LOW.AIRSPEED.LIMIT.VALUE (current scaled data value of 1116 corresponding to 90 knots) and HI.AIRSPEED.LIMIT.VALUE (current scaled data vâlue of 2235 corresponding to 180 knots). Changes in these variables will change the scaling. If LOW.AIRSPEED.LIMIT.VALUE was set to 1861 (150 knots) and HI.AIRSPEED.LIMIT.VALUE was set to 2110 (170 knots) and the other variables unchanged, the new airspeed range of 20 knots would be scaled from 40 to 1000 Hz. Similarly, if the airspeed limit values were at their original values and AIRSPEED.MAX.FREQ was changed to 1500 Hz, airspeeds from 90 to 180 knots would be mapped onto a 50% larger frequency range.

Vertical Velocity

The direction of vertical velocity (climb or dive) is encoded as a change in amplitude modulation (attack or decay) of the airspeed tone. The magnitude of vertical velocity is encoded into the period of the amplitude modulation with long periods corresponding to low vertical velocities and short periods corresponding to high vertical velocities. There are selectable vertical velocities beyond which the modulation period does not change. As shown in Figure 3 the scaled values of vertical velocity from the FIP range from 0 (2048 feet/minute dive) through 2048 (level flight) to 4096 (2048 feet/minute climb). The range between the DIVE.THRESHOLD of 1748 (300 feet/minute climb) and the CLIMB.THRESHOLD of 2348 (300 feet/minute dive) define a null region where there is no amplitude modulation of the airspeed tone. Between the CLIMB.THRESHOLD and the CLIMB.2048.FT/MIN.VALUE, the period of amplitude modulation (decays starting with the maximum intensity and decreasing in 32 logarithmic steps) changes from slow (the maximum envelope period of 65535) to fast (the minimum envelope period of 12000). The signal changes between the DIVE.THRESHOLD and the DIVE.2048.FT/MIN.COUNTS are analogous except when the amplitude modulation consists of attacks starting with the lowest intensity and stepping to the maximum. The VERTICAL.VELOCITY.MINIMUM.ENVELOPE.PERIOD is set to 12000 in order to permit discrimination of the decays and attacks. Although the demo program has symmetrical thresholds and limit values, this is not a requirement.

Heading.error

Heading error is relative to the heading set on the FIP. Heading errors move the apparent location of the acoustic signal in a direction opposite the deviation. Figure 4 shows the encoding of heading error. Scaled heading error values from the FIP range from 0 (180 degrees left heading error) to 180 (on course) to 359 (179 degrees right heading error). Between LEFT.HEADING.ERROR.THRESHOLD (scaled value of 175 which corresponds to a 5 degree left heading error) and RIGHT.HEADING.ERROR.THRESHOLD (185 which corresponds to a 5 degree right heading error) the acoustic signal is centered. Between LEFT.HEADING.ERROR.THRESHOLD and LEFT.HEADING.ERROR.LIMIT.VALUE (160 corresponding to a 20 degree left heading error) there are 50 position steps toward an apparent location of the acoustic signal on the right. Left heading errors larger than the LEFT.HEADING.LIMIT.VALUE are positioned at the maximum location to the right. Heading deviations to the right are similar in concept, except that the apparent acoustic signal location moves to the left with increasing deviations.

Roll

Encoding of roll is directly analogous to the encoding for heading error. Figure 5 shows the particular values of the values in the roll variables used in the supplied version of AOI. Because heading error and roll both encode to position within the head on the interaural axis, only one may be connecteò to its software vector hook at a time.

Altitude Deviation

Altitude deviation is relative to the value set on the FIP. Figure 6 shows the altitude encoding scheme. Altitude deviation covers a range between 2048 feet below the FIP setting (scaled data value of 0) and 2048 above the FIP setting (scaled data value of 4096). Above the HIGH.ALTITUDE.LIMIT.VALUE of 2548 (500 feet above FIP setting), a high pitched triangle modulated annunciatior is added to the acoustic signal. ALTITUDE.>.TOLERANCE.FREQ is the variable which controls the frequency of the annunciator and ALTITUDE.ENVELOPE.PERIOD is the variable which controls the period of the annunciator. Below the LOW.ALTITUDE.LIMIT.VALUE of 1548 (500 feet below FIP setting) the annunciator is added to the acoustic signal with a frequency controlled by the variable ALTITUDE.<.TOLERANCE.FREQ.

Angle.Of.attack

Encoding of angle of attack is shown in Figure 7. Because there is not a functioning angle of attack vane on our test aircraft, this function is not yet implemented. The acoustic signal for angle of attack is intended to provide a warning of an impending stall. ANGLE.OF.ATTACK.NEUTRAL.VALUE is an offset. If the scaled data value for angle of attack is less than ANGLE.OF.ATTACK.NEUTRAL.VALUE plus ANGLE.OF.ATTACK.THRESHOLD, there is no acoustic output for angle of attack. If the scaled data value is greater than ANGLE.OF.ATTACK.THRESHOLD but less than the stall value, angle of attack deviation from threshold is mapped onto the period of a digital noise generator in one of the AY-8930 programmable sound chips. Just above the threshold, the acoustic signal is randomly spaced clicks with a low average rate. As deviation above threshold increases, the average rate increases until just before a stall, the acoustic signal resembles wide band noise. When data value scaling is known, it is merely necessary to insert appropriate scaled values into the variables in the module DEMO as described above.

Changing AOI Function Routines

The diagrams in the section Acoustic Transformations show the important variables for the different functional routines. A great deal of flexibility in the operation of the AOI is available by careful changing of the values of the variables. Alternatively, by inappropriate changes in the variables, it is possible to render the AOI non-functional until the original code is reloaded.

Substituting AOI Function Routines

Using the delivered source code as a guide, new routines can be written in the MAX-FORTH language of the AOI microprocessor. For example, a new vertical velocity routine, for example, DO.NEW.VERTICAL.VELOCITY, could be written and loaded after the file AOI. By editing the line in file named DEMO from

```
MAKE    N.DO.VERTICAL VELOCITY     DO.VERTICAL.VELOCITY
```

to

```
MAKE    N.DO.VERTICAL VELOCITY     DO.NEW.VERTICAL.VELOCITY
```

the new routine would be hooked into the AOI-II software.

Development Environment

The AOI software was developed on IBM-PC compatibles. Either Crosstalk XVI from Digital Communications Associates or Mirror from Softklone Associates were used as communications programs. Mirror includes a screen editor which allows the editing of files while still in serial communications with the AOI.

Delivered Files

The following files are on the MS-DOS format disk delivered with the AOI system. The first group of files are modules which make up the AOI software. These files are listed in Appendix H. These source code files have comments which clarify the actions being performed. Since the source code is written in MAX-FORTH, the language embedded in the ROM of the microprocessor of the AOI, one must be proficient in FORTH to understand the code in detail. However, because of the ability to use meaningful names for routines in Forth, one does not need to be an expert to read the Forth source code and understand the function of the components of the routines.

Forth has a variety of unique characteristics. The language is a postfix notation similar to that of most Hewlett Packard calculators. This means that first the operands are entered and then the operator. For example, to add four and six in Forth, you would type 4 6 + and then a period. (. is the print command) to print out the result of 10. The MAX-Forth reference manual delivered with the AOI includes an introduction to programming Forth. A second feature of Forth is the dictionary. Routines are called words, and as new words are defined they are entered into the dictionary. Because of its structure, Forth requires that any words used within another defining word must have been previously entered into the dictionary (unless it is a forward reference preceded by the word DOER). Another feature of Forth is that most parameter passing is done on a data stack instead of through static data structures. Forth is ideally suited for hardware oriented, real-time control problem applications such as the AOI, because routines (words) may be interactively tested and debugged.

The following source code files are listed in the order in which they are loaded into the AOI system.

SETUP	Configures the system memory and defines utilities
ARRAYS	Defines array and table words
CASE	Defines a case selection execution structure
CHIPCTRL	Hardware control primitives for the AY-8930s
DOERMAKE	Implements the forward reference mechanism
SELECTOR	Connects functional names to AY-8930 registers
VIEW8930	Controls AY-8930 shape/cycle registers
ENVELOPE	Dumps all AY-8930 registers in AOI context
CONVERT	Converts frequency to period
TIME6522	Uses 6522 as a timer
XICOR	Controls digital potentiometers
CONFIGUR	Sets up hardware
TRANSIT	Provides AOI 1&2 compatibility
ACIACOM	Provides communications to FIP
SHOFRAME	Displays scaled data

AOI	Performs the AOI transformations
SYSINITS	Initializes routines

The AOI module is the module relevant for understanding the operation of the AOI and seeing the meaning of the variables which are analogous to the trimmer potentiometers on traditional instruments. Skilled adjustment and change can enhance AOI performance.

The source code modules listed below contain comments and white space to enhance readability.

DEMO is the last module downloaded to the AOI and contains the following:

```
MAKE N.STARTUP SETUP  ( IN FILE SYSINITS )

MAKE HOOK1 SYNCH

( WAITS UNTIL 1 SECOND HAS ELAPSED SINCE TIMER STARTED )

( THE TIME FOR THE AOI PROCESSING IS WITHIN THE SECOND )

MAKE HOOK2 GET.DATA.FRAME

( PUTS A FRAME OF DATA INTO INPUT.DATA ARRAY )

( INPUT.DATA IS DEFINED IN FILE TRANSITS )

( GET.DATA.FRAME IS DEFINED IN ACIACOM )

MAKE HOOK3 SHOW.DATA.FRAME

( SHOWS THE DATA FRAME FROM INPUT.ARRAY )

MAKE HOOK4 GET.OUT

( GET.OUT BREAKS YOU OUT OF THE PROGRAM IF ANY KEY IS HIT )

( GET.OUT DEFINED IN FILE SETUP )

( MAKE N.DO.AOA          DO.AOA )

( AOA VANE IS NOT YET ON AIRPLANE )

MAKE N.DO.AIRSPEED          DO.AIRSPEED

( BOTH DO.ROLL AND DO.HEADING.ERROR USE THE HEADPHONE
        BALANCE )

( THEREFORE ONLY ONE CAN BE USED AT A TIME )

MAKE N.DO.ROLL              DO.ROLL

( MAKE N.DO.HEADING.ERROR     DO.HEADING.ERROR )

MAKE N.DO.ALTITUDE            DO.ALTITUDE

MAKE N.DO.VERTICAL.VELOCITY    DO.VERTICAL.VELOCITY
```

The modules above connect the routines to the hooks in the AOI program (resolves the forward references from NOP to the desired action. DEMO. is the appropriate place to add changes to set variables to new values.

LP.DOC explains how the listings were printed and contains the following:

This is the command line used with Norton's LP program to produce the HP Laserjet listings:

```
lp filespec /h60/w132/set:\aoi\lasercod/l25/r25
```

The file LASERCOD contains the setup for the HP Laserjet and consists of:

```
\027E\027(s16.66H)
```

(This information is only relevant if you have Norton Utilities and an HP-Laserjet printer).

SERIAL COMMUNICATIONS INFORMATION

Serial Communications

The following data frame is used when data is sent from the FIP to the AOI-II.

Frame Start Character (STX ascii 02)
Airspeed
Angle of Attach (not used)
Vertical Velocity
Heading Deviation (+ -)
Roll Angle
Pitch Angle
Altitude Deviation (+ -)
End of Frame Character (ETX - ascii 03)

Data is sent at 9600 baud, 8 data bits, 1 stop bit, no parity. The data are sent as 16 bit binary integers with a check sum.

SYSTEM OVERVIEW

Purpose

The Acoustic Orientation Instrument (AOI) has been designed as an acoustic signal synthesis and output device driven by the Flight Instrumentation Package (FIP). As a system, the AOI is directly connected through a serial communications line to the FIP and generates acoustic parameters based on the scaled values of flight parameters measured by the FIP while in flight.

The FIP-AOI system is designed to operate for up to 4 hours from a battery package that supplies both the AOI and the FIP. Audio output is provided to two stereo headphone jacks located on the front panel of the AOI. The AOI is housed in a 6x6x4 inch metal case and consists of six individual printed circuit boards.

Component Overview

68HC11/F Single Board Computer

The first component in the system is a 68HC11/F, single-board computer operating at 4 MHz from New Micros Inc., Dallas, Texas. The board runs the FORTH programming language as well as assembly code and contains a port replacement unit (PRU), three sockets for RAM, and a total addressable range of 64K, minus the space used by the FORTH language in ROM within the microprocessor. Also included on the board is a RS232 communications port for the console operating at 9600 baud, 8 bits no parity, one stop bit. In this implementation the board utilizes 32K of RAM and the Forth language in ROM. All components in the AOI-II are memory mapped on the bus using the JEDSTACK connector on the board. The bus contains 16 address lines, 8 data lines multiplexed, read/write signals, interrupt signals and other miscellaneous microprocessor generated signals. The clock used on the microprocessor board is used as the AOI system clock and drives all other boards. Bus addressing has been used to guarantee high speed operations and data transfer to all other boards in the system. The bus also provides one leg of the mechanical support for the entire AOI board set. Other information on the 68HC11/F board can be found in the Appendix B. Also included in Appendix B are the Max-Forth and 68HC11 manuals.

6522/8930 Board

The 6522/8930 board is used in the system to generate acoustic signals. Two General Instruments AY-8930s are located on the board and provide up to six independent channels of digital audio. Each 8930 is separately memory mapped and operates independently. During the development of the system it was found that the 8930s are not directly compatible with the bus structure of the JEDSTACK. To correct the interface signal problems, a Rockwell Semiconductor 65C22 was used. This single 65C22 provides two extra parallel ports and thereby the ability to synthesize a secondary bus used to interface the AY-8930s to the 68HC11/F.

Board Function

Signals from the bus are decoded by two 74HC688 8 bit bus comparators. The board is decoded in 16 bytes of RAM and can be located anywhere within the free memory space of the microprocessor. The dual 688s decode address lines A15-A4 based on a low address strobe on the bus. The subsequent P=Q line pulls down memory disable and provides the chip select line to the 6522. All subsequent reads and writes to the 6522 are done using standard bus nomenclature. The 6522 is used in this application to synthesize the 8 bit data bus going to both AY-8930s and the required control lines BDIR, BC1, Reset, A8 and A9. Because of this synthetic bus, transfer rates to the AY-8930s are substantially lower than that which could be achieved using direct bus access. Fortunately, in this application, higher speed access was not a requirement.

Port A on the 65C22 is used to provide data and address lines to both AY-8930s as shown in the schematic labeled AOI Digital Board Rev 3.0 in Appendix A. The 6522 pins CA1, CA2, CB1, CB2 are not used at this time. Port B synthesizes all of the control lines used for both of the AY-8930s. PB1 and PB2 are used to synthesize BDIR and BC1 on the 8930 chips. PB0 is used to select AY-8930-0 or AY-8930-1 . AY-8930-0 has its control line A8 tied high thereby using a low signal on PB 5 to select it. AY-8930-1 has bit A9 tied low, and uses a high signal on A8 to select it. This single line provides simple, quick chip select logic between the two devices and guarantees that bus contention between the two chips never occurs.

All of the parts on this board are clocked using the microprocessor's 2MHz clock. A secondary bus system is also provided on the AY-8930/6522 board. It consists of four rows of 35 pins located on the rear portion of the board and is designed for board-to-board interconnection. All signals from the 8930 as well as 12V, -12V, +5v, and ground are present on this bus. I/0 control ports from both AY-8930s are used to control mixer board functions on the audio mixer board (see that section).

The board-to-board interconnection bus is configured as four rows of 35 pins. Row 1, labeled C1, contains all four of the 8 bit ports found on the AY-8930s. Row 2, C2, is completely grounded to provided isolation and noise immunity to the system. Row 3, C3, carries on pins 1-6, all six channels of audio generated by the AY-8930s chips 0 and 1. Row 4, C4, contains ground, +12V, -12V and 5V. See the schematic for pin labeling on this bus. Silk screens for this board are provided in Appendix A. One 150% board layout is provided in Appendix A. A component list for this board is provided in Appendix G. A bit list for this board is shown on the following page.

AOI II Digital Potentiometer Control Bit List

The following list shows the AY-8930 output bits used to control the digital potentiometers. All bits are shown with the AY-8930 pin number and the function.

I0 = Chip 0

I1 = Chip 1

Negative logic is shown with a ~

Bit	Function	Description
IO-B0	U/~D	All U/D for the Xicors
IO-B1	INC Xicor Pot	Increment all Xicors
I1-AO	Pre-Mixer Pot	Level control
I1-A1	Noise Level	Noise level control
I1-A2	Balance	Balance Xicor control
I1-A3	Left Volume	Post balance gain
I1-A4	Right Volume	Post balance gain
I1-A5	----------	Not Used
I1-A6	HP Filter	Sets the pass band* tied to A7
I1-B0	Ch 0A	Audio output gain
I1-B1	Ch 0B	Audio output gain
I1-B2	Ch 0C	Audio output gain
I1-B3	all Chip 0	Mixer 0
I1-B4	Ch 1A	Audio output gain
I1-B5	Ch 1B	Audio output gain
I1-B6	Ch 1C	Audio output gain
I1-B7	All Chip 1	Mixer 1

Note: I0-A0 through I0-B7 and Ch 0A, 0B, 0C, are from chip zero (0) and I1-A0 through I1-B7 and Ch 1A, 1B, 1C are from chip one (1).

Audio Mixer Board

The Audio mixer board provides six channels of mix-down capability and final amplification of the audio signals generated by the 6522/8930 board. The board contains six channels of independent mixers with independent digitally controlled volume, two 3 to 1 mixers, mixers for four external signal sources with manually controlled potentiometers, computer-controlled balance circuitry, and final amplification. Also included on this board is a variable bandwidth filtered white noise source. Information about this board is in Appendix F.

Each of the audio channels from the AY-8930 board (A,B,C AY-8930-0, A',B',C' AY-8930-1) are provided to this board using the board to board bus interconnections described under the AY-8930 board section. The board provides several stages of modification to the audio signal generated on the AY-8930 board. The first stage, buffering, attenuation and amplification, is realized using a digital XICOR potentiometer. The potentiometer is located on the feedback loop of the Op-amp used to buffer the digital signal to this board. Gains up to 2X and attenuation to .5X can be achieved with the 100 step potentiometer. Individual control of each of the six channels and each of the gain/amplification/attenuation network is provided using AY-8930 control bits (see the AY-8930 board bit list). Once amplified or attenuated, the raw signals are provided to jumper blocks J1-J3 for AY-8930-1 or J5-J7 for AY-8930-0. Each jumper block provides the user the capability of mixing the signal within the output group of that chip, or separating the signal to be used in the post-balance portion of this board's circuitry. If used in the default configuration all signals from the single attenuation /amplification stages, will be routed to the 3:1 mixer on IC1. At this point, all three of the signals from AY-8930-0 will be routed to IC2 and all signals from AY-8930-1 will be routed to IC1. In this configuration, complete 3:1 mixing is achieved on one quad op-amp. The output of this 3:1 mixer is then sent to a 2:1, or final mixing stage, on IC3. IC3 also contains the amplifiers used in the external mixing sources referred to as pre-balance external mixers. Note that the output of both 3:1 mixers and the subsequent 4:1 include only monaural signals from the two AY-8930s and the two external signal sources. The external signal source op-amps, located on IC3, provide up to .2X attenuation or +2X amplification of the external signal source. R12 and R13 (components list appendix G) are used to match the impedance and set the gain of the circuit for the external signals to the internal circuitry used on this PC board.

The final portion of circuitry in the prebalance mixing chain, op-amp 4, located on IC3, allows the user to digitally control all AY-8930 and two external prebalance signal sources before they are processed by the balance circuit. Complete digital control of attenuation and gain of this combined signal is provided using another XICOR digital potentiometer and the control lines found on the external bus. Once the signals are mixed, a balance circuit, utilizing one XICOR digital potentiometer and one MC34082 dual op-amp allows the users to modulate the signal source across the head of the pilot. Near full attenuation on either channel is attained using this system. Note that unity gain is provided through this circuit.

Balance Circuit - A note

Note, at this point in the circuitry the user is allowed to route signals either prebalance or postbalance. If prebalance is selected, all signals are mixed through IC3 op-amp 4. If post balance is selected, signals are routed to jumpers 1-9 pins 6 or 4 and are processed post balance. As was the case with the prebalance signals, two external inputs are provided for the user. In contrast to the prebalance signals, the postbalance external mixers are either external left or external right. Crossover between the two channels is nonexistent. The final stage before power amplification mixes any signals routed from the prebalance mixers or external prebalance signal sources and the external post balance sources. Again, digital gain using the XICOR digital potentiometers is provided with attenuation characteristics similar to those mentioned above.

Final mixers, located on IC2, op-amps 2 and 4, are routed to the LM 1877 power amplifier. This power amplifier is capable of driving loads up to 8 ohms at one watt. Two jacks have been provided on the AOI front panel both wired in parallel. Because of this, impedance calculations should be considered before attaching any low impedance systems.

XICOR Digital Potentiometers

The XICOR digital potentiometers (data sheet shown in Appendix F) provide full digital control of an IC-based 100 step potentiometer. In the AOI system, 10K, 50K, and 100K ICs are used. Implementation of the potentiometers on this board is simple. All up, down, and increment lines are tied to a single pin and are operated using the AY-8930 control pins. Chip select, Pin 7, is individually controlled across the entire board using individual pins on the AY-8930s (see the chip control data sheet under the AY-8930 section). This implementation provides two features. The first, the lessening of control lines by ganging all up/down and increment lines together, provides a simpler and less noisy board layout. The second, individual control of chip select but not up/down and increment, provides high speed chip select and the ability to gang together the operation of several devices at once, i.e., the entire AY-8930-0 audio can be attenuated while not affecting the channels on AY-8930-1.

The Ground Plane/ Power Board

The ground plane power board provides two basic functions. The first, distribution of power and signals for the entire AOI system. The second, isolation of high frequency noise from the microprocessor unit to the sensitive analog audio portions of the AOI. Power from the external DB-9 power connector is routed directly across the board through several bypass capacitors and ferrite beads. This implementation of power filtering provides not only low frequency filtering and decoupling but also high frequency noise attenuation. Once the power is brought to the board, it is distributed to both the AY-8930 board to board bus, busses C1-C4, and connector AC1. Complete schematics, board layouts, and pin numbering for this board can be found in Appendix D.

Noise Filter Board

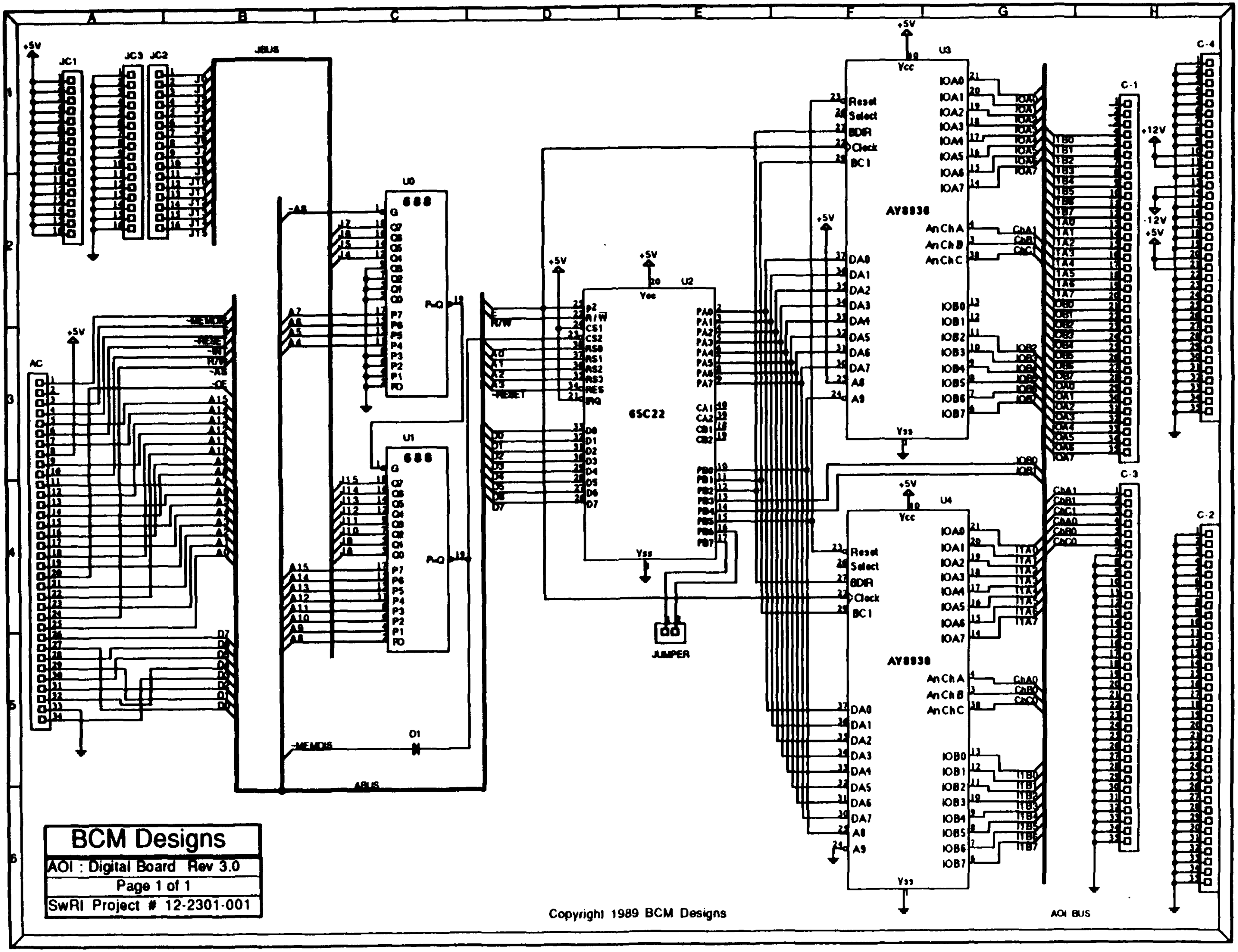
The noise filter board, a final addition to the AOI-II system, consists of a white noise source, a fixed low pass and variable high pass digitally controlled filter system. Wide band white noise is achieved using the LM389 audio op-amp. The op-amp and its associated free transistor are used in conjunction to generate white noise. The transistor shown in the data sheet, located in Appendix E, is used in an open collector mode to generate transistor noise. The resulting noise is amplified and used within this circuit. The resulting signal is processed using a variable high pass, fixed low pass, band passed filter configuration. This filter utilized a fixed low pass edge, located on IC2, and a variable high pass, Located on IC3. The filter building blocks, consisting of four free op-amps on the FLT-U2s, provide not only stable, easy to configure filter configurations, but also in the case of the high pass variability, provide a simple resistor based frequency cut-off variations. In this configuration four XICOR pots are used to set the cut-off filter frequency. The four pots on the board share the up/down and increment lines found on the main board. A single chip select line is used for all of the XICOR pots on this board. The board has been developed as a daughter board, and because of this, all signals provided through connector C0 on the mixer audio board. Note that 5V, -12V, +12V and ground are also provided through this connector. Because of the mechanical features of this board, and because of its mounting position on the audio mixer board, care must be taken when mounting or unmounting these components. See Appendix E for schematics, board layouts and other information on the FLT-U2 Universal Filter Building Block.

6255 Board

The final board in the system is the 6255 New Micros dual ACIA board. This board consisting of a Rockwell 6552 dual ACIA that provides two 8 bit serial ports memory mapped under full control of the microprocessor to the AOI-II. Only a single port on this board is used to communicate between the AOI and the FIP. For a description of the protocols used, see the module ACIACOM in the source code in Appendix H on the software. Detailed information on the 65C52 dual ACIA is provided in Appendix C.

Appendix A - 8930/6522 board

Appendix A - PCB schematics



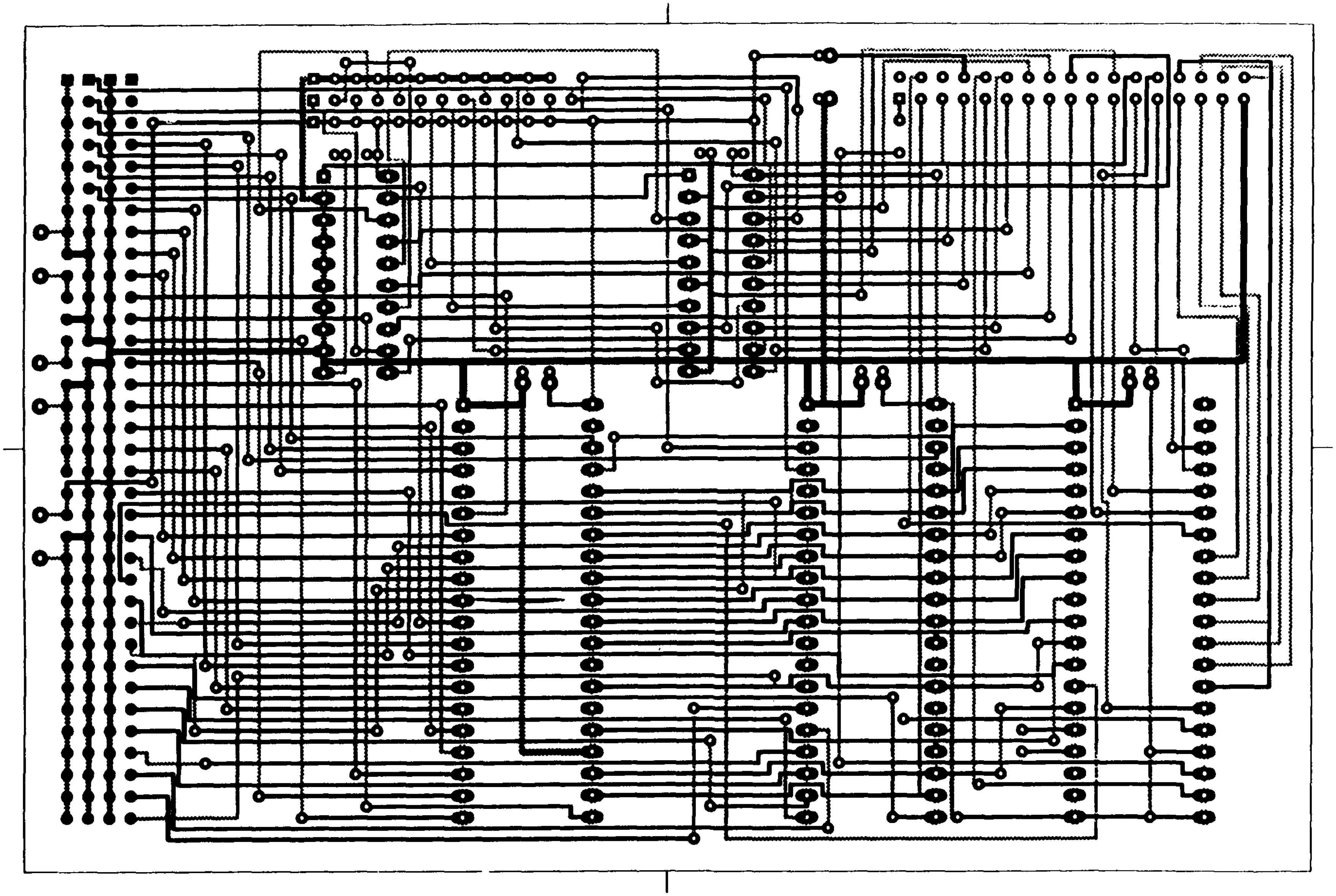
BCM Designs

AOI : Digital Board Rev 3.0

Page 1 of 1

SwRI Project # 12-2301-001

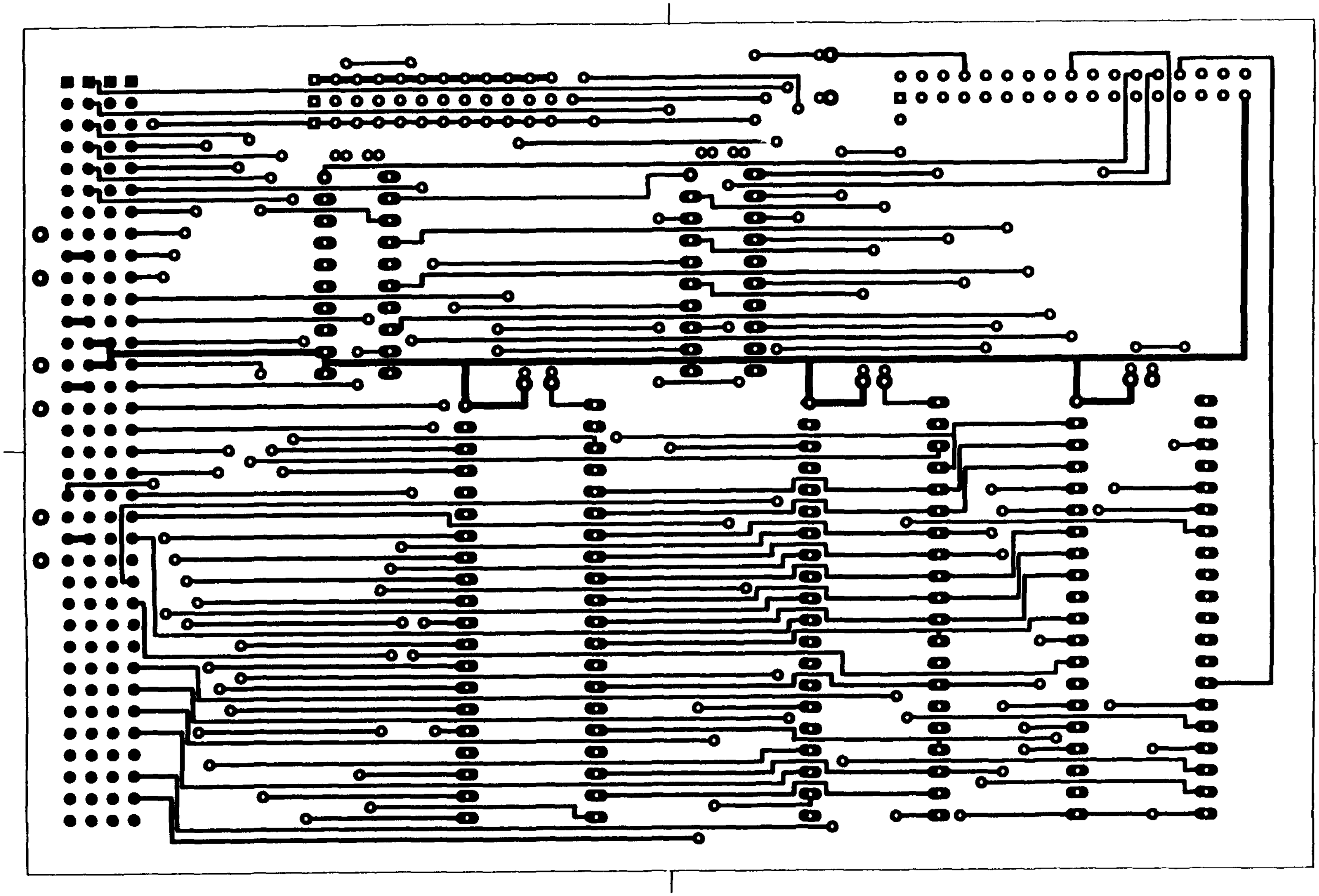
Appendix A - PCB layouts



Digital Board Final 2.0

Wed, Sep 13, 1989 11:24 AM

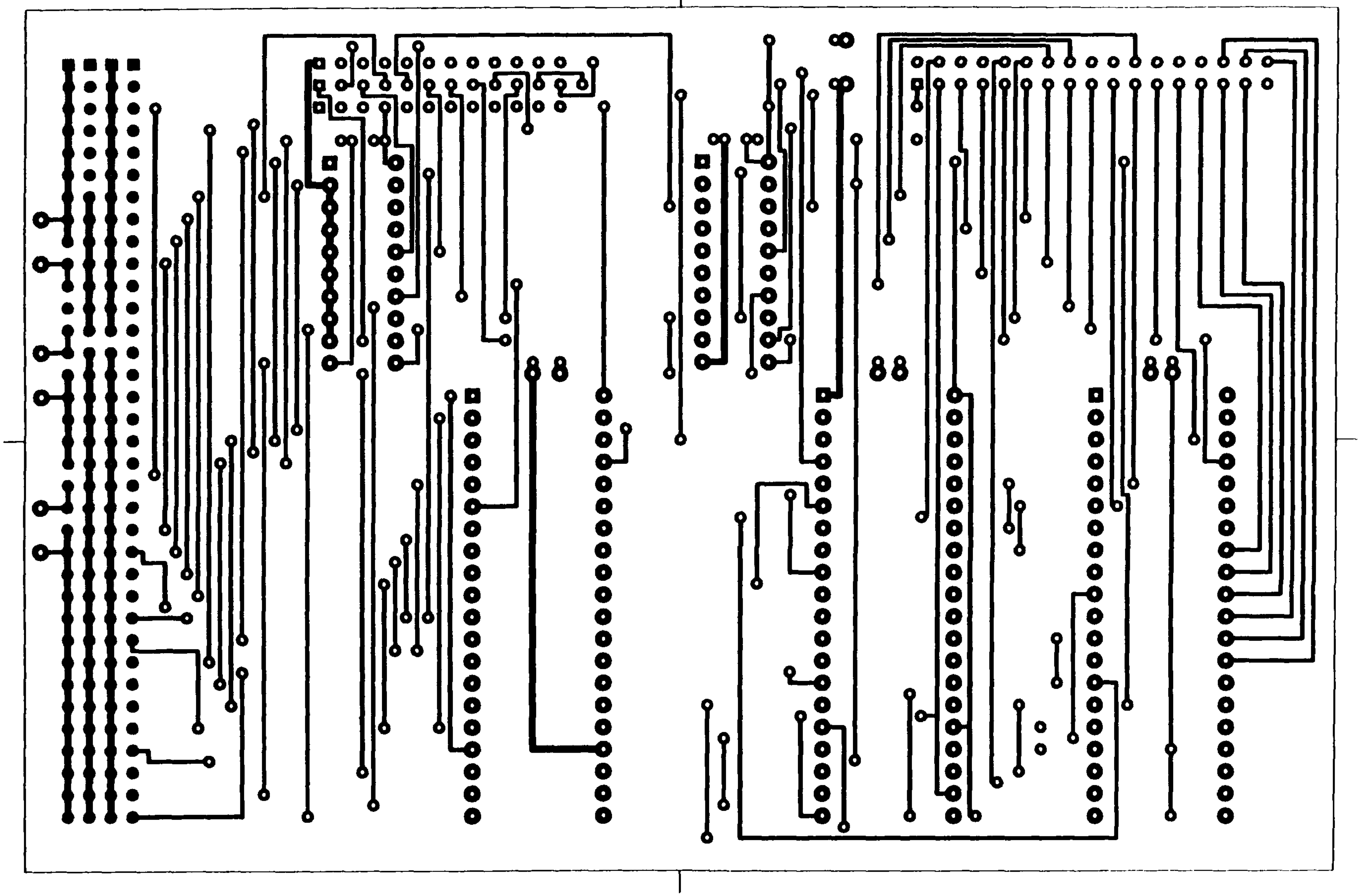
Component Xray Scale150%



Digital Board Final 2.0

Wed, Sep 13, 1989 11:24 AM

Component Side Scale150%



Digital Board Final 2.0

Wed, Sep 13, 1989 11:24 AM

Solder Side Scale150%

Appendix A - data sheets 74LS688

MOTOROLA

SN54/74LS682
SN54/74LS684
SN54/74LS688

8-BIT MAGNITUDE COMPARATORS

LOW POWER SCHOTTKY

DESCRIPTION — The SN54LS/74LS682, 684, 688 are 8-bit magnitude comparators. These device types are designed to perform comparisons between two eight-bit binary or BCD words. All device types provide $\overline{P=Q}$ outputs and the LS682 and LS684 have $\overline{P>Q}$ outputs also.

The LS682, LS684 and LS688 are totem pole devices. The LS682 has a 20 kΩ pullup resistor on the Q inputs for analog or switch data.

TYPE	$\overline{P=Q}$	$\overline{P>Q}$	OUTPUT ENABLE	OUTPUT CONFIGURATION	PULLUP
LS682	yes	yes	no	totem-pole	yes
LS683	yes	yes	no	open-collector	yes
LS684	yes	yes	no	totem-pole	no
LS685	yes	yes	no	open-collector	no
LS686	yes	yes	yes	totem-pole	no
LS687	yes	yes	yes	open-collector	no
LS688	yes	no	yes	totem-pole	no
LS689	yes	no	yes	open-collector	no

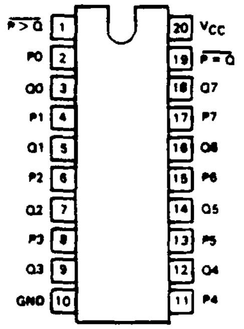
FUNCTION TABLE

INPUTS			OUTPUTS	
DATA	ENABLES			
P, Q	$\overline{G}$, $\overline{G1}$	$\overline{G2}$	$\overline{P=Q}$	$\overline{P>Q}$
P = Q	L	L	L	H
P > Q	L	L	H	L
P < Q	L	L	H	H
X	H	H	H	H

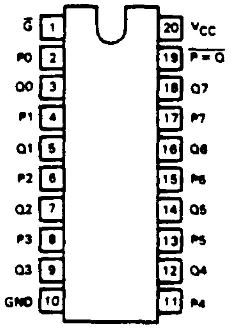
H = high level, L = low level, X = irrelevant

CONNECTION DIAGRAMS
(TOP VIEW)

SN54LS/74LS682/684



SN54LS/74LS688



J Suffix — Case 732-03 (Ceramic)
N Suffix — Case 738-03 (Plastic)

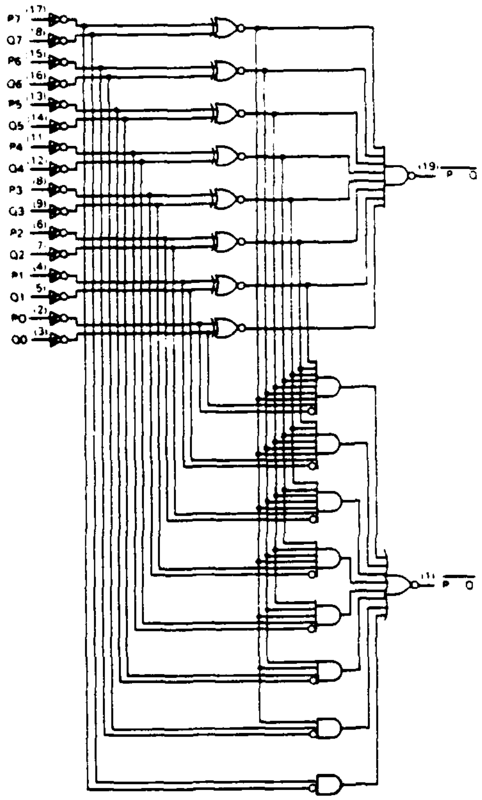
GUARANTEED OPERATING RANGES

SYMBOL	PARAMETER		MIN	TYP	MAX	UNIT
V_{CC}	Supply Voltage	54 74	4.5 4.75	5.0 5.0	5.5 5.25	V
T_A	Operating Ambient Temperature Range	54 74	−55 0	25 25	125 70	°C
I_{OH}	Output Current — High	54, 74			−0.4	mA
I_{OL}	Output Current — Low	54 74			12 24	mA

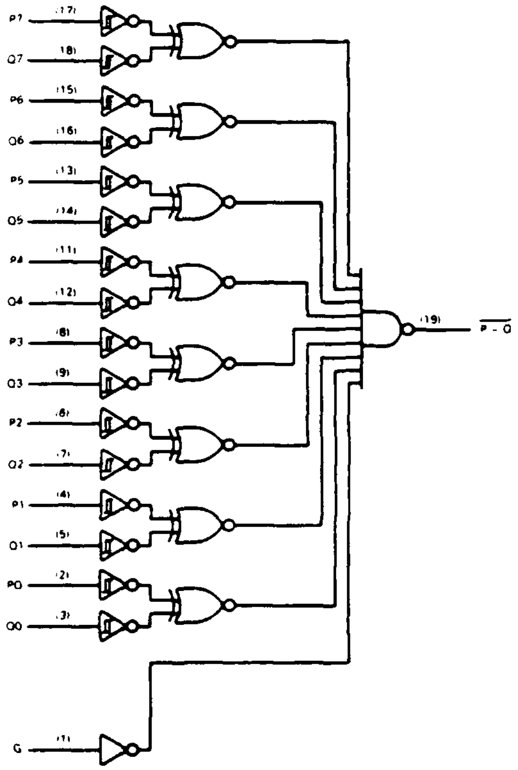
DC CHARACTERISTICS OVER OPERATING TEMPERATURE RANGE *(unless otherwise specified)*

SYMBOL	PARAMETER		LIMITS			UNITS	TEST CONDITIONS	
			MIN	TYP	MAX			
V_{IH}	Input HIGH Voltage		2.0			V	Guaranteed Input HIGH Voltage for All Inputs	
V_{IL}	Input LOW Voltage	54			0.7	V	Guaranteed Input LOW Voltage for All Inputs	
		74			0.8			
V_{IK}	Input Clamp Diode Voltage			−0.65	−1.5	V	V_{CC} = MIN, I_{IN} = −18 mA	
V_{OH}	Output HIGH Voltage	54	2.5	3.5		V	V_{CC} = MIN, I_{OH} = MAX, V_{IN} = V_{IH} or V_{IL} per Truth Table	
		74	2.7	3.5		V		
V_{OL}	Output LOW Voltage	54, 74		0.25	0.4	V	I_{OL} = 12 mA	V_{CC} = V_{CC} MIN, V_{IN} = V_{IL} or V_{IH} per Truth Table
		74		0.35	0.5	V	I_{OL} = 24 mA	
I_{IH}	Input HIGH Current				20	µA	V_{CC} = MAX, V_{IN} = 2.7 V	
		LS682-Q Inputs			0.1	mA	V_{CC} = MAX, V_{IN} = 5.5 V	
		Others			0.1	mA	V_{CC} = MAX, V_{IN} = 7.0 V	
I_{IL}	Input LOW Current	LS682-Q Inputs			−0.4	mA	V_{CC} = MAX, V_{IN} = 0.4 V	
		Others			−0.2	mA		
I_{OS}	Short Circuit Current		−30		−130	mA	V_{CC} = MAX	
I_{CC}	Power Supply Current	LS682			70	mA	V_{CC} = MAX	
		LS684			65	mA		
		LS688			65	mA		

LOGIC DIAGRAMS



SN54LS/74LS682 thru LS684



SN54LS/74LS688

AC CHARACTERISTICS: T_A = 25°C

SN54LS/74LS682

SYMBOL	PARAMETER	LIMITS			UNITS	TEST CONDITIONS
		MIN	TYP	MAX		
t_{PLH} t_{PHL}	Propagation Delay, P to $\overline{P=Q}$		13 15	25 25	ns	V_{CC} = 5.0 V C_L = 45 pF R_L = 667 Ω
t_{PLH} t_{PHL}	Propagation Delay, Q to $\overline{P=Q}$		14 15	25 25	ns	
t_{PLH} t_{PHL}	Propagation Delay, P to $\overline{P>Q}$		20 15	30 30	ns	
t_{PLH} t_{PHL}	Propagation Delay, Q to $\overline{P>Q}$		21 19	30 30	ns	

SN54LS/74LS684

SYMBOL	PARAMETER	LIMITS			UNITS	TEST CONDITIONS
		MIN	TYP	MAX		
t_{PLH} t_{PHL}	Propagation Delay, P to $\overline{P=Q}$		15 17	25 25	ns	V_{CC} = 5.0 V C_L = 45 pF R_L = 667 Ω
t_{PLH} t_{PHL}	Propagation Delay, Q to $\overline{P=Q}$		16 15	25 25	ns	
t_{PLH} t_{PHL}	Propagation Delay, P to $\overline{P>Q}$		22 17	30 30	ns	
t_{PLH} t_{PHL}	Propagation Delay, Q to $\overline{P>Q}$		24 20	30 30	ns	

SN54LS/74LS688

SYMBOL	PARAMETER	LIMITS			UNITS	TEST CONDITIONS
		MIN	TYP	MAX		
t_{PLH} t_{PHL}	Propagation Delay, P to $\overline{P=Q}$		12 17	18 23	ns	V_{CC} = 5.0 V C_L = 45 pF R_L = 667 Ω
t_{PLH} t_{PHL}	Propagation Delay, Q to $\overline{P=Q}$		12 17	18 23	ns	
t_{PLH} t_{PHL}	Propagation Delay, $\overline{G}$, $\overline{G1}$ to $\overline{P=Q}$		12 13	18 20	ns	

Appendix A - AY-8930

™ Microchip

AY8930

Enhanced Programmable Sound Generator

FEATURES

- Two Modes Available On-Chip
  - — AY8930 Expanded Mode
  - — AY-3-8910A-Compatible Mode
- Improved Frequency Range
- Three Independently Programmable Analog Output Channels with Separate Frequency, Duty Cycle and Envelope Controls for Each Channel
- 5 Bits of Logarithmic Digital-to-Analog Conversion Per Channel
- Bus Interface Independent of Clock Frequency
- Input Clock Frequency: 2 or 4MHz
- Two 8-Bit General Purpose I/O Ports

DESCRIPTION

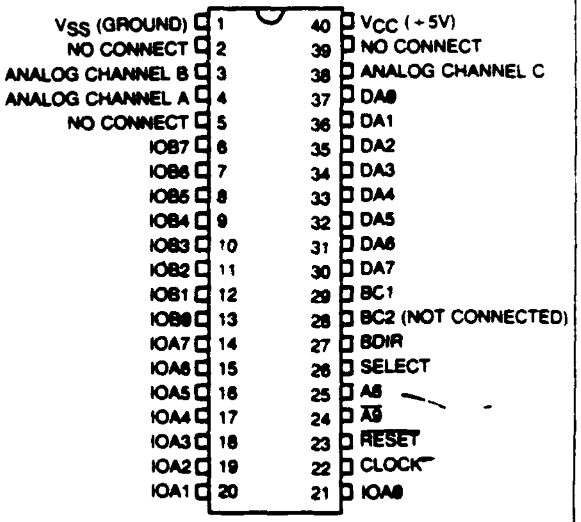
The AY8930 Enhanced Programmable Sound Generator (EPSG) is an LSI circuit that can produce a wide variety of complex sounds under software control. The AY8930 is manufactured in the Microchip Technology Inc. n-channel silicon gate process. The AY8930 is an enhanced version of the company's industry standard AY-3-8910A sound generator. Enhanced features include improved frequency range and noise synthesis and independent control of each channel's envelope and duty cycle.

The PSG is easily interfaced to any bus-oriented system. Its flexibility makes it useful in applications such as music synthesis, sound effects generation, audible alarms, tone signalling, and home computer usage. In order to generate sound effects while allowing the processor to perform other tasks, the PSG can continue to produce sound after the initial commands have been given by the control processor. The fact that realistic sound production often involves more than one effect is satisfied by the three independently controllable analog sound output channels available in the device. These analog sound output channels can each provide five bits of logarithmic digital-to-analog conversion, greatly enhancing the dynamic range of the sounds produced.

PIN CONFIGURATION

AY8930
40 LEAD DUAL IN LINE

Top View



All circuit control signals are digital in nature and can be provided directly by a microprocessor/microcomputer. Therefore, one PSG can produce the full range of required sounds with no change in external circuitry. Since the frequency response of the PSG ranges from sub-audible at its lowest frequency to post-audible at its highest frequency, there are few sounds which are beyond reproduction with only the simplest electrical connections.

DEVICE ARCHITECTURE

The AY8930 is a register oriented PSG. Communication between the microprocessor and the PSG is based on the concept of memory mapped I/O. Control commands are issued to the PSG by writing to these memory mapped registers. Each of the registers within the PSG is readable so that the microprocessor can determine, as necessary, present states or stored data values.

REGISTER ARRAY

The principal element of the AY8930 is an array of 27 control registers arranged in one bank of 16 and one bank of 11 registers. These registers occupy 16 address locations of the 1,024-word memory space in which the PSG resides.

The configuration of this register array is shown on the following pages. Note the two modes of operation: 8910A-compatibility mode and 8930 expanded mode.

The registers are addressed via the combination of the bidirectional data bus (DA0–DA7) and address input pins A8 and $\overline{\text{A9}}$.

$\overline{\text{A9}}$	A8	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
0	1	0	0	0	0	X	X	X	X

DA0–DA3 These four low-order address bits are used to select one of the internal registers within a bank.

DA4–DA7, A8, $\overline{\text{A9}}$ These six high-order address bits function as chip selects and are used to position the register bank(s) within the 1,024-word memory space. In the deselected state, the data bus is in the high impedance state.

The address enable code for bits DA4–DA7 is all zeros.

Inputs A8 and $\overline{\text{A9}}$ are enabled by a high on A8 and a low on $\overline{\text{A9}}$; all other input level combinations result in a deselected condition. Pins A8 and $\overline{\text{A9}}$ have an on-chip pull-up and pull-down resistor, respectively, and will assume the correct logic level if left unconnected.

Address bits DA7–DA0 are latched internally. This internally latched address is updated and modified on every "latch address" signal presented to the PSG via the BDIR and BC1 control lines.

The AY8930 initializes in the AY-3-8910A-compatibility mode. To utilize the expanded features of the AY8930, an access code must be input to register R15 upon program initialization.

Entering a "101" code in bits B7–B5 of register R15 selects the 8930 expanded mode. In the 8930 Expanded mode, bit B4 = 0 (R15) selects BANK A and B4 = 1 selects BANK B. All other bit selections are defined as 8910A-compatibility mode. Registers R15A and R15B are mapped into the same physical register.

Switching modes causes loss of all register data from the previous mode. All registers will be initialized except for the Mode Select code of R15.

Shown on the next page is the register configuration for the AY8930. Note that Bank A of the expanded mode is virtually identical to the single register array of the 8910A-compatibility mode.

AY8930 REGISTER ARRAY: AY-3-8910A-COMPATIBILITY MODE

REGISTER	BIT	B7	B6	B5	B4	B3	B2	B1	B0
R0	Channel A	8-Bit Fine Tune							
R1	Tone Period	▨	▨	▨	▨	4-Bit Coarse Tune			
R2	Channel B	8-Bit Fine Tune							
R3	Tone Period	▨	▨	▨	▨	4-Bit Coarse Tune			
R4	Channel C	8-Bit Fine Tune							
R5	Tone Period	▨	▨	▨	▨	4-Bit Coarse Tune			
R6	Noise Period	▨	▨	▨	5-Bit Period Control				
		IN/OUT		NOISE			TONE		
R7	$\overline{\text{Enable}}$	IOB	IOA	C	B	A	C	B	A
R10	Channel A Amplitude	▨	▨	▨	M	L3	L2	L1	L0
R11	Channel B Amplitude	▨	▨	▨	M	L3	L2	L1	L0
R12	Channel C Amplitude	▨	▨	▨	M	L3	L2	L1	L0
R13		8-Bit Fine Tune							
R14	Envelope Period	8-Bit Coarse Tune							
R15	Envelope Shape/Cycle	MODE SELECT				CONT.	ATT.	ALT.	HOLD
R16	I/O Port A	8-Bit Parallel I/O on Port A							
R17	I/O Port B	8-Bit Parallel I/O on Port B							

AY8930 REGISTER ARRAY: EXPANDED CAPABILITY MODE — BANK A

REGISTER	BIT	B7	B6	B5	B4	B3	B2	B1	B0
R0A	Channel A	8-Bit Fine Tune							
R1A	Tone Period	8-Bit Coarse Tune							
R2A	Channel B	8-Bit Fine Tune							
R3A	Tone Period	8-Bit Coarse Tune							
R4A	Channel C	8-Bit Fine Tune							
R5A	Tone Period	8-Bit Coarse Tune							
R6A	Noise Period	8-Bit Noise Period							
		IN/OUT		NOISE			TONE		
R7A	$\overline{\text{Enable}}$	IOB	IOA	C	B	A	C	B	A
R10A	Channel A Amplitude	▨		M	L4	L3	L2	L1	L0
R11A	Channel B Amplitude	▨		M	L4	L3	L2	L1	L0
R12A	Channel C Amplitude	▨		M	L4	L3	L2	L1	L0
R13A	Channel A	8-Bit Fine Tune							
R14A	Envelope Period	8-Bit Coarse Tune							
R15A	Bank A/B: Envelope A	1	0	1	0	CONT.	ATT.	ALT.	HOLD
R16A	I/O Port A	8-Bit Parallel I/O on Port A							
R17A	I/O Port B	8-Bit Parallel I/O on Port B							

AY8930 REGISTER ARRAY: EXPANDED CAPABILITY MODE — BANK B

REGISTER	BIT	B7	B6	B5	B4	B3	B2	B1	B0
R0B	Channel B	8-Bit Fine Tune							
R1B	Envelope Period	8-Bit Coarse Tune							
R2B	Channel C	8-Bit Fine Tune							
R3B	Envelope Period	8-Bit Coarse Tune							
R4B	Envelope Shape/Cycle B	▨				CONT.	ATT.	ALT.	HOLD
R5B	Envelope Shape/Cycle C	▨				CONT.	ATT.	ALT.	HOLD
R6B	Channel A Duty Cycle	▨				4 Bit			
R7B	Channel B Duty Cycle	▨				4 Bit			
R10B	Channel C Duty Cycle	▨				4 Bit			
R11B	Noise"And" Mask	8 Bit							
R12B	Noise "OR" Mask	8 Bit							
R13B*	▨	▨							
R14B*	▨	▨							
R15B	Bank A/B: Envelope A	1	0	1	1	CONT.	ATT.	ALT.	HOLD
R16B*	▨	▨							
R17B*	TEST	NOT TO BE ACCESSED BY THE USER							

*Not accessible in AY8930 mode.

NOTE: All unused bits will be read back as "0."

SOUND GENERATING BLOCKS

The basic blocks in the PSG that produce the programmed sounds include:

Tone Generators	— Produce the basic pulse tone frequencies for each channel (A, B, C).
Noise Generator	— Produces a frequency modulated pseudorandom noise output.
Mixers	— Combine the outputs of the tone generators and the noise generator, one for each channel (A, B, C).
Amplitude Control	— Provides the D/A converters with either a fixed or variable amplitude pattern. The fixed amplitude is under direct CPU control; the variable amplitude is accomplished by using the output of the envelope generators, one for each channel (A, B, C).
Envelope Generators	— Produce an envelope pattern that can be used to amplitude modulate the output of the mixer, one for each channel (A, B, C).
D/A Converters	— The three D/A converters each produce up to a 32-level output signal as determined by the amplitude control.

Since all functions of the PSG are controlled by a host processor via a series of register loads, a detailed description of the PSG operation can best be accomplished by relating each PSG function to its corresponding register. The function of creating or programming a specific sound or sound effect logically follows the control sequence listed:

OPERATION	REGISTERS*	FUNCTION
Tone Generator Control	R0–R5A	Program tone periods.
Duty Cycle Control	R6–R10B	Select duty cycle.
Noise Generator Control	R6A, R11–12B	Program noise period.
Mixer Control	R7A	Enable tone/noise on selected channels.
Amplitude Control	R10–R12A	Select "fixed" or "variable" amplitudes.
Envelope Generator Control	R13–R15A, R0–R5B	Program envelope period and envelope pattern.
D/A Converters	—	Produces a 32-bit output signal.

*All registers referenced are for the AY8930 Expanded Mode, except as noted.

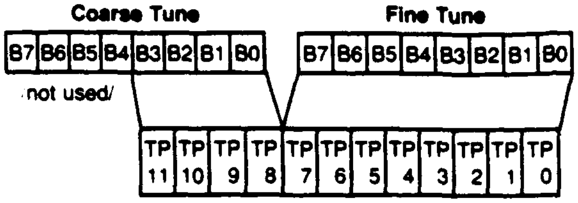
TONE GENERATOR CONTROL

Each analog output channel has associated with it two registers which specify the tone period for that channel, the coarse tune and the fine tune registers. The tone period for each channel is obtained by combining the coarse and fine tune registers as shown.

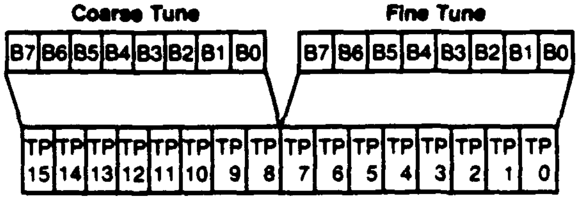
Note that the value programmed in the combined coarse and fine tune registers is a *period* value — the higher the value in the registers, the lower the resultant frequency.

COARSE TUNE REGISTER	CHANNEL	FINE TUNE REGISTER
R1A	A	R0A
R3A	B	R2A
R5A	C	R4A

12-BIT TONE PERIOD (TP) VALUE: AY-3-8910A-COMPATIBILITY MODE



16-BIT TONE PERIOD (TP) VALUE: AY8930 EXPANDED MODE



NOTE:

If the coarse and fine tune registers are both set to 000_8, the resulting period will be minimum, i.e. the generated tone period will be as if the coarse tune register was set to 000_8 and the fine tune register was set to 001_8. The counter will count the period value down to zero. When zero is reached, the period value will be reloaded into the counter.

DUTY CYCLE CONTROL

The duty cycle of each pulse generated by the three tone generators is controlled by an associated 4-bit duty cycle register (R6B, R7B, and R10B).

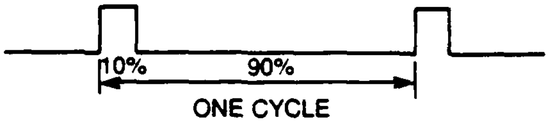
The following duty cycles are selectable:

% DUTY CYCLE	DUTY CYCLE	REGISTER VALUE
3.125%	0	0000
6.25 %	1	0001
12.50 %	2	0010
25.00 %	3	0011
50.00 %	4	0100
75.00 %	5	0101
87.50 %	6	0110
93.75 %	7	0111
96.875%	8*	1111

*NOTE: Any value greater than 8_{10} decodes as an 8_{10}.

NOTE:

The percent duty cycles refers to the high (logic high) portion of the duty cycle. The low portion is then 100 percent duty cycle. A 10% duty cycle is then 10% up and 90% down, as shown below.



In 8910A-compatibility mode, the duty cycle is fixed at 50%. The capability for a variable duty cycle exists only in the expanded AY8930 mode.

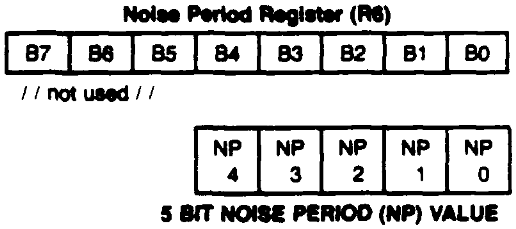
In order to change a duty cycle, the appropriate duty cycle register must be updated. The new duty cycle will then remain constant at this value until the duty cycle register is modified. The new duty cycle value will take effect immediately. This may result in one period with a "random" duty cycle at the time the register is updated.

NOISE GENERATOR CONTROL

AY-3-8910A-Compatibility Mode:

Noise is generated by a 17-bit polynomial shift register. The period of the clock to this shift register is specified by the 8-bit binary value NP.

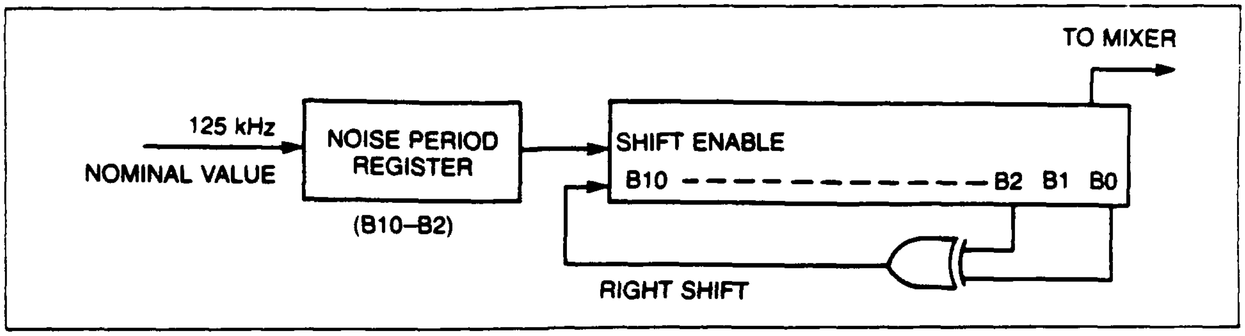
The noise period value is derived from the lower five bits (B4–B0) of the noise period register (R6) as shown.



NOTE:

As with the tone period, the lowest period value is 00001 (divide by 1), an entry of 00000 will have the same value as 00001; the highest period value is 11111 (divide by 31_{10}).

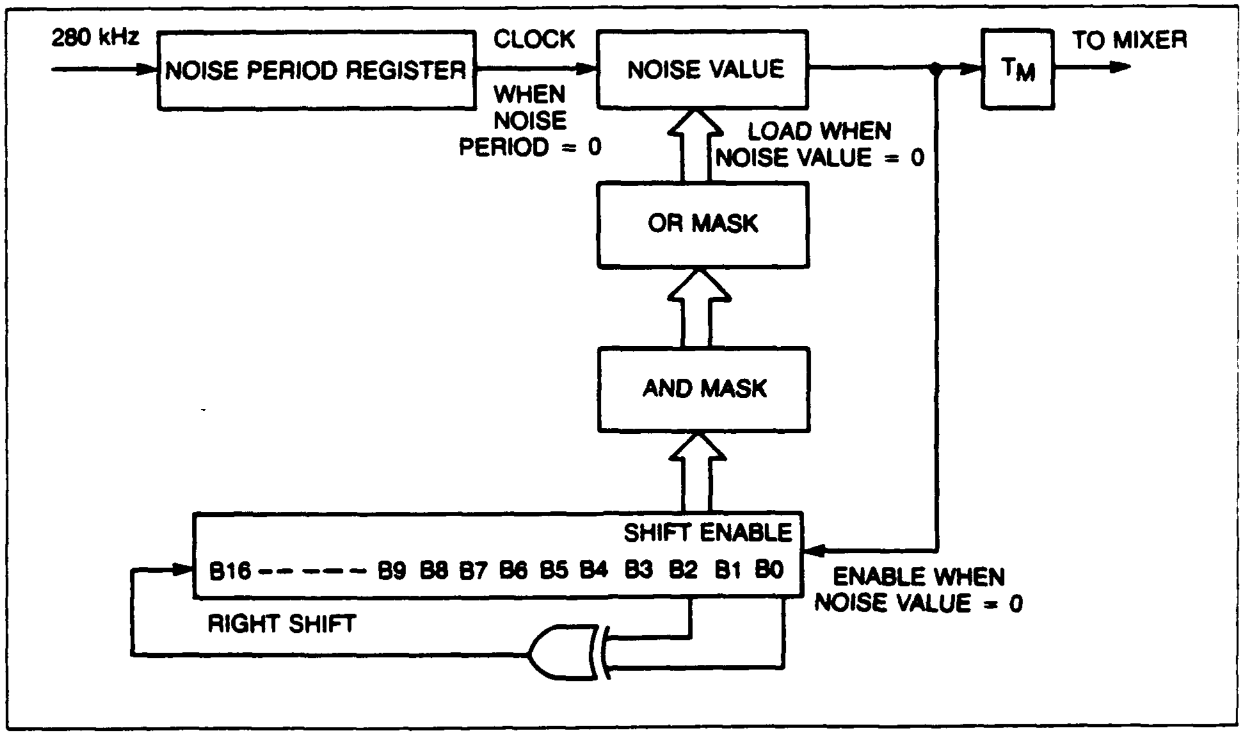
AY-3-8910A-COMPATIBILITY MODE NOISE BLOCK DIAGRAM (Figure 1)



AY8930 Expanded Mode:

In the AY8930 expanded mode, noise is generated using a 17-bit polynomial shift register, an "AND" mask, an "OR" mask, and an 8-bit noise period value. The least significant byte of the polynomial shift register is logically AND'ed with the "AND" mask specified in Register 11B, then logically OR'ed with the "OR" mask specified in Register 12B. The result is stored in a temporary register which is clocked each time the counter associated with the 8-bit noise period register (R6A) reaches zero. When the noise value reaches zero, a new value is fetched from the polynomial shift register and the process is repeated. The noise output is toggled each time the noise value reaches zero.

AY8930 EXPANDED MODE NOISE BLOCK DIAGRAM (Figure 2)



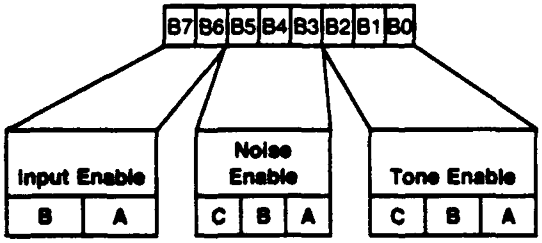
MIXER CONTROL — I/O ENABLE

Register 7A is a multi-function $\overline{\text{Enable}}$ register which controls the three Noise/Tone Mixers and the two general purpose I/O Ports. The mixers as previously described, combine the noise and tone frequencies for each of the three channels. The determination of combining neither/either/both noise and tone frequencies on each channel is made by the state of bits B5–B0 of R7A.

The direction (input or output) of the two general purpose I/O Ports (IOA and IOB) is determined by the state of bits B7 and B6 of R7A.

These functions are illustrated in the following:

Mixer Control — I/O Enable Register (R7A)



Noise Enable Truth Table

R7A Bits B5	B4	B3	Noise Enabled On Channel		
0	0	0	C	B	A
0	0	1	C	B	—
0	1	0	C	—	A
0	1	1	C	—	—
1	0	0	—	B	A
1	0	1	—	B	—
1	1	0	—	—	A
1	1	1	—	—	—

Tone Enable Truth Table

R7A Bits B2	B1	B0	Tone Enabled On Channel		
0	0	0	C	B	A
0	0	1	C	B	—
0	1	0	C	—	A
0	1	1	C	—	—
1	0	0	—	B	A
1	0	1	—	B	—
1	1	0	—	—	A
1	1	1	—	—	—

The direction of the I/O Port(s) is determined as follows:

I/O Port Truth Table

R7A Bits		I/O Direction	
B7	B6	IOB	IOA
0	0	In	In
0	1	In	Out
1	0	Out	In
1	1	Out	Out

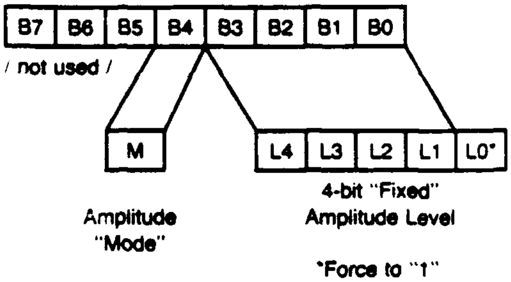
Note: The Mixer — I/O Control function is identical in both modes of operation.

AMPLITUDE CONTROL

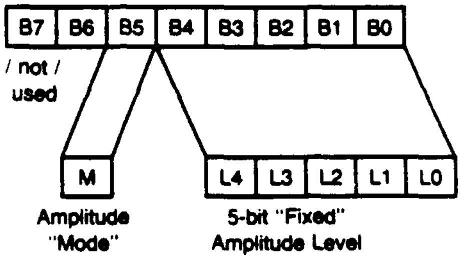
The amplitudes of the signals generated by each of the three D/A converters (one each for channels A, B, and C) are determined by the contents of the amplitude control registers as illustrated in the following:

Amplitude Control Register #	Channel
R10A	A
R11A	B
R12A	C

AY-3-8910A-Compatibility Mode:



AY8930 Expanded Mode:



The amplitude "mode" (bit M) selects either fixed level amplitude (M = 0) or variable level amplitude (M = 1). It follows that bits L4–L0, defining the value of a "fixed" level amplitude, are only active when M = 0. The amplitude is only "fixed" in the sense that the amplitude level is under the direct control of the system processor via an address latch/write data sequence.

When "variable amplitude" is selected (M = 1), the amplitude of each channel is determined by the envelope pattern as defined by the envelope generators 5-bit output (E4–E0). The amplitude "mode" bit (bit M) can also be thought of as an envelope enable bit, i.e. when M = 1, the envelope is enabled.

AMPLITUDE CONTROL (continued)

The following is a chart describing all combinations of the 6-bit Amplitude Control.

M	L4	L3	L2	L1	L0	
0	0	0	0	0	0*	The amplitude is fixed at 1 of 31 levels as determined by L4, L3, L2, L1, L0.
0	0	0	0	0	1*	
•	•	•	•	•	•	
•	•	•	•	•	•	
0	1	1	1	1	1	
1	X	X	X	X	X	The amplitude is variable at 31 levels as determined by the output of the Envelope Generator.

(X = Don't Care)

*These codes are used to turn a channel "off."

NOTE:

In the AY-3-8910A-compatibility mode, the externally driven "fixed" amplitude is limited to a total of 16 possible levels determined by amplitude bits L4–L1.

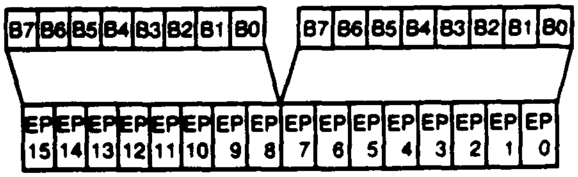
ENVELOPE GENERATOR CONTROL

Envelope Period Control

The period of the sound envelope, in the 8910A-compatibility mode, is controlled by two 8-bit registers, R13 and R14 (the envelope fine and coarse tune, respectively). In the 8930 expanded mode, each analog output channel has its own independent sound envelope. Changes to the envelope period counter will occur at envelope period boundary or when envelope shape/cycle register is loaded.

COARSE TUNE REGISTER	CHANNEL	FINE TUNE REGISTER
R14A	A	R13B
R 1B	B	R 0B
R 3B	C	R 2B

16-BIT ENVELOPE PERIOD TO ENVELOPE GENERATOR



Note that the value programmed in the combined coarse and fine tune registers is a period value — the higher the value in the registers, the lower the resultant frequency.

Note also, that as the tone period, the lowest period values is 000001_8 (divided by 1); the highest period value is 177777_8 (divided by $65{,}535_{10}$).

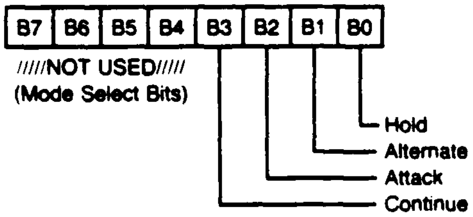
Envelope Shape/Cycle Control

The envelope generator further counts down the envelope frequency by 32, producing a 32-state per cycle envelope pattern defined by its 5-bit counter output, E4, E3, E2, E1, and E0. The particular shape and cycle pattern of any desired envelope is accomplished by controlling the count pattern (count up/count down) of the 5-bit counter and by defining a single cycle or repeat-cycle pattern.

Loading of the envelope shape/cycle control register will reset the associated counter to the appropriate initial state and reset the envelope period counter for that channel.

The envelope shape/cycle control is contained in the lower 4 bits (B3–B0) of the respective envelope control registers. Each of these 4 bits controls a function in the envelope generator, as illustrated in the following:

Envelope Shape/Cycle Control Register



The definition of each function is as follows:

Hold — when set to logic "1", limits the envelope to one cycle, holding the last count of the envelope counter (E3—E0 = 0000 or 1111, depending on whether the envelope counter was in a count-down or count-up mode, respectively).

Alternate — when set to logic "1", the envelope counter reverses count direction (up-down) after each cycle.

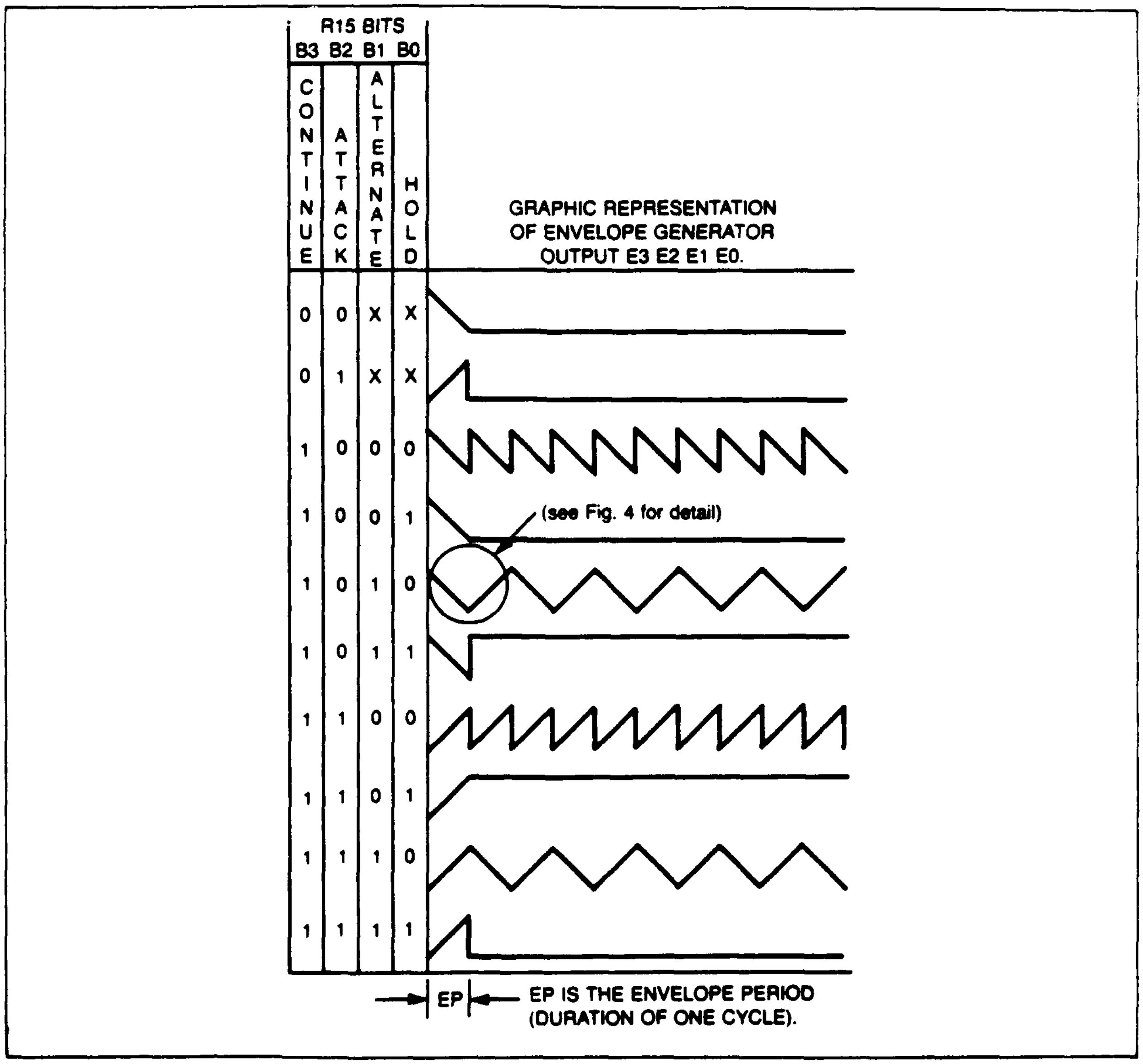
NOTE: When both the hold bit and the alternate bit are ones, the envelope counter is reset to its initial count before holding.

Attack — when set to logic "1", the envelope counter will count up (attack) from E3 E2 E1 E0 = 0000 to E3 E2 E1 E0 = 1111; when set to logic "0", the envelope counter will count down (decay) from 1111 to 0000.

Continue — when set to logic "1", the cycle pattern will be as defined by the hold bit; when set to logic "0", the envelope generator will reset to 0000 after one cycle and hold at that count.

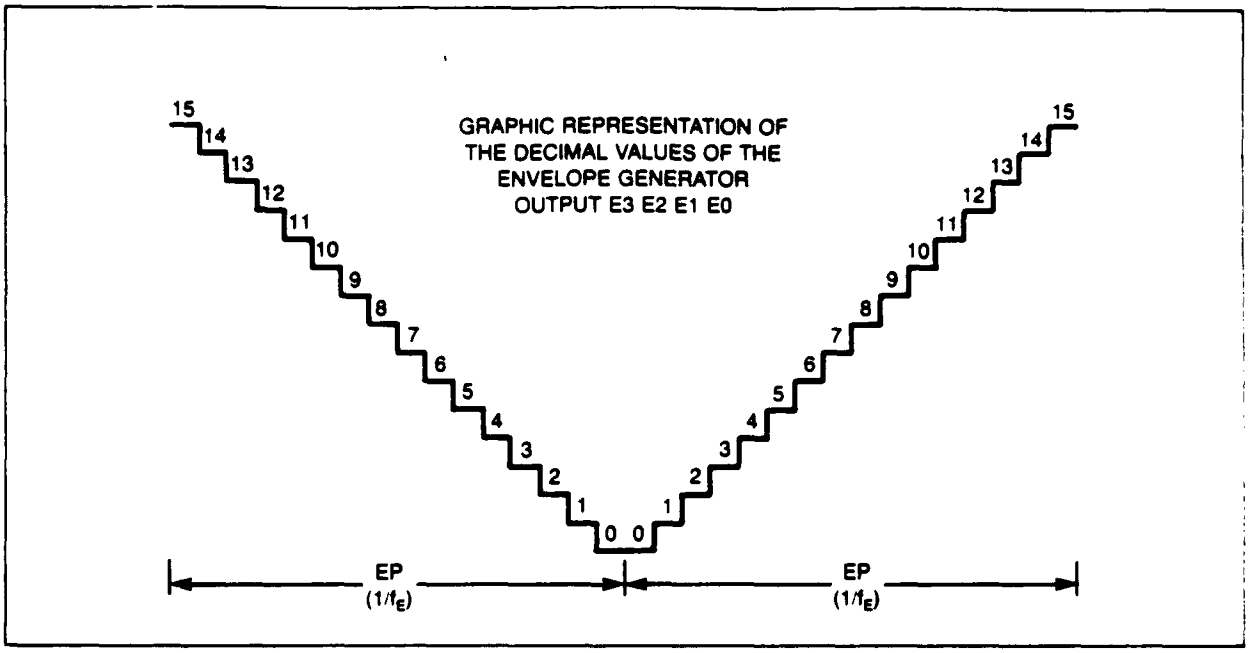
Further description of the above functions could be accomplished by numerous charts of the binary count sequence of E3 E2 E1 E0 for each combination of hold, alternate, attack and continue. However, since these outputs are used (when selected by the amplitude control registers) to amplitude modulate the output of the mixers, a better understanding of their effect can be accomplished via a graphic representation of their value for each condition selected, as illustrated in Figs. 3 and 4.

ENVELOPE SHAPE/CYCLE CONTROL (Figure 3)



R15 BITS			
B3	B2	B1	B0
CONTINUE	ATTACK	ALTERNATE	HOLD
0	0	X	X
0	1	X	X
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

DETAIL OF TWO CYCLES OF Fig. 3 — AY-3-8910A MODE (Figure 4) (ref. waveform "1010" in Fig. 3)

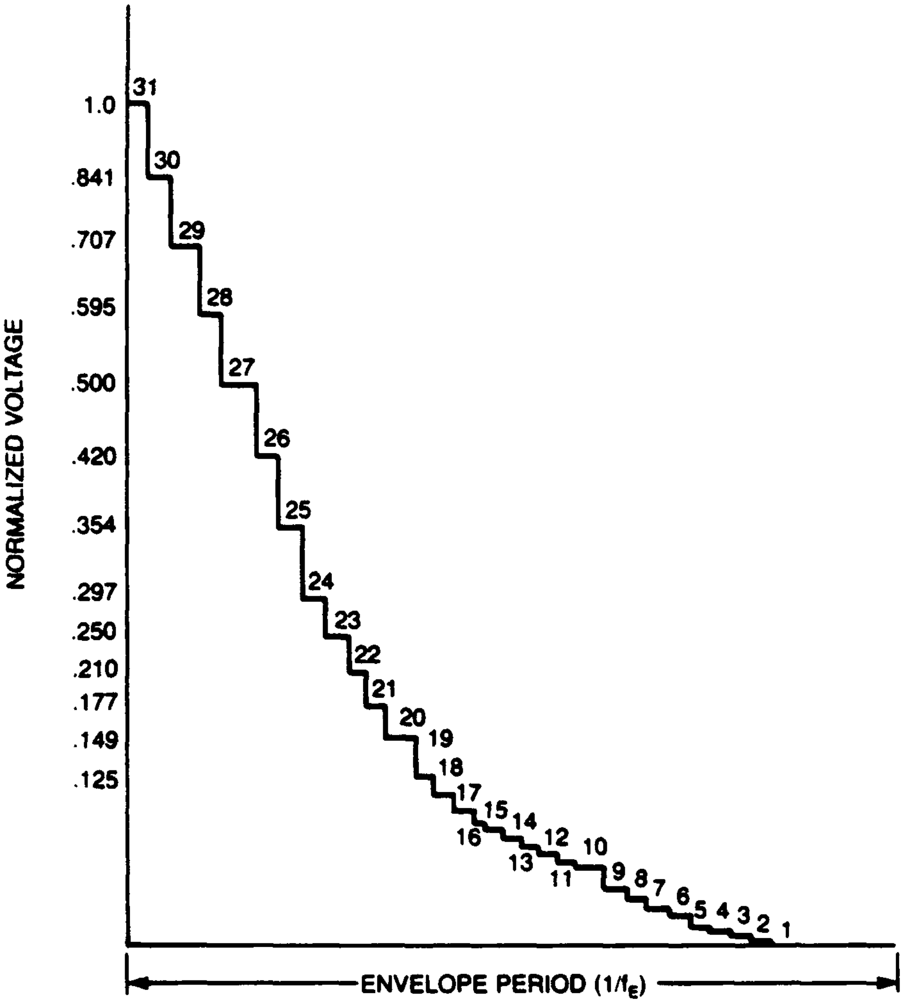

GRAPHIC REPRESENTATION OF THE DECIMAL VALUES OF THE ENVELOPE GENERATOR OUTPUT E3 E2 E1 E0
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
EP
$(1/f_E)$
EP
$(1/f_E)$

DIGITAL TO ANALOG CONVERTER

The Digital to Analog conversion is performed in logarithmic steps with a normalized voltage range of 0V to 1.0V. The specified amplitude of each converter is controlled by a 5-bit word from either the amplitude control register* or the envelope generator. The signal of the output is the Noise/Tone specified for that channel.

*Except in the 8910A-compatibility mode, which only allows for 4 bits of external amplitude control.

D/A CONVERTER OUTPUT (Figure 5)



This figure illustrates the D/A converter output which would result if noise and tones were disabled and an envelope controlled variable amplitude were selected.

NOTE: The RESET condition is zero current.

I/O PORT DATA STORE

Registers R16A and R17A function as intermediate data storage registers between the PSG/CPU data bus (DA0–DA7) and the two I/O ports (IOA0–IOA7, IOB0–IOB7). Both I/O ports are available on the AY8930.

Using registers R16A and R17A for the transfer of I/O data has no effect at all on sound generation.

To output data from the CPU Bus to a peripheral device connected to I/O port A:

1. Address the enable register (R7A).
2. Set the port A direction bit to output (write "1" to bit B6 of R7A).
3. Address the I/O port A register (R16A).
4. Write data to I/O port A register. The data will pass through the PSG I/O port A register to the I/O port bus.

To input data from I/O port A to the CPU bus:

1. Address the enable register (R7A).
2. Set the port A direction bit to input (write a "0" to bit B6 of R7A).
3. Address the I/O port A register (R16A). The contents of the port register will follow the signals applied to the I/O port.
4. Read data from I/O port A register. The data will be transferred from the PSG I/O port A register to the CPU bus as in a normal read operation.

If a logic 1 has been written to any bit position of register R16A or register R17A and the corresponding I/O pins of port A or port B are externally pulled below the logic 0, (V_{IL}) level, a subsequent CPU read instruction of registers R16A or R17A will actually contain a logic 0 in the pulled down bit positions. The output pins will return to logic 1 if the pull down condition is removed.

If a logic 0 has been written to any bit position of the I/O registers and the external world wishes to pull these pins to a 1, the user should be aware that an impedance conflict will exist between the pull down transistor and the external driver.

CLOCK DIVIDE — SELECT

Select = 0 Input Clock = 4MHz max.
(Divided internally by 2)

Select = 1 Input Clock = 2MHz max.

The select pin is provided with an internal pull-up resistor such that the pin default condition is Select = 1.

INPUT CONTROL SIGNALS

Interfacing to the AY8930 requires the generation of only two of the three AY-3-8910A input control signals, BDIR and BC1. BC2 is shown on the pinout diagram for reference only; the pin is not internally connected.

The input control signals for the AY8930 are as follows:

BDIR	BC1	FUNCTION	DESCRIPTION
0	0	INACTIVE	INACTIVE. The PSG/CPU bus is inactive. DA7–DA0 are in a high impedance state.
0	1	READ FROM PSG	READ FROM PSG. This signal causes the contents of the register which is currently addressed to appear on the PSG/CPU bus. DA7–DA0 are in the output mode.
1	0	WRITE TO PSG	WRITE TO PSG. This signal indicates that the bus contains register data which should be latched into the currently addressed register. DA7–DA0 are in the input mode.
1	1	LATCH ADDRESS	LATCH ADDRESS. This signal indicates that the bus contains a register address which should be latched by the PSG. DA7–DA0 are in the input mode.

RESET

After a RESET condition the internal state of the PSG will be the following:

REGISTER	B7	B6	B5	B4	B3	B2	B1	B0
R0/R0A	0	0	0	0	0	0	0	0
R1/R1A	0	0	0	0	0	0	0	0
R2/R2A	0	0	0	0	0	0	0	0
R3/R3A	0	0	0	0	0	0	0	0
R4/R4A	0	0	0	0	0	0	0	0
R5/R5A	0	0	0	0	0	0	0	0
R6/R6A	0	0	0	0	0	0	0	0
R7/R7A	0	0	0	0	0	0	0	0
R10/R10A	#	#	0	0	0	0	0	0
R11/R11A	#	#	0	0	0	0	0	0
R12/R12A	#	#	0	0	0	0	0	0
R13/R13A	0	0	0	0	0	0	0	0
R14/R14A	0	0	0	0	0	0	0	0
R15/R15A/R15B	0	0	0	0	0	0	0	0
R16/R16A	0	0	0	0	0	0	0	0
R17/R17A	0	0	0	0	0	0	0	0
R0B	0	0	0	0	0	0	0	0
R1B	0	0	0	0	0	0	0	0
R2B	0	0	0	0	0	0	0	0
R3B	0	0	0	0	0	0	0	0
R4B	#	#	#	#	X	X	X	X
R5B	#	#	#	#	X	X	X	X
R6B	#	#	#	#	X	X	X	X
R7B	#	#	#	#	X	X	X	X
R10B	#	#	#	#	X	X	X	X
R11B	0	0	0	0	0	0	0	0
R12B	X	X	X	X	X	X	X	X
R17B	0	0	0	0	0	0	0	0

X indicates a don't care.

# indicates that there is no physical memory element for a bit; if read a 0 will be returned.

All counter work registers should be initialized to zeros.

The noise generator 17-bit shift register should be initialized to ones.

The noise value register should be initialized to zeros.

ELECTRICAL SPECIFICATIONS

Maximum Ratings*

Storage temperature . . . . −55°C to +150°C
Maximum temperature under bias . . . . +125°C
V_{DD} and all other input/output voltages
with respect to V_{SS} . . . . −0.3V to +7.0V

Standard Conditions

Free air ambient operating
temperature. . . . . 0°C to +70°C
V_{DD} . . . . +4.5V to +5.5V
V_{SS} . . . . 0.0V (Ground)

*Exceeding these ratings could case permanent damage to this device. This is a stress rating only and functional operation of this device at these conditions is not implied — operating ranges are specified in Standard Conditions. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC CHARACTERISTICS

CHARACTERISTICS	SYM	MIN	TYP	MAX	UNITS	CONDITIONS
Input Logic Levels						
Logic 0	V_{IL}	−0.3	—	+0.4	Volts	
Logic 1	V_{IH}	+2.4	—	V_{DD}	Volts	
Input Leakage						
Clock	—	—	—	10	µA	
BC1, BDIR	—	—	—	10	µA	
Inputs with Pullups A8, $\overline{\text{Reset}}$, Select	I_{IL}	10	—	100	µA	V_{IN} = +0.4V
Inputs with Pulldowns $\overline{\text{A9}}$	I_{IH}	10	—	50	µA	V_{IN} = +2.4V
I/O with Pullups						
A7–A0, B7–B0	I_{IL}	20	—	150	µA	V_{IN} = +0.4V
	V_{OH}	+2.4	—	V_{DD}	Volts	I_{OH} = 100µA w/100pF
	V_{OL}	0.0	—	+0.4	Volts	I_{OL} = 1.6mA w/100pF
Data/Address						
DA7–DA0	V_{OH}	+2.4	—	V_{DD}	Volts	I_{OH} = 100µA w/100pF
	V_{OL}	0.0	—	+0.4	Volts	I_{OL} = 1.6mA w/100pF
Power Supply	I_{DD}	—	—	85	mA	All inputs and outputs tied to V_{SS} or V_{DD}.

ELECTRICAL SPECIFICATIONS (continued)

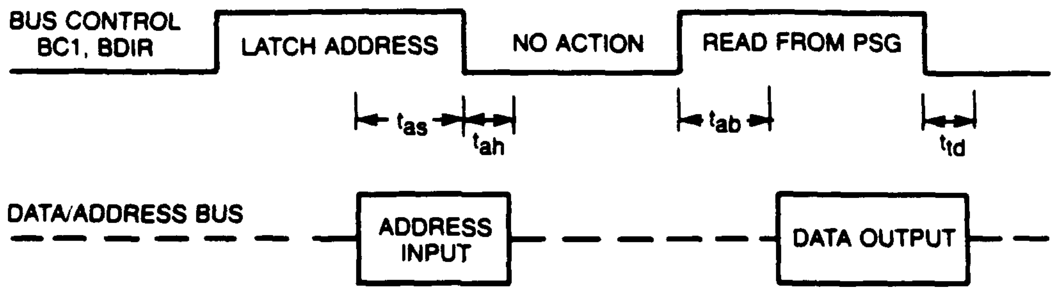
AC CHARACTERISTICS*

CHARACTERISTICS	SYM	MIN	TYP	MAX	UNITS	CONDITIONS
Clock Input						
Frequency	—	1	—	4	MHz	40/60 asymmetry
Rise/Fall Time	t_r, t_f	0	—	50	ns	allowed
Master Reset						
$\overline{\text{Reset}}$	t_{ms}	*	—	—	ns	*Two Clock Periods
Control Signals BC1, BC2, BDIR						
Skew	t_{cs}	—	—	40	ns	
Valid	t_{ccs}	300	—	—	ns	
Data Address Bus DA7–DA0, A8, $\overline{\text{A9}}$						
Address Setup Time	t_{as}	300	—	—	ns	
Address Hold Time	t_{ah}	65	—	—	ns	
Read Mode:						
Data Setup Time	t_{ab}	—	—	200	ns	
Data Hold Time	t_{td}	20	—	100	ns	
Write Mode:						
Data Setup Time	t_{ds}	300	—	—	ns	
Data Hold Time	t_{dh}	65	—	—	ns	
Input/Output Port IOA7–IOA0, IOB7–IOB0						
Output Mode:						
Data Setup Time	t_{pw}	500	—	—	ns	
Input Mode:						
Data Setup Time	t_{prs}	200	—	—	ns	
Data Hold Time	t_{prh}	65	—	—	ns	

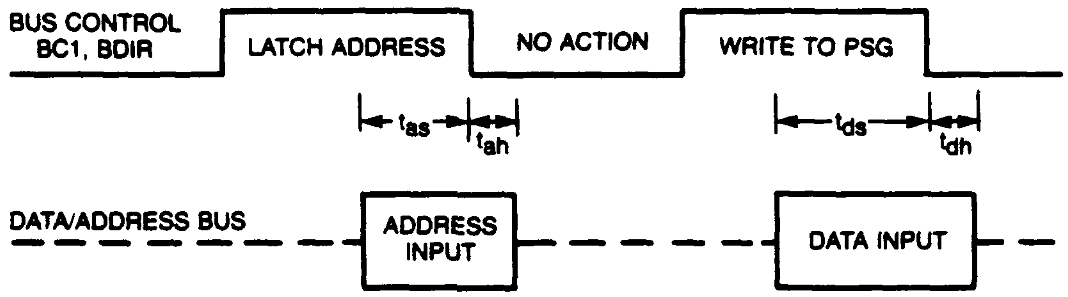
*The address/data read cycle is latch address followed by an inactive state then the read command. The address/data write cycle would be the same with the substitution of the write command in place of the read. An inactive state is required between each cycle (or active command).

TIMING DIAGRAMS

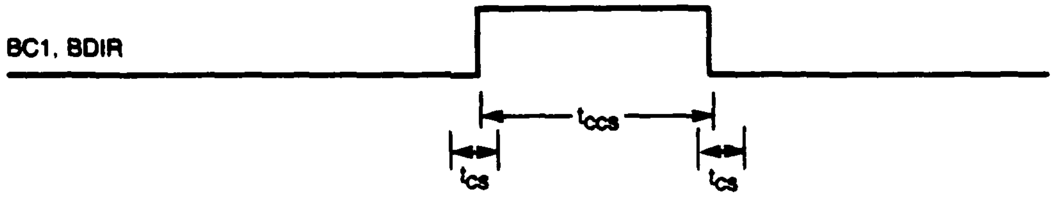
Read Mode



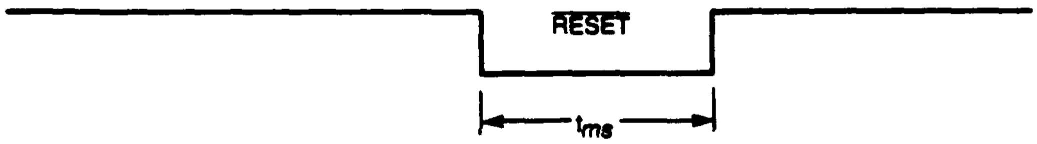
Write Mode



Bus Control Signals

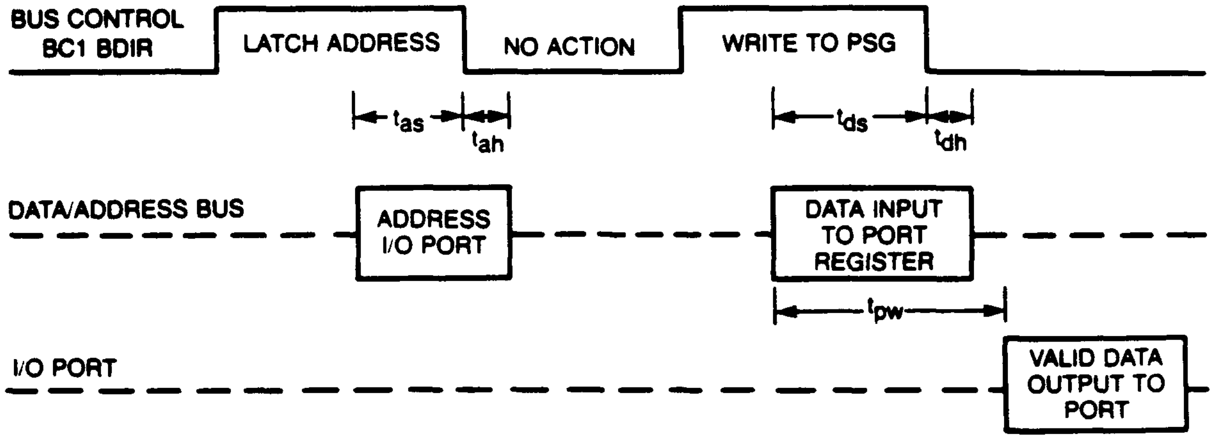


Master Reset

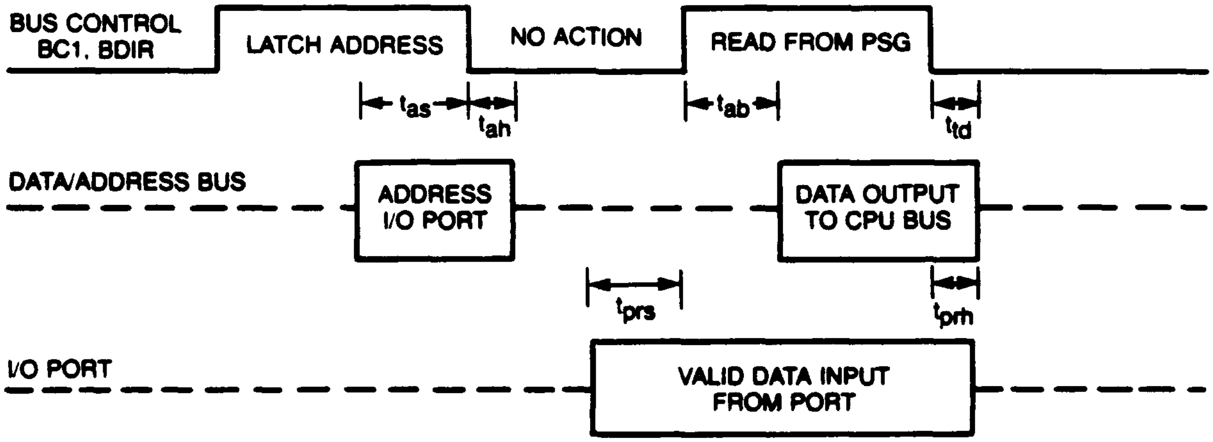


TIMING DIAGRAMS (continued)

I/O Port Output Mode: Write Data From CPU Bus To I/O Port*



I/O Port Input Mode: Write Data From I/O Port To CPU Bus*



*Assume the direction of the I/O port has already been determined via a write to the $\overline{\text{Enable}}$ register (R7A).

Appendix A - R65C22

Rockwell

R65C22
Versatile Interface Adapter (VIA)

DESCRIPTION

The R65C22 Versatile Interface Adapter (VIA) is a very flexible I/O control device. In addition, this device contains a pair of very powerful 16-bit interval timers, a serial-to-parallel/parallel-to serial shift register and input data latching on the peripheral ports. Expanded handshaking capability allows control of bidirectional data transfers between VIA's in multiple processor systems.

The R65C22 includes functions for programmed control of up to two peripheral devices (Ports A and B). These two program controlled 8-bit bidirectional peripheral I/O ports allow direct interfacing between the microprocessor and selected peripheral units. Each port has input data latching capability. Two programmable Data Direction Registers (A and B) allow selection of data direction (input or output) on an individual line basis.

The R65C22 also has two programmable 16-bit Interval Timer/Counters with latches. Timer 1 may be operated in a One-Shot Interrupt Mode with interrupts on each count-to-zero, or in a Free-Run Mode with a continuous series of evenly spaced interrupts. Timer 2 functions as both an interval and pulse counter. Serial data transfers are provided by a serial-to-parallel/parallel-to-serial shift register. Application versatility is further increased by various control registers, including—the Interrupt Flag Register, the Interrupt Enable Register, the Auxiliary Control Register and the Peripheral Control Register.

FEATURES

- Low power CMOS N-well silicon gate technology
- Fully compatible with NMOS 6522 devices
- Two 8-bit bidirectional I/O ports
- Two 16-bit programmable timer/counters
- Serial bidirectional peripheral I/O
- TTL compatible peripheral control lines
- Expanded "handshake" capability allows positive control of data transfers between processor and peripheral devices.
- Latched output and input registers on both I/O ports
- 1, 2, 3, and 4 MHz operation
- Commercial and industrial temperature range versions
- Single +5 Vdc power requirement
- Wide variety of packages
  - — 40-pin plastic and ceramic DIP
  - — 44-pin plastic leaded chip carrier (PLCC)

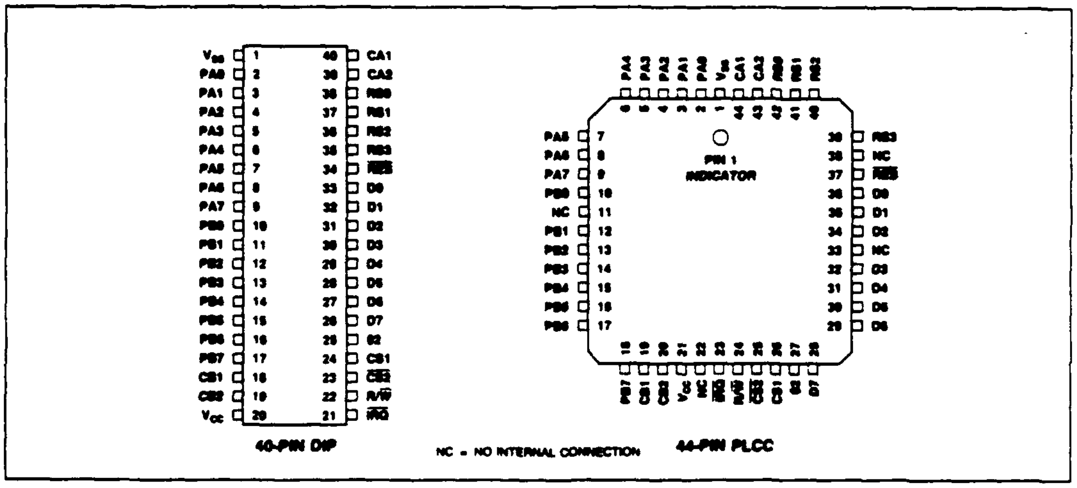


Figure 1. R65C22 Pin Assignments

ORDERING INFORMATION

Part Number:
R65C22 _ _ _

Temperature Range (T_L to T_H):
Blank = 0°C to +70°C
E = -40°C to +85°C

Frequency
1 = 1 MHz
2 = 2 MHz
3 = 3 MHz
4 = 4 MHz

Package
C = 40-Pin Ceramic DIP
P = 40-Pin Plastic DIP
J = 44-Pin Plastic Leaded Chip Carrier (PLCC)

INTERFACE SIGNALS

Figure 1 (on the front page) shows the R65C22 VIA pin assignments and Figure 2 groups the signals by functional interface.

RESET ($\overline{\mathrm{RES}}$)

Reset ($\overline{\mathrm{RES}}$) clears all internal registers (except T1 and T2 counters and latches, and the Shift Register (SR)). In the $\overline{\mathrm{RES}}$ condition, all peripheral interface lines (PA and PB) are placed in the input state. Also, the Timers (T1 and T2), SR and interrupt logic are disabled from operation.

INPUT CLOCK (PHASE 2)

The system Phase 2 (Φ2) Input Clock controls all data transfers between the R65C22 and the microprocessor.

READ/WRITE (R/$\overline{\mathrm{W}}$)

The direction of the data transfers between the R65C22 and the system processor is controlled by the R/$\overline{\mathrm{W}}$ line in conjunction with the CS1 and $\overline{\mathrm{CS2}}$ inputs. When R/$\overline{\mathrm{W}}$ is low (write operation) and the R65C22 is selected, data is transferred from the processor bus into the selected R65C22 register. When R/$\overline{\mathrm{W}}$ is high (read operation) and the R65C22 is selected, data is transferred from the selected R65C22 register to the processor bus.

DATA BUS (D0–D7)

The eight bidirectional Data Bus lines transfer data between the R65C22 and the microprocessor. During a read operation, the contents of the selected R65C22 internal register are transferred to the microprocessor via the Data Bus lines. During a write operation, the Data Bus lines serve as high impedance inputs over which data is transferred from the microprocessor to a selected R65C22 register. The Data Bus lines are in the high impedance state when the R65C22 is unselected.

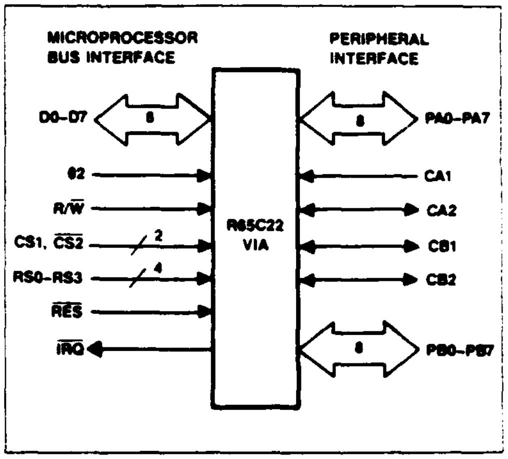


Figure 2. R65C22 VIA Interface Signals

CHIP SELECTS (CS1, $\overline{\mathrm{CS2}}$)

Normally, the two chip select lines are connected to the microprocessor address lines. This connection may be direct or through decoding. To access a selected R65C22 register, CS1 must be high (logic 1) and $\overline{\mathrm{CS2}}$ must be low (logic 0).

REGISTER SELECTS (RS0–RS3)

The Register Select inputs allow the microprocessor to select one of 16 internal registers within the R65C22. Refer to Table 1 for Register Select coding and a functional description.

INTERRUPT REQUEST ($\overline{\mathrm{IRQ}}$)

The Interrupt Request ($\overline{\mathrm{IRQ}}$) output signal is generated whenever an internal Interrupt Flag bit is set and the corresponding Interrupt Enable bit is a Logic 1. The Interrupt Request output is an open-drain configuration, thus allowing the $\overline{\mathrm{IRQ}}$ signal to be wire-ORed to a common microprocessor $\overline{\mathrm{IRQ}}$ input line.

PERIPHERAL PORT A (PA0–PA7)

Peripheral Data Port A is an 8-line, bidirectional bus for the transfer of data, control and status information between the R65C22 and a peripheral device. Each Peripheral Data Port bus line may be individually programmed as either an input or output under control of a Data Direction Register. Data flow direction may be selected on a line-by-line basis with intermixed input and output lines within the same port. When a "0" is written to any bit position of the Data Direction Register, the corresponding line will be programmed as an input. When a "1" is written into any bit position of the register, the corresponding data line will serve as an output. Polarity of the data output is determined by the Output Register, while input data may be latched into the Input Register under control of the CA1 line. All modes are program controlled by the microprocessor by way of the R65C22's internal control registers. Each Peripheral Data Port line represents two TTL loads in the input mode and will drive two standard TTL loads in the output mode. A typical output circuit for Peripheral Data Port A is shown in Figure 3.

PORT A CONTROL LINES (CA1, CA2)

Control lines CA1 and CA2 serve as interrupt inputs or handshake outputs for Peripheral Data Port A. Each line controls an internal Interrupt Flag with a corresponding Interrupt Enable bit. CA1 also controls the latching of Input Data on Port A. CA1 is a high impedance input, while CA2 represents two standard TTL loads in the input mode. In the output mode, CA2 will drive two standard TTL loads.

PORT B (PB0–PB7)

Peripheral Data Port B is an 8-line, bidirectional bus which is controlled by an Output Register, Input Register and Data Direction Register in a manner much the same as Data Port A. With respect to Port B, the output signal on line PB7 may be controlled by Timer 1 while Timer 2 may be programmed to count pulses on the PB6 line. Port B lines represent two standard TTL loads in the input mode and will drive two TTL loads in the output mode. Port B lines are also capable of sourcing 3.2 mA at 1.5 Vdc in the output mode. This allows the outputs to directly drive Darlington transistor circuits. A typical output circuit for Port B is shown in Figure 3.

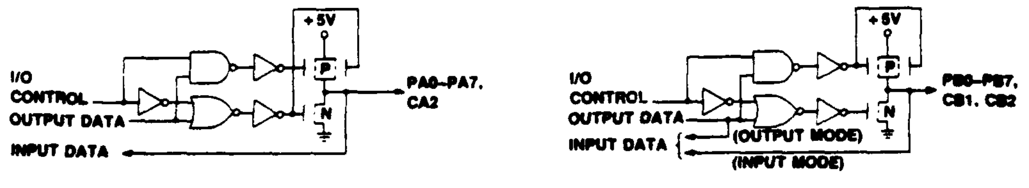
PORT B CONTROL LINES (CB1, CB2)

Control lines CB1 and CB2 serve as interrupt inputs or handshake outputs for Peripheral Data Port B. Like Port A, these two control lines control an internal Interrupt Flag with a corresponding Interrupt Enable bit. These lines also serve as a serial data port under control of the Shift Register (SR). Each control line represents two standard TTL loads in the input mode and can drive two TTL loads in the output mode. Note that CB1 and CB2 can drive Darlington transistor circuits, because they both will source 3.2 mA at 1.5 Vdc. Each line represents two standard TTL loads in the input mode and will drive two standard TTL loads in the output mode.

Table 1. R65C22 Register Addressing

Register Number	RS Coding				Register Desig.	Register/Description	
	RS3	RS2	RS1	RS0		Write (R/W̄ = L)	Read (R/W̄ = H)
0	0	0	0	0	ORB/IRB	Output Register B	Input Register B
1	0	0	0	1	ORA/IRA	Output Register A	Input Register A
2	0	0	1	0	DDRB	Data Direction Register B	
3	0	0	1	1	DDRA	Data Direction Register A	
4	0	1	0	0	T1C-L	T1 Low-Order Latches	T1 Low-Order Counter
5	0	1	0	1	T1C-H	T1 High-Order Counter	
6	0	1	1	0	T1L-L	T1 Low-Order Latches	
7	0	1	1	1	T1L-H	T1 High-Order Latches	
8	1	0	0	0	T2C-L	T2 Low-Order Latches	T2 Low-Order Counter
9	1	0	0	1	T2C-H	T2 High-Order Counter	
10	1	0	1	0	SR	Shift Register	
11	1	0	1	1	ACR	Auxiliary Control Register	
12	1	1	0	0	PCR	Peripheral Control Register	
13	1	1	0	1	IFR	Interrupt Flag Register	
14	1	1	1	0	IER	Interrupt Enable Register	
15	1	1	1	1	ORA/IRA	Output Register A*	Input Register A*

NOTE: *Same as Register 1 except no handshake.



a. Port A Data and Control Line Output Circuit

b. Port B Data and Control Line Output Circuit

Figure 3. Port A and B Output Circuits

FUNCTIONAL DESCRIPTION

The internal organization of the R65C22 VIA is illustrated in Figure 4.

PORT A AND PORT B OPERATION

The R65C22 VIA has two 8-bit bidirectional I/O ports (Port A and Port B) and each port has two associated control lines.

Each 8-bit peripheral port has a Data Direction Register (DDRA, DDRB) for specifying whether the peripheral pins are to act as inputs or outputs. A "0" in a bit of the Data Direction Register causes the corresponding peripheral pin to act as an input. A "1" causes the pin to act as an output.

Each peripheral pin is also controlled by a bit in the Output Register (ORA, ORB) and the Input Register (IRA, IRB). When the pin is programmed as an output, the voltage on the pin is controlled by the corresponding bit of the Output Register. A "1" in the Output Register causes the output to go high, and a "0" causes the output to go low. Data may be written into Output Register bits corresponding to pins which are programmed as inputs. In this case, however, the output signal is unaffected.

Reading a peripheral port causes the contents of the Input Register (IRA, IRB) to be transferred onto the Data Bus. With input latching disabled, IRA will always reflect the levels on the PA pins. With input latching enabled, IRA will reflect the levels on the PA pins at the time the latching occurred (via CA1).

The IRB register operates similar to the IRA register. However, for pins programmed as outputs there is a difference. When reading IRA, the *level on the pin* determines whether a "0" or a "1" is sensed. When reading IRB, however, the bit stored in the *output register*, ORB, is the bit sensed. Thus, for outputs which have large loading effects and which pull an output "1" down or which pull an output "0" up, reading IRA may result in reading a "0" when a "1" was actually programmed, and reading a "1" when a "0" was programmed. Reading IRB, on the other hand, will read the "1" or "0" level actually programmed, no matter what the loading on the pin.

Figures 5 through 8 illustrate the formats of the port registers.

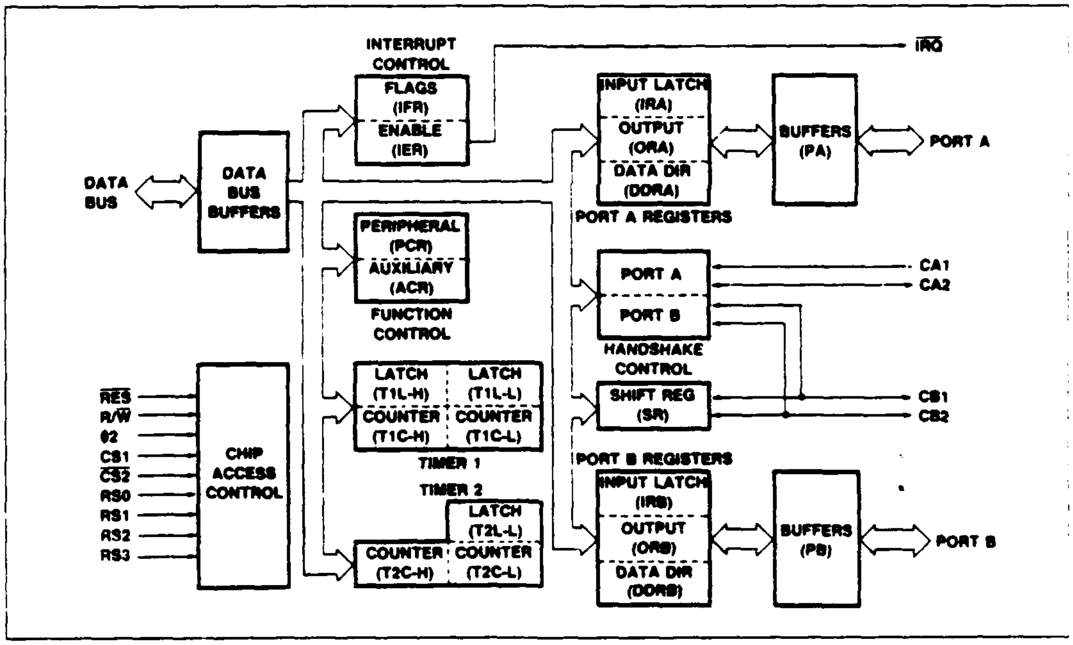


Figure 4. R65C22 VIA Block Diagram

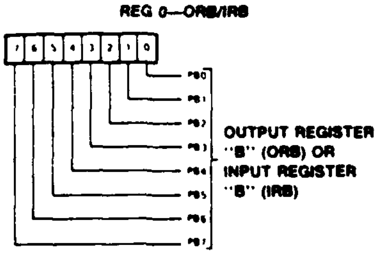
HANDSHAKE CONTROL OF DATA TRANSFERS

The R65C22 allows positive control of data transfers between the system processor and peripheral devices through the operation of "handshake" lines. Port A lines (CA1, CA2) handshake data on both a read and a write operation while the Port B lines (CB1, CB2) handshake on a write operation only.

Read Handshake

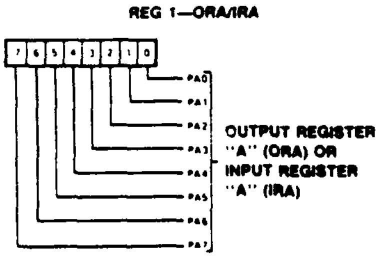
Positive control of data transfers from peripheral devices into the system processor can be accomplished very effectively using Read Handshaking. In this case, the peripheral device must generate the equivalent of a "Data Ready" signal to the processor signifying that valid data is present on the peripheral port. This signal normally interrupts the processor, which then reads the data, causing generation of a "Data Taken" signal. The peripheral device responds by making new data available This process continues until the data transfer is complete.

In the R65C22, automatic "Read" Handshaking is possible on the Peripheral A port only. The CA1 interrupt input pin accepts the "Data Ready" signal and CA2 generates the "Data Taken" signal. The "Data Ready" signal will set an internal flag which may interrupt the processor or which may be polled under program control. The "Data Taken" signal can either be a pulse or a level which is set low by the system processor and is cleared by the "Data Ready" signal. These options are shown in Figure 9 which illustrates the normal Read Handshake sequence.



PIN DATA DIRECTION SELECTION	WRITE	READ
DDRB = "1" (OUTPUT)	MPU WRITES OUTPUT LEVEL (ORB)	MPU READS OUTPUT REGISTER BIT IN ORB. PIN LEVEL HAS NO AFFECT
DDRB = "0" (INPUT) (INPUT LATCHING DISABLED)	MPU WRITES INTO ORB, BUT NO EFFECT ON PIN LEVEL UNTIL DDRB CHANGED	MPU READS INPUT LEVEL ON PB PIN
DDRB = "0" (INPUT) (INPUT LATCHING ENABLED)		MPU READS IRB BIT, WHICH IS THE LEVEL OF THE PB PIN AT THE TIME OF THE LAST CB1 ACTIVE TRANSITION

Figure 5. Output Register B (ORB), Input Register B (IRB)



PIN DATA DIRECTION SELECTION	WRITE	READ
DDRA = "1" (OUTPUT) (INPUT LATCHING DISABLED)	MPU WRITES OUTPUT LEVEL (ORA)	MPU READS LEVEL ON PA PIN
DDRA = "1" (OUTPUT) (INPUT LATCHING ENABLED)		MPU READS IRA BIT WHICH IS THE LEVEL OF THE PA PIN AT THE TIME OF THE LAST CA1 ACTIVE TRANSITION
DDRA = "0" (INPUT) (INPUT LATCHING DISABLED)	MPU WRITES INTO ORA, BUT NO EFFECT ON PIN LEVEL, UNTIL DDRA CHANGED	MPU READS LEVEL ON PA PIN
DDRA = "0" (INPUT) (INPUT LATCHING ENABLED)		MPU READS IRA BIT, WHICH IS THE LEVEL OF THE PA PIN AT THE TIME OF THE LAST CA1 ACTIVE TRANSITION

Figure 6. Output Register A (ORA), Input Register A (IRA)

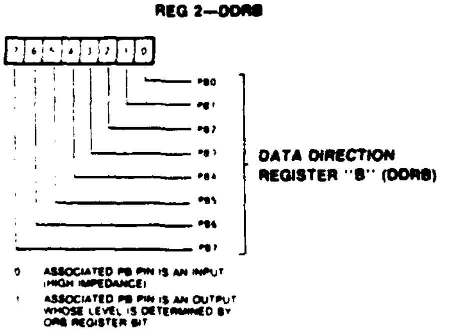

REG 2—DDRB
7
6
5
4
3
2
1
0
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
DATA DIRECTION REGISTER "B" (DDRB)
0 ASSOCIATED PB PIN IS AN INPUT (HIGH IMPEDANCE)
1 ASSOCIATED PB PIN IS AN OUTPUT WHOSE LEVEL IS DETERMINED BY ORB REGISTER BIT

Figure 7. Data Direction Register B (DDRB)

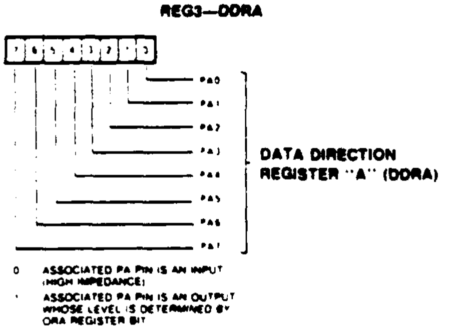

REG3—DDRA
7
6
5
4
3
2
1
0
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
DATA DIRECTION REGISTER "A" (DDRA)
0 ASSOCIATED PA PIN IS AN INPUT (HIGH IMPEDANCE)
1 ASSOCIATED PA PIN IS AN OUTPUT WHOSE LEVEL IS DETERMINED BY ORA REGISTER BIT

Figure 8. Data Direction Register A (DDRA)

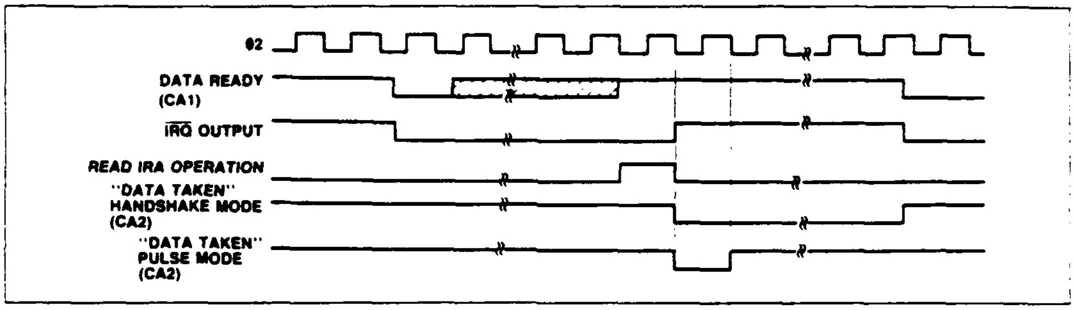

ϕ2
DATA READY (CA1)
IRQ OUTPUT
READ IRA OPERATION
"DATA TAKEN" HANDSHAKE MODE (CA2)
"DATA TAKEN" PULSE MODE (CA2)

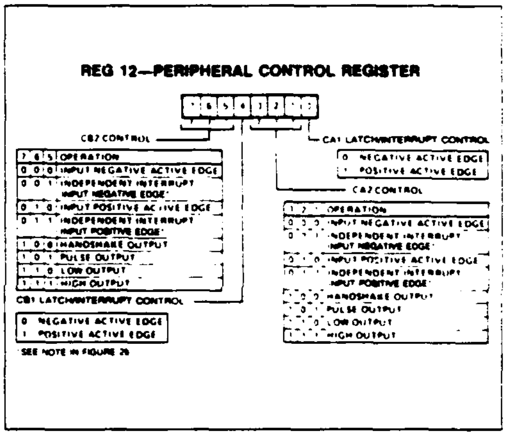
Figure 9. Read Handshake Timing (Port A Only)

Write Handshake

The sequence of operations which allows handshaking data from the system processor to a peripheral device is very similar to that described for Read Handshaking. However, for Write Handshaking, the R65C22 generates the "Data Ready" signal and the peripheral device must respond with the "Data Taken" signal. This can be accomplished on both the PA port and the PB port on the R65C22. CA2 or CB2 act as a "Data Ready" output in either the handshake mode or pulse mode and CA1 or CB1 accept the "Data Taken" signal from the peripheral device, setting the interrupt flag and clearing the "Data Ready" output. This sequence is shown in Figure 10.

Latching

The PA port and the PB port on the R65C22 can be enabled in the Auxiliary Control Register (Figure 14) to be latched by their individual port control lines (CA1, CB1). Latching is selectable to be on the rising or falling edge of the signal at each individual port control line. Selection of operating modes for CA1, CA2, CB1 and CB2 is accomplished by the Peripheral Control Register (Figure 11).



REG 12—PERIPHERAL CONTROL REGISTER

7	6	5	OPERATION
0	0	0	INPUT NEGATIVE ACTIVE EDGE
0	0	1	INDEPENDENT INTERRUPT INPUT NEGATIVE EDGE*
0	1	0	INPUT POSITIVE ACTIVE EDGE
0	1	1	INDEPENDENT INTERRUPT INPUT POSITIVE EDGE*
1	0	0	HANDSHAKE OUTPUT
1	0	1	PULSE OUTPUT
1	1	0	LOW OUTPUT
1	1	1	HIGH OUTPUT

CB1 LATCH/INTERRUPT CONTROL

0	NEGATIVE ACTIVE EDGE
1	POSITIVE ACTIVE EDGE

*SEE NOTE IN FIGURE 26

CA1 LATCH/INTERRUPT CONTROL

0	NEGATIVE ACTIVE EDGE
1	POSITIVE ACTIVE EDGE

CA2 CONTROL

3	2	1	OPERATION
0	0	0	INPUT NEGATIVE ACTIVE EDGE
0	0	1	INDEPENDENT INTERRUPT INPUT NEGATIVE EDGE*
0	1	0	INPUT POSITIVE ACTIVE EDGE
0	1	1	INDEPENDENT INTERRUPT INPUT POSITIVE EDGE*
1	0	0	HANDSHAKE OUTPUT
1	0	1	PULSE OUTPUT
1	1	0	LOW OUTPUT
1	1	1	HIGH OUTPUT

Figure 11. Peripheral Control Register (PCR)

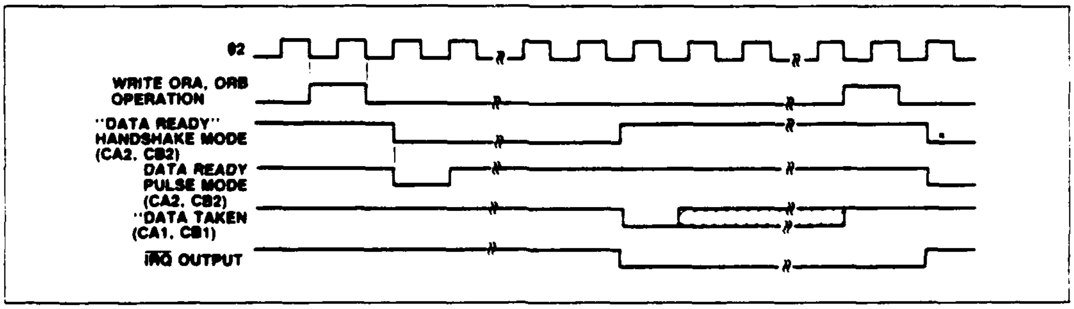

ϕ2
WRITE ORA, ORB OPERATION
"DATA READY" HANDSHAKE MODE (CA2, CB2)
DATA READY PULSE MODE (CA2, CB2)
"DATA TAKEN (CA1, CB1)
IRQ OUTPUT

Figure 10. Write Handshake Timing

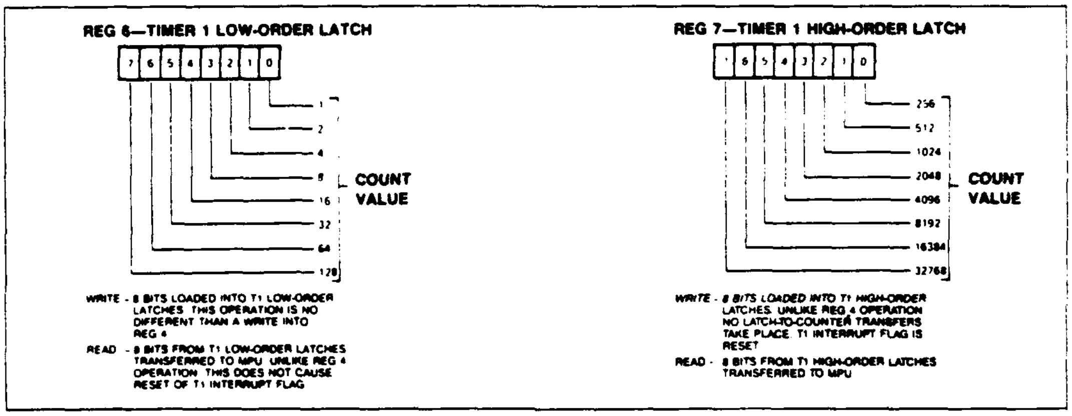
COUNTER/TIMERS

There are two independent 16-bit counter/timers (called Timer 1 and Timer 2) in the R65C22. Each timer is controlled by writing bits into the Auxiliary Control Register (ACR) to select the mode of operation (Figure 14).

Timer 1 Operation

Interval Timer T1 consists of two 8-bit latches (Figure 12) and a 16-bit counter (Figure 13). The latches store data which is to be loaded into the counter. After loading, the counter decrements at ϕ2 clock rate. Upon reaching zero, an interrupt flag is set, and IRQ goes low if the T1 interrupt is enabled. Timer 1 then disables any further interrupts and automatically transfers the contents of the latches into the counter and continues to decrement. In addition, the timer may be programmed to invert the output signal on peripheral pin PB7 each time it "times-out." Each of these modes is discussed separately below.

Note that the processor does not write directly into the low-order counter (T1C-L). Instead, this half of the counter is loaded automatically from the low order latch (T1L-L) when the processor writes into the high order counter (T1C-H). In fact, it may not be necessary to write to the low order counter in some applications since the timing operation is triggered by writing to the high order latch.



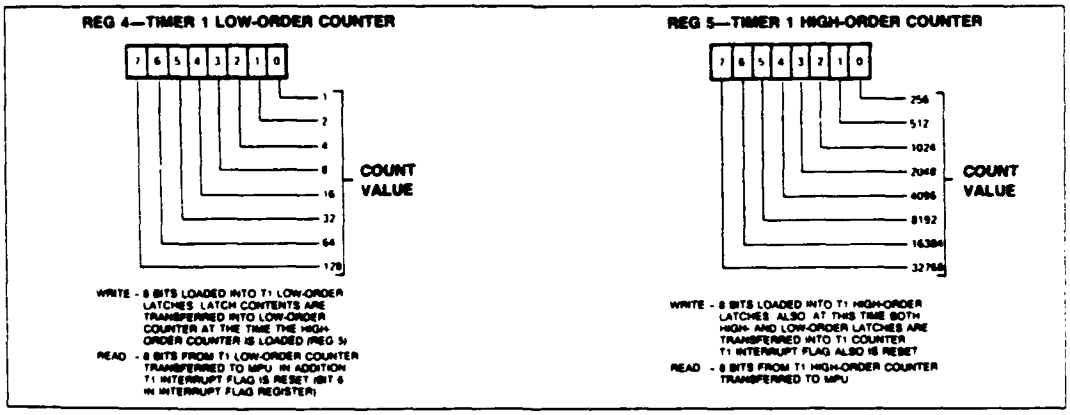
WRITE - 8 BITS LOADED INTO T1 LOW-ORDER LATCHES. THIS OPERATION IS NO DIFFERENT THAN A WRITE INTO REG 4

READ - 8 BITS FROM T1 LOW-ORDER LATCHES TRANSFERRED TO MPU. UNLIKE REG 4 OPERATION, THIS DOES NOT CAUSE RESET OF T1 INTERRUPT FLAG

WRITE - 8 BITS LOADED INTO T1 HIGH-ORDER LATCHES. UNLIKE REG 4 OPERATION NO LATCH-TO-COUNTER TRANSFERS TAKE PLACE. T1 INTERRUPT FLAG IS RESET

READ - 8 BITS FROM T1 HIGH-ORDER LATCHES TRANSFERRED TO MPU

Figure 12. Timer 1 (T1) Latch Registers



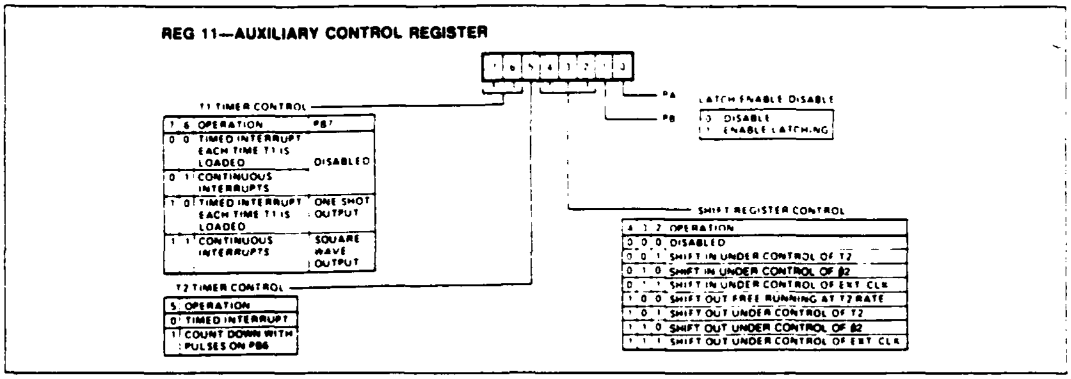
WRITE - 8 BITS LOADED INTO T1 LOW-ORDER LATCHES. LATCH CONTENTS ARE TRANSFERRED INTO LOW-ORDER COUNTER AT THE TIME THE HIGH-ORDER COUNTER IS LOADED (REG 5)

READ - 8 BITS FROM T1 LOW-ORDER COUNTER TRANSFERRED TO MPU. IN ADDITION T1 INTERRUPT FLAG IS RESET (BIT 6 IN INTERRUPT FLAG REGISTER)

WRITE - 8 BITS LOADED INTO T1 HIGH-ORDER LATCHES. ALSO AT THIS TIME BOTH HIGH- AND LOW-ORDER LATCHES ARE TRANSFERRED INTO T1 COUNTER. T1 INTERRUPT FLAG ALSO IS RESET

READ - 8 BITS FROM T1 HIGH-ORDER COUNTER TRANSFERRED TO MPU

Figure 13. Timer 1 (T1) Counter Registers



7	6	OPERATION	PB7
0	0	TIMED INTERRUPT EACH TIME T1 IS LOADED	DISABLED
0	1	CONTINUOUS INTERRUPTS	
1	0	TIMED INTERRUPT EACH TIME T1 IS LOADED	ONE SHOT OUTPUT
1	1	CONTINUOUS INTERRUPTS	SQUARE WAVE OUTPUT

5	OPERATION
0	TIMED INTERRUPT
1	COUNT DOWN WITH PULSES ON PB6

0	DISABLE
1	ENABLE LATCHING

4	3	2	OPERATION
0	0	0	DISABLED
0	0	1	SHIFT IN UNDER CONTROL OF T2
0	1	0	SHIFT IN UNDER CONTROL OF Ø2
0	1	1	SHIFT IN UNDER CONTROL OF EXT CLK
1	0	0	SHIFT OUT FREE RUNNING AT T2 RATE
1	0	1	SHIFT OUT UNDER CONTROL OF T2
1	1	0	SHIFT OUT UNDER CONTROL OF Ø2
1	1	1	SHIFT OUT UNDER CONTROL OF EXT CLK

Figure 14. Auxiliary Control Register (ACR)

Timer 1 One-Shot Mode

The Timer 1 one-shot mode generates a single interrupt for each timer load operation. As with any interval timer, the delay between the "write T1C-H" operation and generation of the processor interrupt is a direct function of the data loaded into the timing counter. In addition to generating a single interrupt, Timer 1 can be programmed to produce a single negative pulse on the PB7 peripheral pin. With the output enabled (ACR7 = 1) a "write T1C-H" operation will cause PB7 to go low. PB7 will return high when Timer 1 times out. The result is a single programmable width pulse.

Timing for the R65C22 interval timer one-shot modes is shown in Figure 15.

In the one-shot mode, writing into the T1L-H has no effect on the operation of Timer 1. However, it will be necessary to assure that the low order latch contains the proper data before initiating the count-down with a "write T1C-H" operation. When the processor writes into the high order counter (T1C-H), the T1 interrupt flag will be cleared, the contents of the low order latch will be transferred into the low order counter, and the timer will begin to decrement at system clock rate. If the PB7 output is enabled, this signal will go low on the falling edge of Ø2 following the write operation. When the counter reaches zero, the T1 interrupt flag will be set, the $\overline{IRQ}$ pin will go low (interrupt enabled), and the signal on PB7 will go high. At this time the counter will continue to decrement at system clock rate. This allows the system processor to read the contents of the counter to determine the time since interrupt. However, the T1 interrupt flag cannot be set again unless it has been cleared as described in this specification.

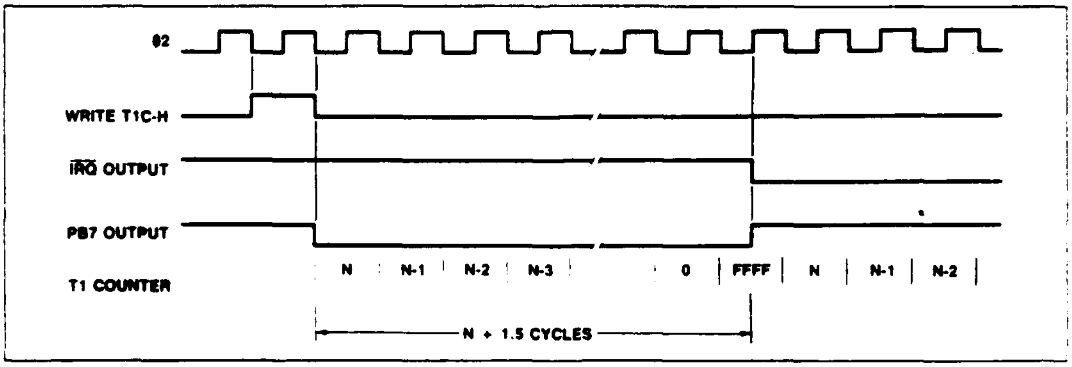


Figure 15. Timer 1 One-Shot Mode Timing

Timer 1 Free-Run Mode

The most important advantage associated with the latches in T1 is the ability to produce a continuous series of evenly spaced interrupts and the ability to produce a square wave on PB7 whose *frequency is not affected by variations in the processor interrupt* response time. This is accomplished in the "free-running" mode.

In the free-running mode, the interrupt flag is set and the signal on PB7 is inverted each time the counter reaches zero at which time the timer automatically transfers the contents of the latch into the counter (16 bits) and continues to decrement from there. The interrupt flag can be cleared by writing T1C-H or T1L-H, by reading T1C-L, or by writing directly into the flag as described later. However, it is not necessary to rewrite the timer to enable setting the interrupt flag on the next time-out.

All interval timers in the R65C22 are "re-triggerable". Rewriting *the counter will always re-initialize the time-out period. In fact,* the time-out can be prevented completely if the processor continues to rewrite the timer before it reaches zero. Timer 1 will operate in this manner if the processor writes into the high order counter (T1C-H). However, by loading the latches only, the processor can access the timer during each down-counting operation without affecting the time-out in process. Instead, the data loaded into the latches will determine the length of the next time-out period. This capability is particularly valuable in the free-running mode with the output enabled. In this mode, the signal on PB7 is inverted and the interrupt flag is set with each time-out. By responding to the interrupts with new data for the latches, the processor can determine the period of the next half cycle during each half cycle of the output signal on PB7. In this manner, very complex waveforms can be generated.

A precaution to take in the use of PB7 as the timer output concerns the Data Direction Register contents for PB7. *Both* DDRB bit 7 and ACR bit 7 must be 1 for PB7 to function as the timer output. If one is 1 and other is 0, then PB7 functions as a nor*mal outpin pin, controlled* by ORB bit 7.

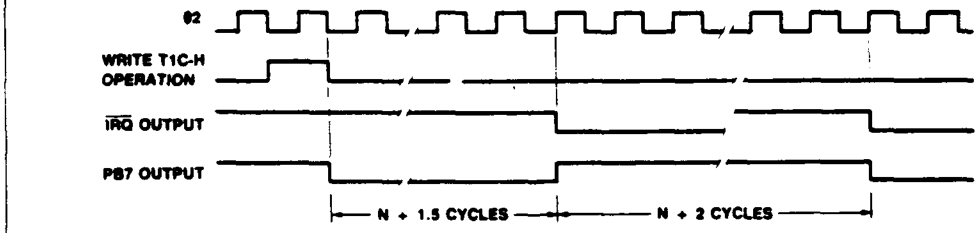


Figure 16. Timer 1 Free-Run Mode Timing

Timer 2 Operation

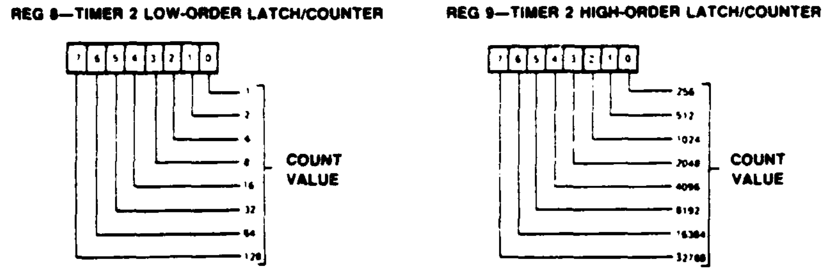
Timer 2 operates as an interval timer (in the "one-shot" mode only), or as a counter for counting negative pulses on the PB6 peripheral pin. A single control bit in the Auxiliary Control Register selects between these two modes. This timer is comprised of a "write-only" lower-order latch (T2L-L), a "read-only" low-order counter (T2C-L) and a read/write high order counter (T2C-H). The counter registers act as a 16-bit counter which decrements at ø2 rate. Figure 17 illustrates the T2 Latch/Counter Registers.

Timer 2 One-Shot Mode

As an interval timer, T2 operates in the "one-shot" mode similar to Timer 1. In this mode, T2 provides a single interrupt for each "write T2C-H" operation. After timing out, the counter will continue to decrement. However, setting of the interrupt flag is disabled *after initial time-out so that it will not be set by the counter* decrementing again through zero. The processor must rewrite T2C-H to enable setting of the interrupt flag. The interrupt flag is cleared by reading T2C-L or by writing T2C-H. Timing for this operation is shown in Figure 18.

Timer 2 Pulse Counting Mode

In the pulse counting mode, T2 counts a predetermined number of negative-going pulses on PB6. This is accomplished by first loading a number into T2. Writing into T2C-H clears the interrupt flag and allows the counter to decrement each time a pulse is applied to PB6. The interrupt flag is set when T2 counts down past zero. The counter will then continue to decrement with each pulse on PB6. However, it is necessary to rewrite T2C-H to allow the interrupt flag to set on a subsequent time-out. Timing for this mode is shown in Figure 19. The pulse must be low on the leading edge of ø2.



WRITE - 8 BITS LOADED INTO T2 LOW ORDER LATCH

READ - 8 BITS FROM T2 LOW ORDER COUNTER TRANSFERRED TO MPU T2 INTERRUPT FLAG IS RESET

WRITE - 8 BITS LOADED INTO T2 HIGH ORDER COUNTER ALSO LOW ORDER LATCH TRANSFERRED TO LOW ORDER COUNTER IN ADDITION T2 INTERRUPT FLAG IS RESET

READ - 8 BITS FROM T2 HIGH ORDER COUNTER TRANSFERRED TO MPU

Figure 17. Timer 2 (T2) Latch/Counter Registers

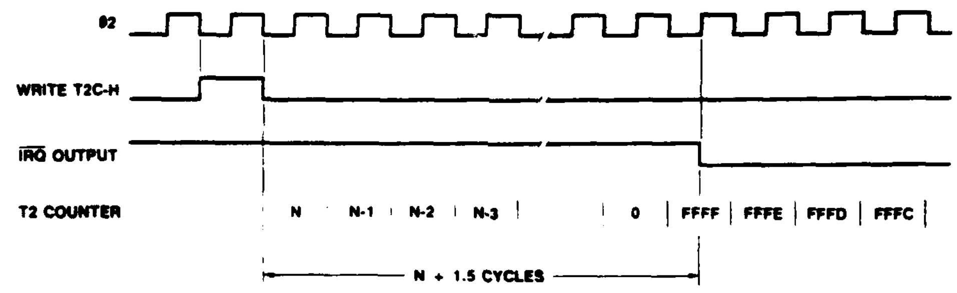


Figure 18. Timer 2 One-Shot Mode Timing

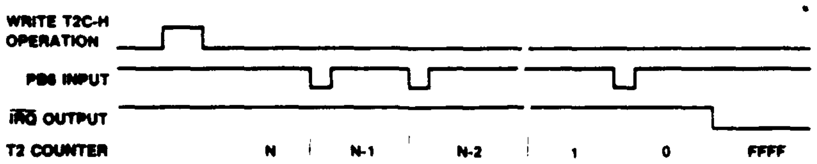


Figure 19. Timer 2 Pulse Counting Mode

SHIFT REGISTER OPERATION

The Shift Register (SR) performs serial data transfers into and out of the CB2 pin under control of an internal modulo-8 counter. Serial data transfer in and out of the Shift Register (SR) begin with the most significant bit (MSB) first. Shift pulses can be applied to the CB1 pin from an external source or, with the proper mode selection, shift pulses generated internally will appear on the CB1 pin for controlling external devices.

The control bits which select the various shift register operating modes are located in the Auxiliary Control Register. Figure 20 illustrates the configuration of the SR data bits and Figure 21 shows the SR control bits of the ACR.

SR Mode 0 — Shift Register Interrupt Disabled

Mode 0 disables the Shift Register interrupt. In this mode the microprocessor can write or read the SR and the SR will shift on each CB1 positive edge shifting in the value on CB2. In this mode the SR interrupt Flag is disabled (held to a logic 0).

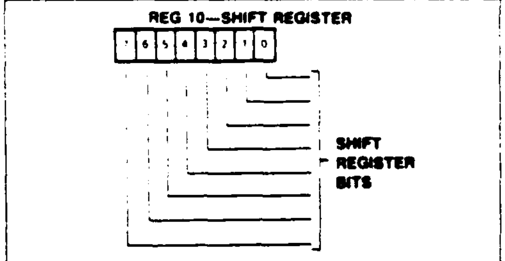
SR Mode 1 — Shift In Under Control of T2

In mode 1, the shifting rate is controlled by the low order 8 bits of T2 (Figure 22). Shift pulses are generated on the CB1 pin to control shifting in external devices. The time between transitions of this output clock is a function of the system clock period and the contents of the low order T2 latch (N).

The shifting operation is triggered by the read or write of the SR if the SR flag is set in the IFR. Otherwise the first shift will occur at the next time-out of T2 after a read or write of the SR. The input data should change before the positive-going edge of CB1 clock pulse. This data is shifted into the shift register during the Ø2 clock cycle following the positive-going edge of the CB1 clock pulse. After 8 CB1 clock pulses, the shift register interrupt flag will set and IRQ will go low.

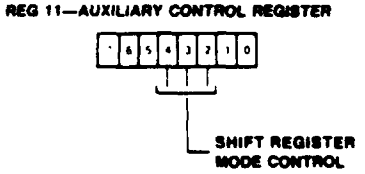
SR Mode 2 — Shift In Under Ø2 Control

In mode 2, the shift rate is a direct function of the system clock frequency (Figure 23). CB1 becomes an output which generates shift pulses for controlling external devices. Timer 2 operates as an independent interval timer and has no effect on SR. The shifting operation is triggered by reading or writing the Shift Register. Data is shifted, first into bit 0 and is then shifted into the next higher order bit of the shift register on the trailing edge of each Ø2 clock pulse. After 8 clock pulses, the shift register interrupt flag will be set, and the output clock pulses on CB1 will stop.



NOTES
1 WHEN SHIFTING OUT BIT 7 IS THE FIRST BIT OUT AND SIMULTANEOUSLY IS ROTATED BACK INTO BIT 0
2 WHEN SHIFTING IN BITS INITIALLY ENTER BIT 0 AND ARE SHIFTED TOWARDS BIT 7

Figure 20. Shift Register


REG 11—AUXILIARY CONTROL REGISTER
7 6 5 4 3 2 1 0
SHIFT REGISTER MODE CONTROL

4	3	2	OPERATION
0	0	0	DISABLED
0	0	1	SHIFT IN UNDER CONTROL OF T2
0	1	0	SHIFT IN UNDER CONTROL OF Ø2
0	1	1	SHIFT IN UNDER CONTROL OF EXT CLK
1	0	0	SHIFT OUT FREE RUNNING AT T2 RATE
1	0	1	SHIFT OUT UNDER CONTROL OF T2
1	1	0	SHIFT OUT UNDER CONTROL OF Ø2
1	1	1	SHIFT OUT UNDER CONTROL OF EXT CLK

Figure 21. Shift Register Modes

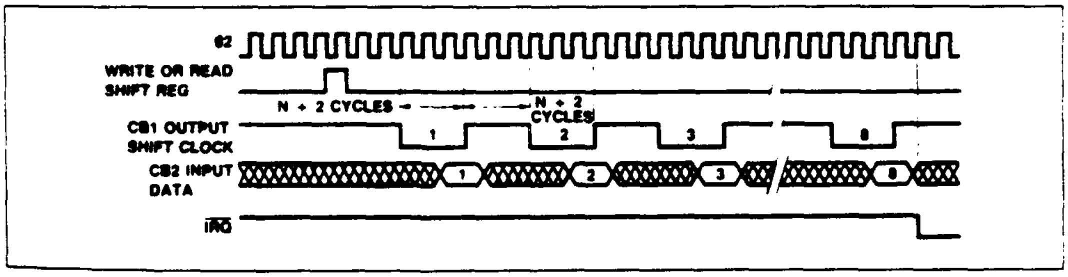


Figure 22. SR Mode 1 — Shift In Under T2 Control

SR Mode 3 — Shift In Under CB1 Control

In mode 3, external pin CB1 becomes an input (Figure 24). This allows an external device to load the shift register at its own pace. The shift register counter will interrupt the processor each time 8 bits have been shifted in. The shift register stops after 8 counts and must be reset to start again. Reading or writing the Shift Register resets the Interrupt Flag and initializes the SR counter to count another 8 pulses.

Note that the data is shifted during the first system clock cycle following the positive-going edge of the CB1 shift pulse. For this reason, data must be held stable during the first full cycle following CB1 going high.

SR Mode 4 — Shift Out Under T2 Control (Free-Run)

Mode 4 is very similar to mode 1 in which the shifting rate is set by T2. However, in mode 4 the SR Counter does not stop the shifting operation (Figure 25). Since the Shift Register bit 7 (SR7) is recirculated back into bit 0, the 8 bits loaded into the shift register will be clocked onto CB2 repetitively. In this mode the shift register counter is disabled.

SR Mode 5 — Shift Out Under T2 Control

In mode 5, the shift rate is controlled by T2 (as in mode 1). The shifting operation is triggered by the read or write of the SR if the SR flag is set in the IFR (Figure 26). Otherwise the first shift will occur at the next time-out of T2 after a read or write of the SR. However, with each read or write of the shift register the SR Counter is reset and 8 bits are shifted onto CB2. At the same time, 8 shift pulses are generated on CB1 to control shifting in external devices. After the 8 shift pulses, the shifting is disabled, the SR Interrupt Flag is set and CB2 remains at the last data level.

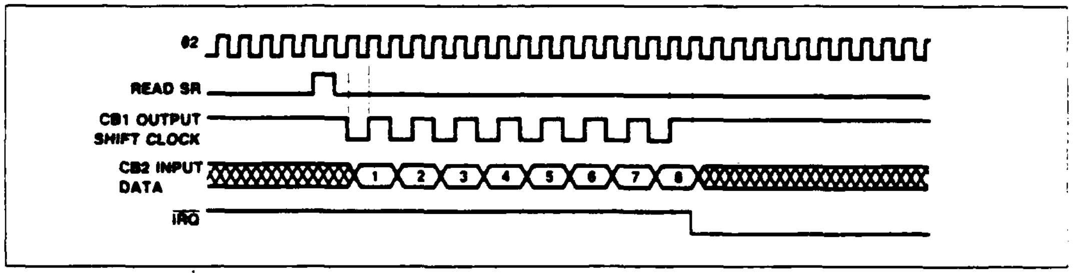


Figure 23. SR Mode 2 — Shift In Under φ2 Control

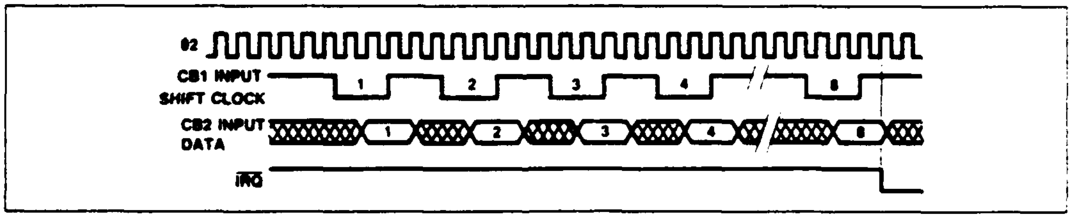


Figure 24. SR Mode 3 — Shift In Under CB1 Control

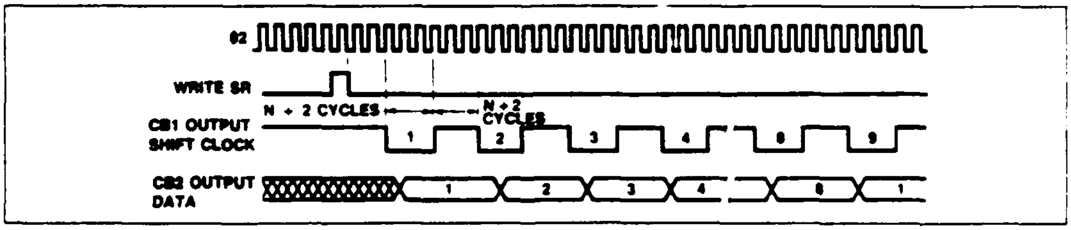


Figure 25. SR Mode 4 — Shift Out Under T2 Control (Free-Run)

SR Mode 6 — Shift Out Under ϕ2 Control

In mode 6, the shift rate is controlled by the ϕ2 system clock (Figure 27).

SR Mode 7 — Shift Out Under CB1 Control

In mode 7, shifting is controlled by pulses applied to the CB1 pin by an external device (Figure 28). The SR counter sets the SR Interrupt Flag each time it counts 8 pulses but it does not disable the shifting function. Each time the microprocessor, writes or reads the shift register, the SR Interrupt Flag is reset and the SR counter is initialized to begin counting the next 8 shift pulses on pin CB1. After 8 shift pulses, the Interrupt Flag is set. The microprocessor can then load the shift register with the next byte of data.

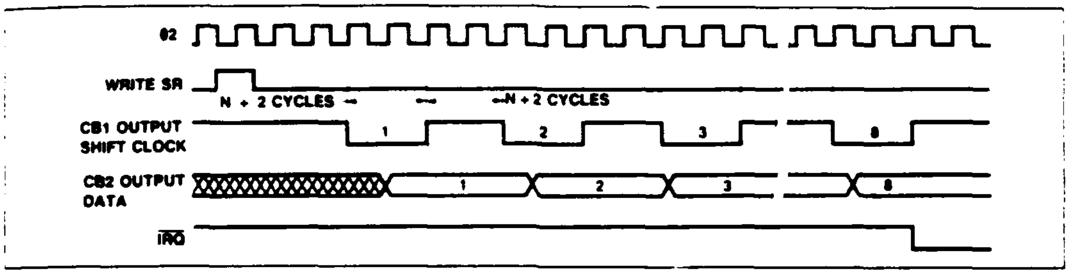


Figure 26. SR Mode 5 — Shift Out Under T2 Control

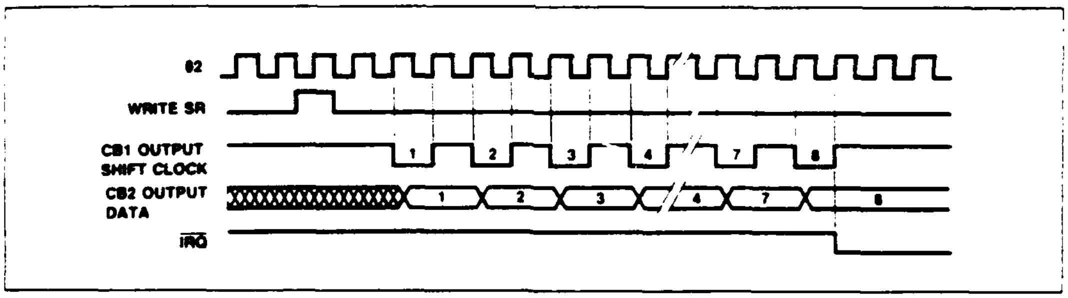


Figure 27. SR Mode 6 — Shift Out Under ϕ2 Control

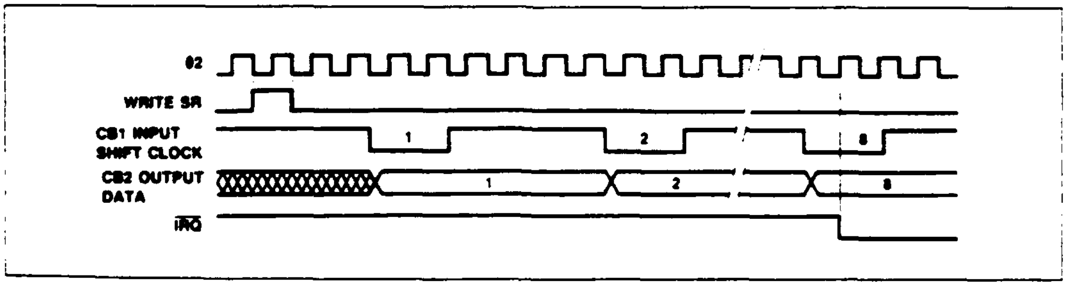


Figure 28. SR Mode 7 — Shift Out Under CB1 Control

INTERRUPT OPERATION

Controlling interrupts within the R65C22 involves three principal operations. These are flagging the interrupts, enabling interrupts and signaling to the processor that an active interrupts exists within the chip. Interrupt f.ags are set in the Interrupt Flag Register (IFR) by conditions detected within the R65C22 or on inputs to the R65C22. These flags normally remain set until the interrupt has been serviced. To determine the source of an interrupt, the microprocessor must examine these flags in order, from highest to lowest priority.

Associated with each interrupt flag is an interrupt enable bit in the Interrupt Enable Register (IER). This can be set or cleared by the processor to enable interrupting the processor from the corresponding interrupt flag. If an interrupt flag is set to a logic 1 by an interrupting condition, and the corresponding interrupt enable bit is set to a 1, the Interrupt Request Output ($\overline{IRQ}$) will go low. $\overline{IRQ}$ is an "open-collector" output which can be "wire-OR'ed" with other devices in the system to interrupt the processor.

Interrupt Flag Register (IFR)

In the R65C22, all the interrupt flags are contained in one register, i.e., the IFR (Figure 29). In addition, bit 7 of this register will be read as a logic 1 when an interrupt exists within the chip. This allows very convenient polling of several devices within a system to locate the source of an interrupt.

The Interrupt Flag Register (IFR) may be read directly by the processor. In addition, individual flag bits may be cleared by writing a "1" into the appropriate bit of the IFR. When the proper chip select and register signals are applied to the chip, the contents of this register are placed on the data bus. Bit 7 indicates the status of the $\overline{IRQ}$ output. This bit corresponds to the logic function: $\overline{IRQ}$ = IFR6 x IER6 + IFR5 x IER5 + IFR4 x IER4 + IFR3 x IER3 + IFR2 x IER2 + IFR1 x IER1 + IFR0 x IER0.

Note:

x = logic AND, + = Logic OR.

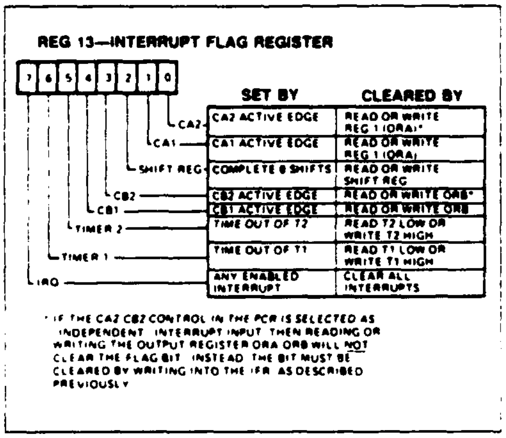
The IFR bit 7 is not a flag. Therefore, this bit is not directly cleared by writing a logic 1 into it. It can only be cleared by clearing all the flags in the register or by disabling all the active interrupts as discussed in the next section.

Interrupt Enable Register (IER)

For each interrupt flag in IFR, there is a corresponding bit in the Interrupt Enable Register (IER) (Figure 30). Individual bits in the IER can be set or cleared to facilitate controlling individual interrupts without affecting others. This is accomplished by writing to the (IER) after bit 7 set or cleared to, in turn, set or clear selected enable bits. If bit 7 of the data placed on the system data bus during this write operation is a 0, each 1 in bits 6 through 0 clears the corresponding bit in the Interrupt Enable Register. For each zero in bits 6 through 0, the corresponding bit is unaffected.

Selected bits in the IER can be set by writing to the IER with bit 7 in the data word set to a 1. In this case, each 1 in bits 6 through 0 will set the corresponding bit. For each zero, the corresponding bit will be unaffected. This individual control of the setting and clearing operations allows very convenient control of the interrupts during system operation.

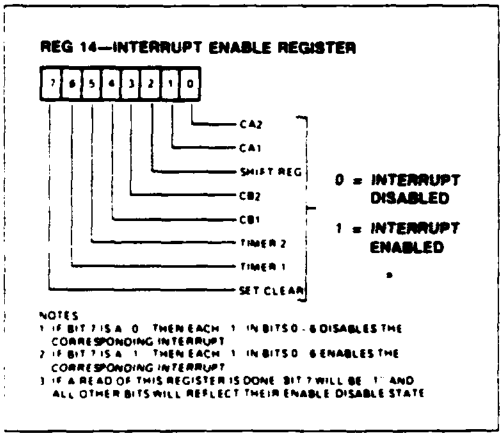
In addition to setting and clearing IER bits, the contents of this register can be read at any time. Bit 7 will be read as a logic 1, however.



	SET BY	CLEARED BY
CA2	CA2 ACTIVE EDGE	READ OR WRITE REG 1 (ORA)*
CA1	CA1 ACTIVE EDGE	READ OR WRITE REG 1 (ORA)
SHIFT REG	COMPLETE 8 SHIFTS	READ OR WRITE SHIFT REG
CB2	CB2 ACTIVE EDGE	READ OR WRITE ORB*
CB1	CB1 ACTIVE EDGE	READ OR WRITE ORB
TIMER 2	TIME OUT OF T2	READ T2 LOW OR WRITE T2 HIGH
TIMER 1	TIME OUT OF T1	READ T1 LOW OR WRITE T1 HIGH
IRQ	ANY ENABLED INTERRUPT	CLEAR ALL INTERRUPTS

* IF THE CA2 CB2 CONTROL IN THE PCR IS SELECTED AS INDEPENDENT INTERRUPT INPUT THEN READING OR WRITING THE OUTPUT REGISTER ORA ORB WILL NOT CLEAR THE FLAG BIT INSTEAD THE BIT MUST BE CLEARED BY WRITING INTO THE IFR AS DESCRIBED PREVIOUSLY

Figure 29. Interrupt Flag Register (IFR)



NOTES
1 IF BIT 7 IS A 0 THEN EACH 1 IN BITS 0 - 6 DISABLES THE CORRESPONDING INTERRUPT
2 IF BIT 7 IS A 1 THEN EACH 1 IN BITS 0 6 ENABLES THE CORRESPONDING INTERRUPT
3 IF A READ OF THIS REGISTER IS DONE BIT 7 WILL BE 1 AND ALL OTHER BITS WILL REFLECT THEIR ENABLE DISABLE STATE

Figure 30. Interrupt Enable Register (IER)

SWITCHING CHARACTERISTICS

(V_{CC} = 5.0 Vdc ±5%, V_{SS} = 0, T_A = T_L to T_H, unless otherwise noted)

PERIPHERAL INTERFACE TIMING

Parameter	Symbol	Min.	Max.	Unit	Figure
Rise and Fall Time for CA1, CB1, CA2 and CB2 Input Signals	t_r, t_f	—	1.0	µs	—
Delay Time, Clock Negative Transition to CA2 Negative Transition (read handshake or pulse mode)	t_{CA2}	—	1.0	µs	31a, 31b
Delay Time, Clock Negative Transition to CA2 Positive Transition (pulse mode)	t_{RS1}	—	1.0	µs	31a
Delay Time, CA1 Active Transition to CA2 Positive Transition (handshake mode)	t_{RS2}	—	2.0	µs	31b
Delay Time, Clock Positive Transition to CA2 or CB2 Negative Transition (write handshake)	t_{WHS}	0.05	1.0	µs	31c, 31d
Delay Time, Periphral Data Valid to CB2 Negative Transition	t_{DS}	0.20	1.5	µs	31c, 31d
Delay Time, Clock Positive Transition to CA2 or CB2 Positive Transition (pulse mode)	t_{RS3}	—	1.0	µs	31c
Delay Time, CA1 or CB1 Active Transition to CA2 or CB2 Positive Transition (handshake mode)	t_{RS4}	—	2.0	µs	31d
Delay Time Required from CA2 Output to CA1 Active Transition (handshake mode)	t_{21}	400	—	ns	31d
Setup Time, Peripheral Data Valid to CA1 or CB1 Active Transition (input latching)	t_{IL}	300	—	ns	31e
CA1, CB1 Setup Prior to Transition to Arm Latch	t_{AL}	300	—	ns	31e
Peripheral Data Hold After CA1, CB1 Transition	t_{PDH}	150	—	ns	31e
Shift-Out Delay Time — Time from ϕ2 Falling Edge to CB2 Data Out	t_{SR1}	—	300	ns	31f
Shift-In Setup Time — Time from CB2 Data In to ϕ2 Rising Edge	t_{SR2}	300	—	ns	31g
External Shift Clock (CB1) Setup Time Relative to ϕ2 Trailing Edge	t_{SR3}	100	T_{CY}	ns	31g
Pulse Width — PB6 Input Pulse	t_{IPW}	2 × T_{CY}	—		31i
Pulse Width — CB1 Input Clock	t_{ICW}	2 × T_{CY}	—		31h
Pulse Spacing — PB6 Input Pulse	t_{IPS}	2 × T_{CY}	—		31i
Pulse Spacing — CB1 Input Pulse	t_{ICS}	2 × T_{CY}	—		31h

BUS TIMING

Parameter	Symbol	1 MHz		2 MHz		3 MHz		4 MHZ		Unit	Figure
		Min	Max	Min	Max	Min	Max	Min	Max		
Cycle Time	t_{CY}	1000	—	500	—	330	—	250	—	ns	32a, 32b
Phase 2 Pulse Width High	t_{PWH}	470	—	240	—	160	—	120	—	ns	
Phase 2 Pulse Width Low	t_{PWL}	470	—	240	—	160	—	120	—	ns	
Phase 2 Transition	$t_{R,F}$	—	30	—	30	—	30	—	30	ns	
Read											
Select, R/$\overline{W}$ Setup	t_{ACR}	160	—	90	—	65	—	45	—	ns	32a
Select, R/$\overline{W}$ Hold	t_{CAR}	0	—	0	—	0	—	0	—	ns	
Data Bus Delay	t_{CDR}	—	320	—	150	—	130	—	75	ns	
Data Bus Hold	t_{HR}	10	—	10	—	10	—	10	—	ns	
Peripheral Data Setup	t_{PCR}	300	—	150	—	110	—	75	—	ns	
Write											
Select R/$\overline{W}$ Setup	t_{ACW}	160	—	90	—	65	—	45	—	ns	32b
Select, R/$\overline{W}$ Hold	t_{CAW}	0	—	0	—	0	—	0	—	ns	
Data Bus Setup	t_{DCW}	195	—	75	—	65	—	45	—	ns	
Data Bus Hold	t_{HW}	10	—	10	—	10	—	10	—	ns	
Peripheral Data Delay	t_{CPW}	—	1000	—	500	—	330	—	250	ns	

PERIPHERAL INTERFACE WAVEFORMS

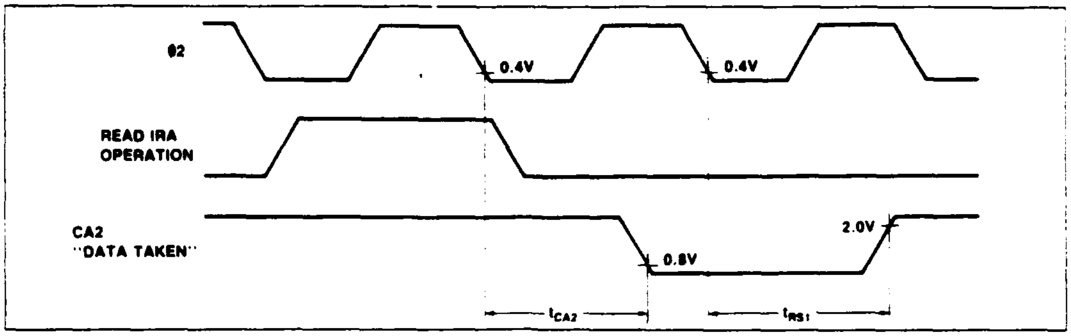


Figure 31a. CA2 Timing for Read Handshake, Pulse Mode

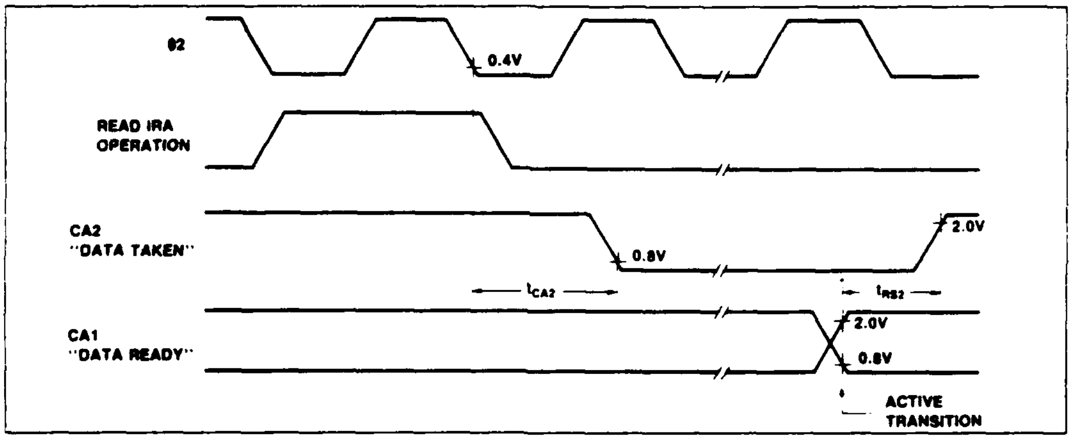


Figure 31b. CA2 Timing for Read Handshake, Handshake Mode

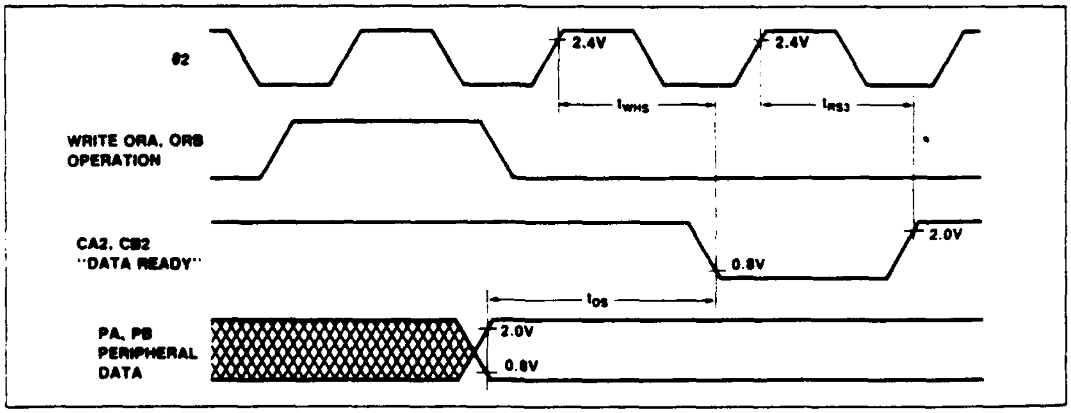


Figure 31c. CA2, CB2 Timing for Write Handshake, Pulse Mode

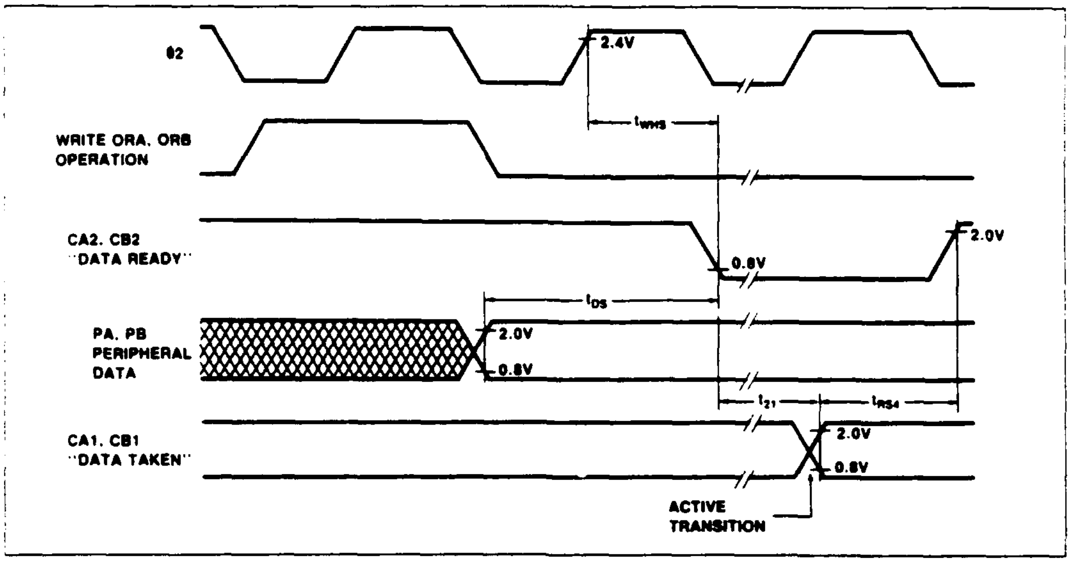


Figure 31d. CA2, CB2 Timing for Write Handshake, Handshake Mode

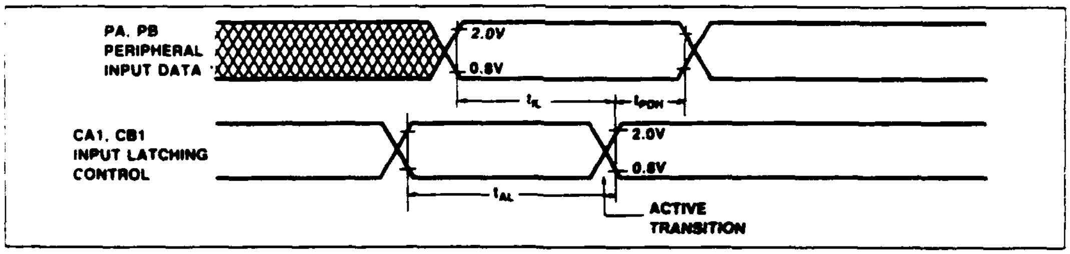


Figure 31e. Peripheral Data Input Latching Timing

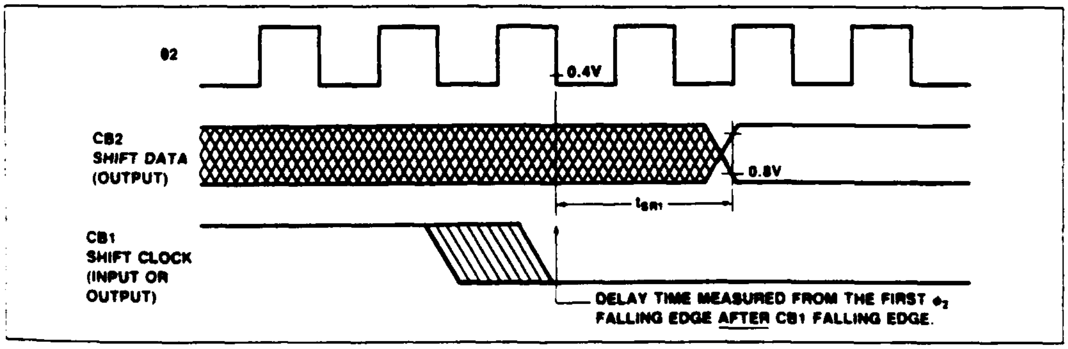

Φ2
0.4V
CB2 SHIFT DATA (OUTPUT)
0.8V
t_{SR1}
CB1 SHIFT CLOCK (INPUT OR OUTPUT)
DELAY TIME MEASURED FROM THE FIRST ϕ_2 FALLING EDGE AFTER CB1 FALLING EDGE.

Figure 31f. Timing for Shift Out with Internal or External Shift Clocking

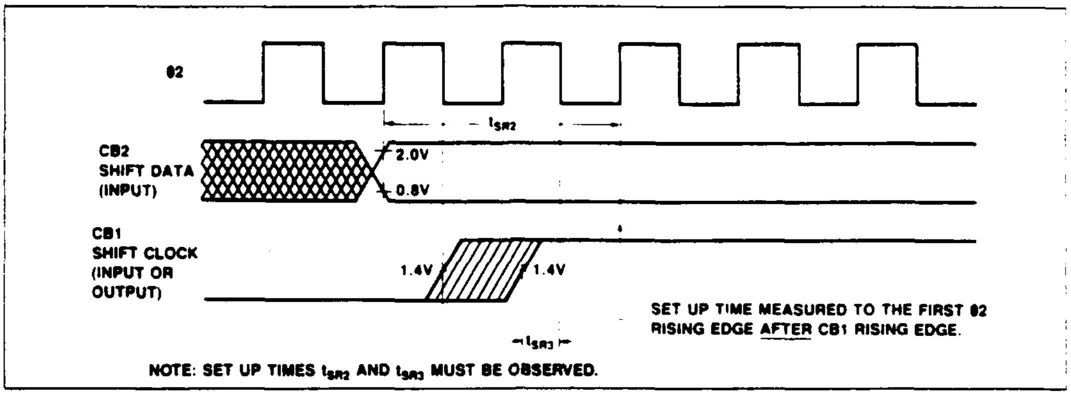

ϕ2
t_{SR2}
CB2
SHIFT DATA
(INPUT)
2.0V
0.8V
CB1
SHIFT CLOCK
(INPUT OR
OUTPUT)
1.4V
1.4V
t_{SR3}
SET UP TIME MEASURED TO THE FIRST ϕ2 RISING EDGE AFTER CB1 RISING EDGE.
NOTE: SET UP TIMES t_{SR2} AND t_{SR3} MUST BE OBSERVED.

Figure 31g. Timing for Shift In with Internal or External Shift Clocking

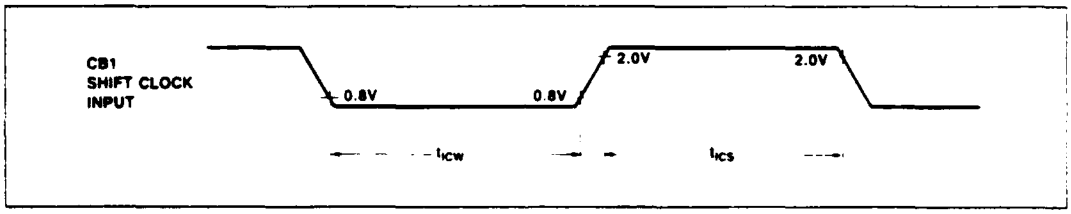


Figure 31h. External Shift Clock Timing

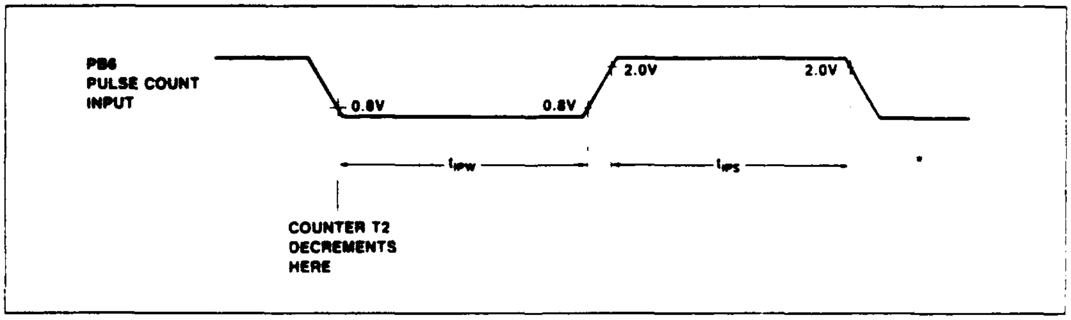


Figure 31i. Pulse Count Input Timing

BUS TIMING WAVEFORMS

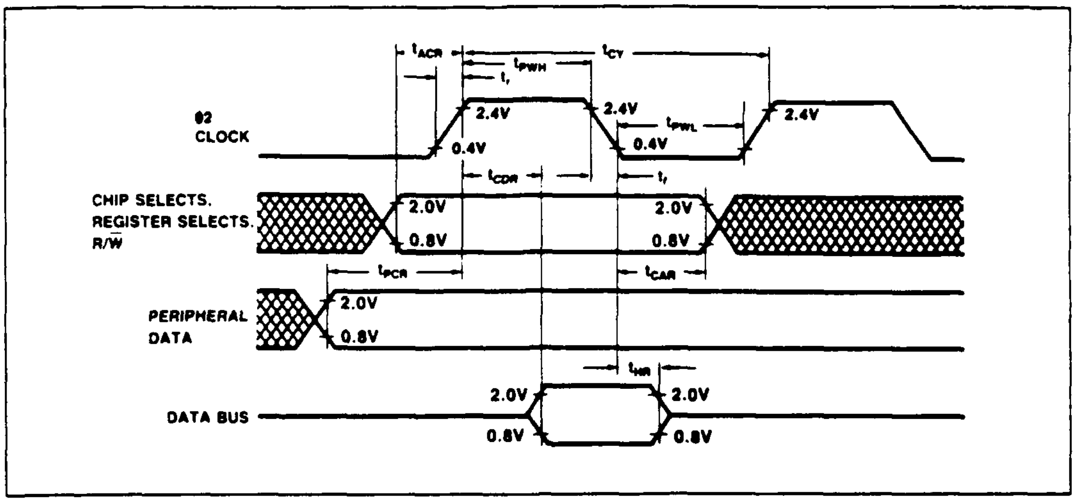


Figure 32a. Read Timing Waveforms

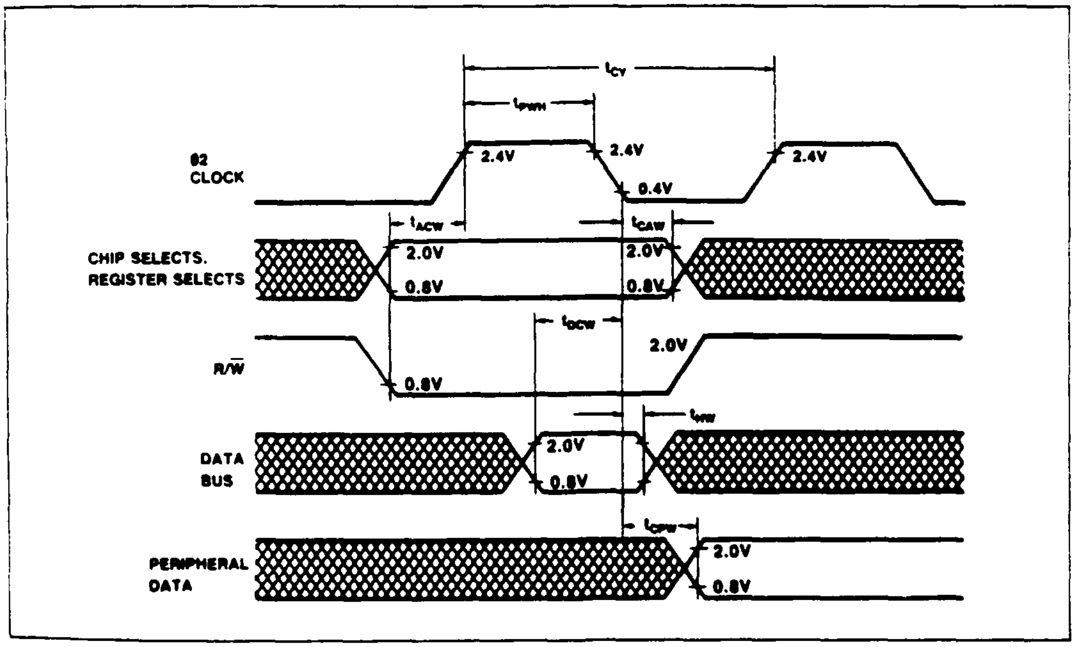


Figure 32b. Write Timing Waveforms

ABSOLUTE MAXIMUM RATINGS*

Parameter	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	Vdc
Input Voltage	V_{IN}	−0.3 to V_{CC} +0.3	Vdc
Output Voltage	V_{OUT}	−0.3 to V_{CC} +0.3	Vdc
Operating Temperature Commercial Industrial	T_A	 0 to +70 −40 to +85	°C
Storage Temperature	T_{STG}	−55 to +150	°C

***NOTE:** Stresses above those listed 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 indicated in the other sections of this document is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

OPERATING CONDITIONS

Parameter	Symbol	Value
Supply Voltage	V_{CC}	5V ±5%
Temperature Range Commercial Industrial	T_A	T_L to T_H 0°C to 70°C −40°C to +85°C

ELECTRICAL CHARACTERISTICS

(V_{CC} = 5.0 Vdc ±5%, V_{SS} = 0, T_A = T_L to T_H, unless otherwise noted)

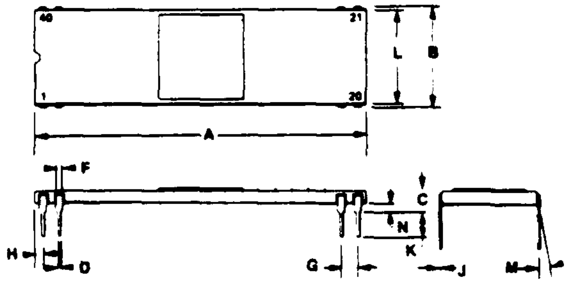
Parameter	Symbol	Min.	Typ.[3]	Max.	Unit[1]	Test Conditions
Input High Voltage Logic ϕ_2	V_{IH}	 +2.0 +2.4	 — —	 V_{CC} V_{CC}	V	
Input Low Voltage Logic ϕ_2	V_{IL}	 −0.3 −0.4	 — —	 +0.8 +0.4	V	
Input Leakage Current $R/\overline{W}$, $\overline{\text{RES}}$, RS0, RS1, RS2, RS3, CS1, $\overline{\text{CS2}}$, CA1, $\phi 2$	I_{IN}	—	±1	±2.5	µA	V_{IN} = 0V to V_{CC} V_{CC} = 5.25V
Input Leakage Current for Three-State Off D0–D7	I_{TSI}	—	±2	±10	µA	V_{IN} = 0.4V to 2.4V V_{CC} = 5.25V
Input High Current PA0–PA7, CA2, PB0–PB7, CB1, CB2	I_{IH}	−200	−400	—	µA	V_{IH} = 2.4V
Input Low Current PA0–PA7, CA2, PB0–PB7, CB1, CB2	I_{IL}	—	−2	−2.6	mA	V_{IL} = 0.4V
Output High Voltage All outputs PB0–PB7, CB1 and CB2 (Darlington Drive)	V_{OH}	 2.4 1.5	 — —	 — —	 V V	V_{CC} = 4.75V I_{LOAD} = 200 µA I_{LOAD}[2] = −3.2 mA
Output Low Voltage PA0–PA7, CA2, PB0–PB7, CB1, CB2, D0–D7, $\overline{\text{IRQ}}$	V_{OL}	—	—	+0.4	V	V_{CC} = 4.75V I_{LOAD} = 3.2 mA I_{LOAD} = 1.6 mA
Output High Current (Sourcing) Logic PB0–PB7, CB1 and CB2 (Darlington Drive)	I_{OH}	 −200 −3.2	 −1500 −6	 — —	 µA mA	 V_{OH} = 2.4V V_{OH} = 1.5V
Output Low Current (Sinking)	I_{OL}	3.2	—	—	mA	V_{OL} = 0.4V
Output Leakage Current (Off State) $\overline{\text{IRQ}}$	I_{OFF}	—	1	±10	µA	V_{OH} = 2.4V V_{CC} = 5.25V
Power Dissipation	P_D	—	7	10	mW/MHz	
Input Capacitance D0–D7, PA0–PA7, CA1, CA2, PB0–PB7, CB1, CB2 $R/\overline{W}$, $\overline{\text{RES}}$, RS0, RS1, RS2, RS3, CS1, $\overline{\text{CS2}}$, $\phi 2$	C_{IN}	 — — —	 — — —	 10 7 20	 pF pF pF	V_{CC} = 5.0V V_{IN} = 0V f = 2 MHz T_A = 25°C
Output Capacitance	C_{OUT}	—	—	10	pF	

Notes: .

1. All units are direct current (DC) except for capacitance
2. Negative sign indicates outward current flow, positive indicates inward flow
3. Typical values shown for V_{CC} = 5.0V and T_A = 25°C

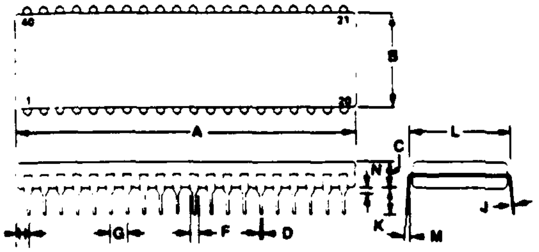
PACKAGE DIMENSIONS

40-PIN CERAMIC DIP



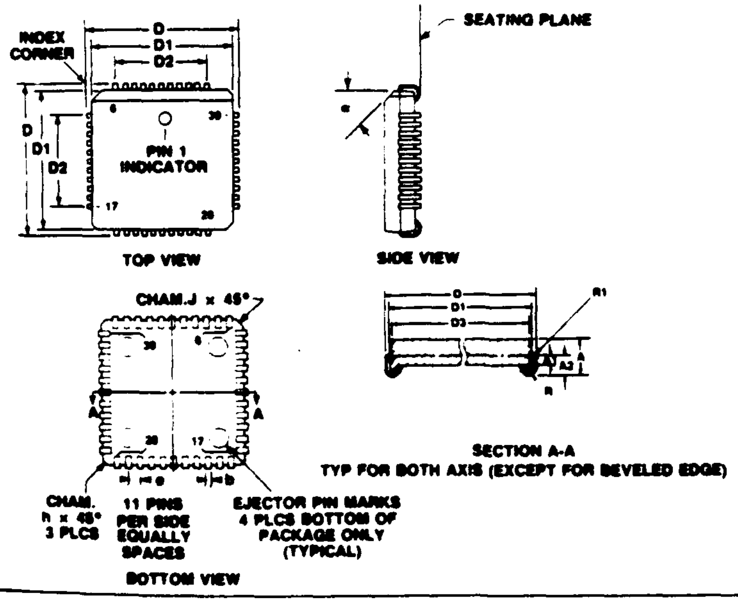
DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	50.29	51.31	1.980	2.020
B	15.11	15.88	0.595	0.625
C	2.54	4.19	0.100	0.165
D	0.38	0.53	0.015	0.021
F	0.76	1.27	0.030	0.050
G	2.54 BSC		0.100 BSC	
H	0.76	1.78	0.030	0.070
J	0.20	0.33	0.008	0.013
K	2.54	4.19	0.100	0.165
L	14.60	15.37	0.575	0.605
M	0°	10°	0°	10°
N	0.51	1.52	0.020	0.060

40-PIN PLASTIC DIP



DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	51.28	52.32	2.040	2.080
B	13.72	14.22	0.540	0.560
C	3.55	5.08	0.140	0.200
D	0.36	0.51	0.014	0.020
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.30	0.008	0.012
K	3.30	4.32	0.130	0.170
L	15.24 BSC		0.600 BSC	
M	7°	10°	7°	10°
N	0.51	1.02	0.020	0.040

44-PIN PLASTIC LEADED CHIP CARRIER (PLCC)

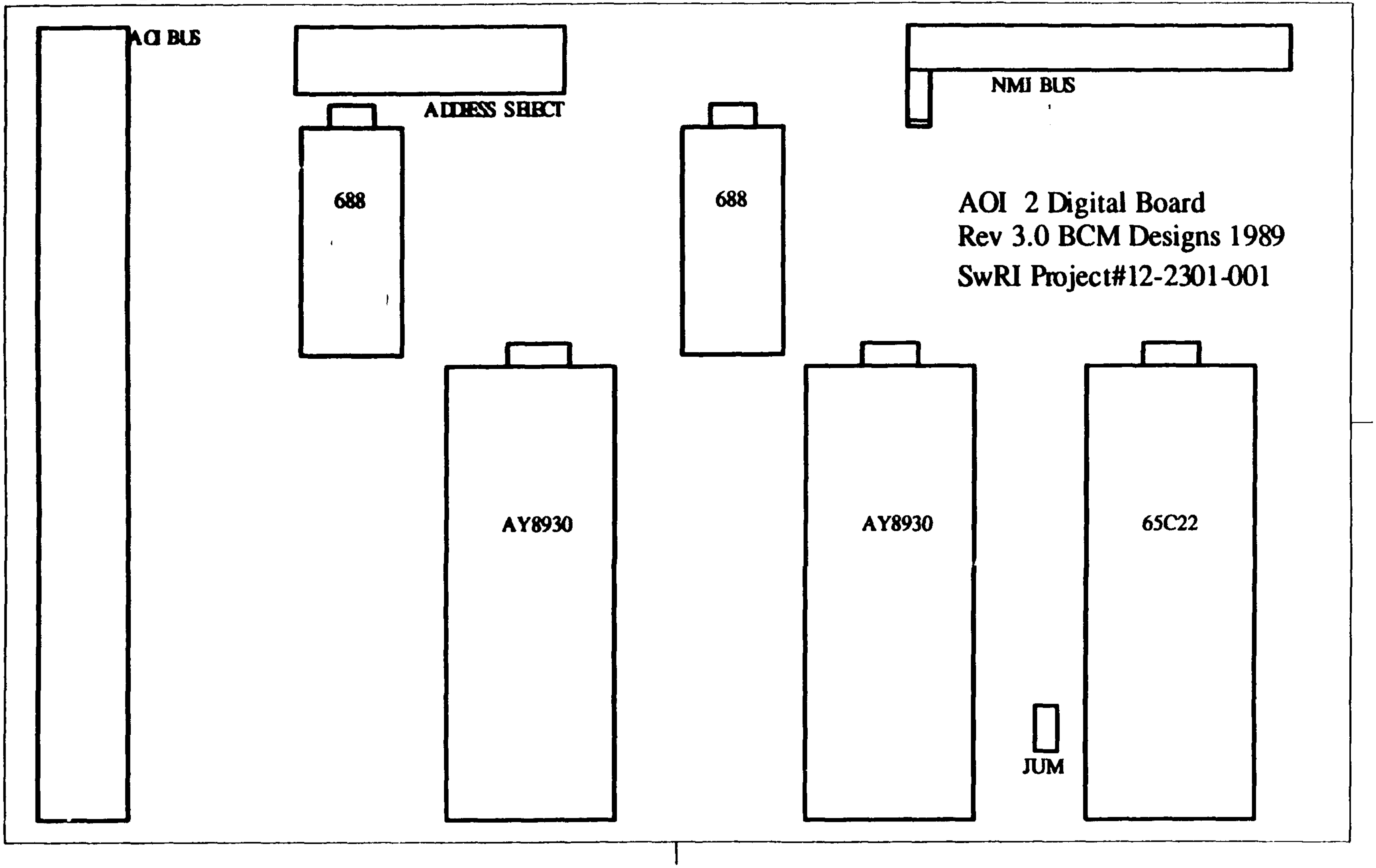


DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	4.14	4.39	0.163	0.173
A1	1.37	1.47	0.054	0.058
A2	2.31	2.46	0.091	0.097
b	0.457 TYP		0.018 TYP	
D	17.45	17.60	0.687	0.693
D1	16.46	16.56	0.648	0.652
D2	12.62	12.78	0.497	0.503
D3	15.75 REF		0.620 REF	
e	1.27 BSC		0.050 BSC	
h	1.15 TYP		0.045 TYP	
J	0.25 TYP		0.010 TYP	
α2	45° TYP		45° TYP	
R	0.89 TYP		0.035 TYP	
R1	0.25 TYP		0.010 TYP	

R65NC22/R65C22 DIFFERENCES

R65C22	R65NC22
1. Register select lines are decoded during $\overline{\phi 2}$.	1. Register select lines are decoded during $\overline{\phi 2}$ only if $\overline{CS2}$ is active low.
2. CB1 *must not change during* last 100 ns of $\phi 2$. CB1 must have a pulse width greater than one period.	2. CB1 can change anytime but is sampled only during $\overline{\phi 2}$. CB1 must have a pulse greater than one period.
3. PB0–PB7 and CB1, CB2 have active pull-ups.	3. PB0–PB7 and CB1, CB2 have passive pull ups (≈ 3 KΩ)
4. PB0–PB7, CB1 and CB2 represent two standard TTL loads in the input mode and will drive two standard TTL loads in the output mode.	4 PB0–PB7, CB1 and CB2 represent one standard TTL load in the input mode and will drive one standard TTL load in the output mode

Appendix A - PCB silkscreens



Digital Board SILK
Fri, Sep 15, 1989 10:24 AM
Component Side Scale150%

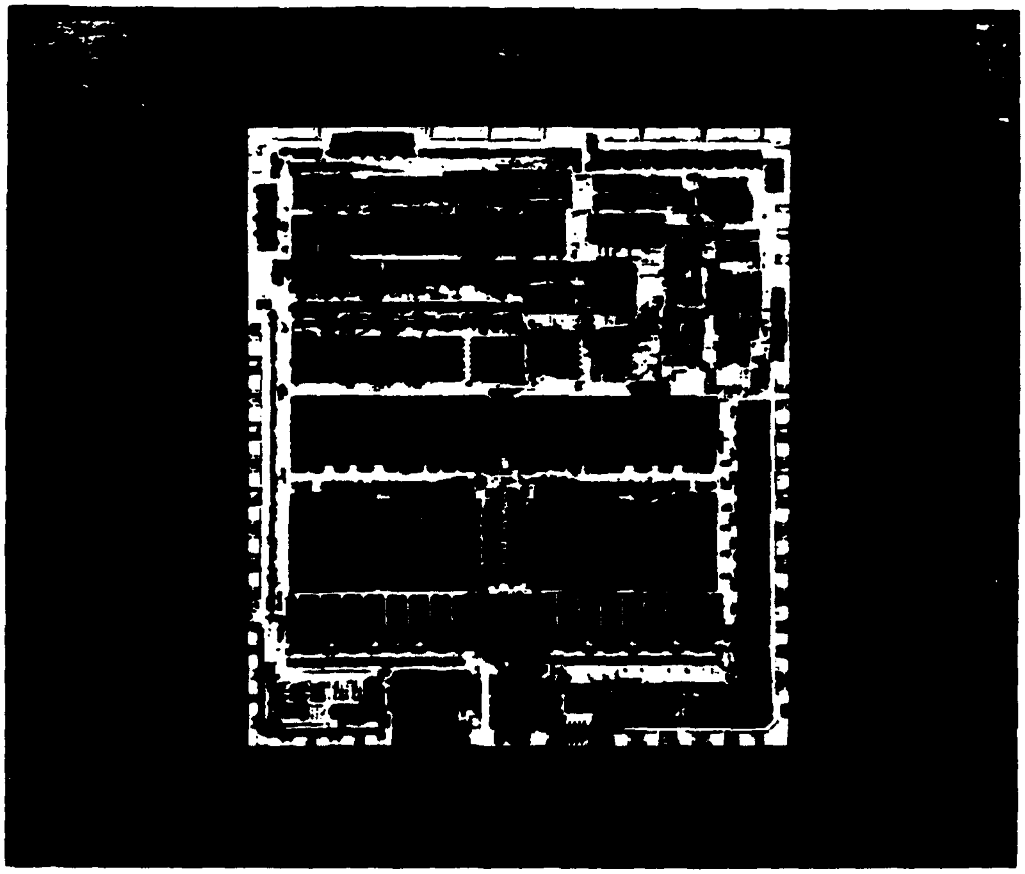
Appendix B - 68HC11

MOTOROLA
SEMICONDUCTOR
TECHNICAL DATA

MC68HC11A8

Advance Information

HCMOS Single-Chip Microcomputer



This document contains information on a new product. Specifications and information herein are subject to change without notice.

MOTOROLA

TABLE OF CONTENTS

Paragraph Number — Title — Page Number

Section 1
Introduction

Section 2
CPU Registers, Functional Pin Description, Operating Modes, Input/Output Programming, and Memory

TABLE OF CONTENTS
(Continued)

Section 3
Resets, Interrupts, and Low Power Modes

Section 4
Parallel I/O and System Configuration

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Concluded)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Continued)

LIST OF TABLES

SECTION 1
INTRODUCTION

The HCMOS MC68HC11A8 is an advanced single-chip microcomputer (MCU) with highly sophisticated on-chip peripheral functions. New design techniques are used to achieve a nominal bus speed of two megahertz. In addition, the fully static design allows operation at frequencies down to dc, further reducing its low power consumption.

1.1 FEATURES

The following are some of the hardware and software highlights.

Hardware Features

- 8K Bytes of ROM
- 512 Bytes of EEPROM
- 256 Bytes of RAM (All Saved During Standby) Mappable to Any 4K Boundary
- Enhanced 16-Bit Timer System:
  - Four Stage Programmable Prescaler
  - Three Input Capture Functions
  - Five Output Compare Functions
- An 8-Bit Pulse Accumulator Circuit
- An Enhanced NRZ Serial Communications Interface (SCI)
- A Serial Peripheral Interface (SPI)
- Eight Channel, 8-Bit Analog-to-Digital Converter
- Real Time Interrupt Circuit
- Computer Operating Properly (COP) Watchdog System
- Available in Dual-in-Line or Leaded Chip Carrier Packages

Software Features

- Enhanced M6800/M6801 Instruction Set
- 16 × 16 Integer and Fractional Divide Features
- Bit Manipulation
- WAIT Mode
- STOP Mode

1.2 GENERAL DESCRIPTION

The MC68HC11A8 is a single-chip microcomputer that utilizes HCMOS technolcgy to provide the low-power characteristics and high noise immunity of CMOS plus the high-speed operation of HMOS. On-chip memory systems include a 8K byte ROM, 512 bytes of electrically erasable programmable ROM (EEPROM), and 256 bytes of static RAM. The MC68HC11A8 microcomputer also provides highly sophisticated, on-chip peripheral functions including: an 8-channel analog-to-digital (A/D) converter, a serial communications interface (SCI) subsystem, and a serial peripheral interface (SPI) subsystem.

The timer system provides three input capture lines, five output compare lines, and a real time interrupt circuit.

Other features include: a pulse accumulator which can be used to count external events (event counting mode) or measure an external period; a computer operating properly (COP) watchdog system which helps protect against software failures; a clock monitor system which causes generation of a system reset in case the clock is lost or running too slow; an illegal opcode detection circuit which provides an unmaskable interrupt if an illegal opcode fetch is detected; and two power saving standby modes, STOP and WAIT.

A block diagram of the MC68HC11A8 is given in Figure 1-1.

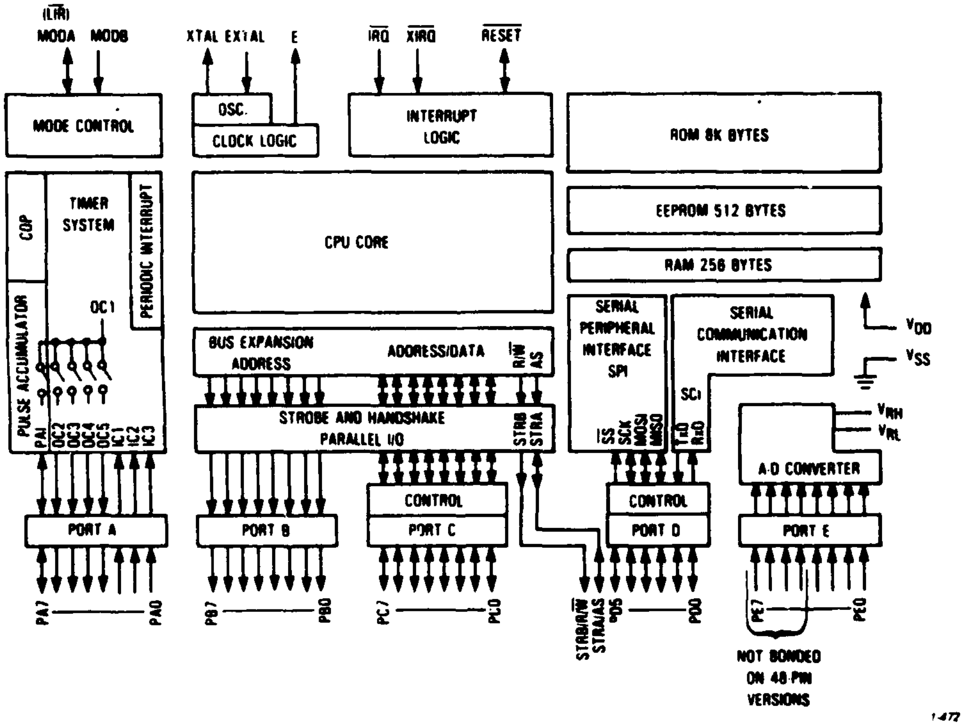


Figure 1-1. Block Diagram

SECTION 2
CPU REGISTERS, FUNCTIONAL PIN DESCRIPTION, OPERATING MODES, INPUT/OUTPUT PROGRAMMING, AND MEMORY

This section provides a description of the CPU registers, functional pins, input/output programming, and memory.

2.1 CPU REGISTERS

In addition to being able to execute all M6800 and M6801 instructions, the MC68HC11A8 uses a 4-page opcode map to allow execution of 91 new opcodes. Seven registers, discussed in the following paragraphs, are available to programmers as shown in Figure 2-1. Figure 2-2 gives the interrupt stacking order.

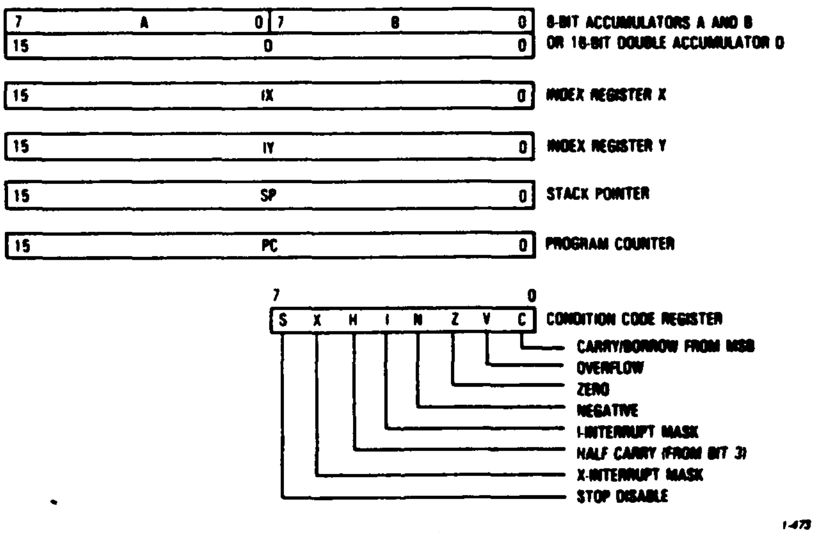


Figure 2-1. Programming Model

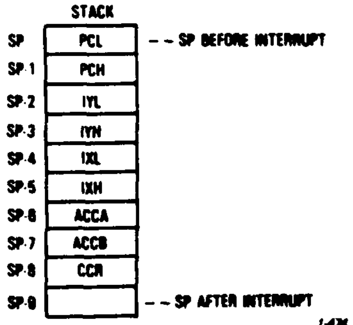


Figure 2-2. Interrupt Stacking Order

2.1.1 Accumulators A and B

Accumulator A and accumulator B are general-purpose 8-bit registers used to hold operands and results of arithmetic calculations or data manipulations. These two accumulators can be concatenated into a single 16-bit accumulator called the D accumulator.

2.1.2 Index Register X (IX)

The 16-bit IX register is used for indexed mode addressing. It provides a 16-bit indexing value which is added to an 8-bit offset provided in an instruction to create an effective address. The IX register can also be used as a counter or as a temporary storage register.

2.1.3 Index Register Y (IY)

The 16-bit IY register is also used for indexed mode addressing similar to the IX register; however, all instructions using the IY register require an extra byte of machine code and an extra cycle of execution time since they are two byte opcodes.

2.1.4 Stack Pointer (SP)

The stack pointer (SP) is a 16-bit register that contains the address of the next free location on the stack. The stack is configured as a sequence of last-in-first-out read/write registers which allow important data to be stored during interrupts and subroutine calls. Each time a new byte is added to the stack (a push), the SP is decremented; whereas, each time a byte is removed from the stack (a pull) the SP is incremented.

2.1.5 Program Counter (PC)

The program counter is a 16-bit register that contains the address of the next instruction to be executed.

2.1.6 Condition Code Register (CCR)

The condition code register is an 8-bit register in which each bit signifies the results of the instruction just executed. These bits can be individually tested by a program and a specific action can be taken as a result of the test. Each individual condition code register bit is explained below.

2.1.6.1 CARRY/BORROW (C). The C bit is set if there was a carry or borrow out of the arithmetic logic unit (ALU) during the last arithmetic operation. The C bit is also affected during shift and rotate instructions.

2.1.6.2 OVERFLOW (V). The overflow bit is set if there was an arithmetic overflow as a result of the operation; otherwise, the V bit is cleared.

2.1.6.3 ZERO (Z). The zero bit is set if the result of the last arithmetic, logic, or data manipulation operation was zero; otherwise, the Z bit is cleared.

2.1.6.4 NEGATIVE (N). The negative bit is set if the result of the last arithmetic, logic, or data manipulation operation was negative; otherwise, the N bit is cleared.

2.1.6.5 I INTERRUPT MASK (I). The interrupt mask bit is set either by hardware or program instruction to disable (mask) all maskable interrupt sources (both external and internal).

2.1.6.6 HALF CARRY (H). The half carry bit is set to a logic one when a carry occurs between bits 3 and 4 of the arithmetic logic unit during an ADD, ABA, or ADC instruction; otherwise, the H bit is cleared.

2.1.6.7 X INTERRUPT MASK (X). The X interrupt mask bit is set only by hardware ($\overline{\text{RESET}}$ or $\overline{\text{XIRQ}}$ acknowledge); and it is cleared only by program instruction (TAP or RTI).

2.1.6.8 STOP DISABLE (S). The stop disable bit is set to disable the STOP instruction, and cleared to enable the STOP instruction. The S bit is program controlled. The STOP instruction is treated as no operation (NOP) if the S bit is set.

2.2 FUNCTIONAL PIN DESCRIPTION

The following paragraphs describe all of the function pins except for the ports which are discussed separately under **2.3 OPERATING MODES AND INPUT/OUTPUT PROGRAMMING.**

2.2.1 V_{DD} AND V_{SS}

Power is supplied to the MC68HC11A8 using these two pins. V_{DD} is the power input (+ 5 volts) and V_{SS} is ground.

2.2.2 $\overline{\text{RESET}}$

This active low bidirectional control pin is used as an input to initialize the MC68HC11A8 to a known startup state, and as an open-drain output to indicate an internal failure has been detected in either the clock monitor or computer operating properly (COP) circuit. Refer to Figure 11-19 for a typical reset circuit.

2.2.3 XTAL and EXTAL

These two pins provide for an interface with either a crystal or a CMOS compatible clock to control the MC68HC11A8 internal clock generator circuitry. The frequency applied to these pins should be four times the desired internal clock rate. The XTAL output is only intended to drive the crystal. It should not be used to drive external circuitry. The XTAL pin must be left unconnected when using an external CMOS compatible clock on EXTAL. Refer to Figure 11-18 for a diagram of the oscillator circuits.

2.2.4 E (ENABLE) CLOCK

The E pin provides an output for the internally generated E clock which can be used as a timing reference. The frequency of the E output is actually one fourth that of the input frequency at the XTAL and EXTAL pins. In general when the E pin is low, an internal process is taking place and, when high, data is being accessed. The E signal is halted when the MCU is in a STOP state.

2.2.5 $\overline{\text{IRQ}}$

The $\overline{\text{IRQ}}$ pin provides a means for requesting asynchronous interrupts to the MC68HC11A8. It is program selectable (OPTION register) with a choice of either negative edge-sensitive or level-sensitive triggering, and is always configured to level-sensitive triggering during reset. The $\overline{\text{IRQ}}$ pin requires an external resistor to V_{DD}.

2.2.6 $\overline{\text{XIRQ}}$

The $\overline{\text{XIRQ}}$ pin provides a means of requesting asynchronous non-maskable interrupts to the MC68HC11A8, after a power-on reset. During reset, the X bit in the condition code register is set and the $\overline{\text{XIRQ}}$ interrupt is masked to preclude interrupts on this line until MCU operation is stabilized. The $\overline{\text{XIRQ}}$ is a level sensitive pin and requires an external resistor to V_{DD}.

2.2.7 MODA/$\overline{\text{LIR}}$ AND MODB

During reset, these two pins are used to control the two basic operating modes of the MC68HC11A8 plus two special operating modes (Table 2-1). Refer to **2.3 OPERATING MODES AND INPUT/OUTPUT PROGRAMMING** for more detailed information.

Table 2-1. Operating Modes versus MODA and MODB

MODB	MODA	Mode Selected
1	0	Single-Chip (Mode 0)
1	1	Expanded Multiplexed (Mode 1)
#	0	Special Bootstrap
#	1	Special Test

NOTE:
1 = Logic High
0 = Logic Low
= 1.8 Times V_{DD} (or Higher)

1-475

In addition to the MODA function, the MODA/$\overline{\text{LIR}}$ pin provides an output as an aid in debugging once reset is completed. The $\overline{\text{LIR}}$ pin goes to an active low during the first E clock cycle of each instruction and remains low for the duration of that cycle (opcode fetch).

2.2.8 V_{RL} and V_{RH}

These two pins provide the reference voltage for the analog-to-digital converter.

2.2.9 R/$\overline{\text{W}}$/STRB

This pin provides two different functions depending on the operating mode.

In the single-chip mode, the STRB pin acts as a programmable strobe for handshake to a parallel I/O device.

In the expanded multiplexed mode, R/$\overline{\text{W}}$ (read/write) is used to control the direction of transfers on the external data bus. A low level (write) on the R/$\overline{\text{W}}$ pin enables the data bus output drivers to the external data bus. A high level (read) on this pin forces the output drivers to a high-impedance state and data is read from the external bus. R/$\overline{\text{W}}$ will stay low during consecutive data bus write cycles, such as in a double-byte store.

2.2.10 AS/STRA

This pin provides two different functions depending on the operating mode.

In the single-chip mode, the STRA pin acts as a programmable input strobe, which can be used with STRB and port C for full handshake modes of parallel I/O.

In the expanded multiplexed mode, the AS (address strobe) output may be used to demultiplex the address and data signals at port C.

2.3 OPERATING MODES AND INPUT/OUTPUT PROGRAMMING

There are five 8-bit ports on the MC68HC11A8 MCU. Three of these ports serve more than one purpose, depending on the mode configuration of the MCU. A summary of the pins versus function and mode is provided in Table 2-2 and discussed in the following paragraphs. Because some of the port functions are controlled by the particular mode selected, each port is discussed for its function(s) during the mode of operation. Unused port input or I/O pins should be tied high or low.

Table 2-2. Port Signal Summary

Port-Bit	Single-Chip Modes 0 and Bootstrap Mode	Expanded Multiplexed Mode 1 and Special Test Mode
A-0	PA0/IC3	PA0/IC3
A-1	PA1/IC2	PA1/IC2
A-2	PA2/IC1	PA2/IC1
A-3	PA3/OC5/and-or OC1	PA3/OC5/and-or OC1
A-4	PA4/OC4/and-or OC1	PA4/OC4/and-or OC1
A-5	PA5/OC3/and-or OC1	PA5/OC3/and-or OC1
A-6	PA6/OC2/and-or OC1	PA6/OC2/and-or O
A-7	PA7/PAI/OC1	PA7/PAI/OC1
B-0	PB0	A8
B-1	PB1	A9
B-2	PB2	A10
B-3	PB3	A11
B-4	PB4	A12
B-5	PB5	A13
B-6	PB6	A14
B-7	PB7	A15
C-0	PC0	A0/D0
C-1	PC1	A1/D1
C-2	PC2	A2/D2
C-3	PC3	A3/D3
C-4	PC4	A4/D4
C-5	PC5	A5/D5
C-6	PC6	A6/D6
C-7	PC7	A7/D7
D-0	PD0/RxD	PD0/RxD
D-1	PD1/TxD	PD1/TxD
D-2	PD2/MISO	PD2/MISO
D-3	PD3/MOSI	PD3/MOSI
D-4	PD4/SCK	PD4/SCK
D-5	PD5/SS	PD5/$\overline{SS}$
D-6	STRA	AS
D-7	STRB	R/$\overline{W}$
E-0	PE0/AN0	PE0/AN0
E-1	PE1/AN1	PE1/AN1
E-2	PE2/AN2	PE2/AN2
E-3	PE3/AN3	PE3/AN3
E-4	PE4/AN4 ##	PE4/AN4 ##
E-5	PE5/AN5 ##	PE5/AN5 ##
E-6	PE6/AN6 ##	PE6/AN6 ##
E-7	PE7/AN7 ##	PE7/AN7 ##

- not bonded in 48-pin variations

1-47

2.3.1 Single-Chip Mode

In the single-chip mode, the MC68HC11A8 functions as a monolithic microcomputer without external address or data buses.

2.3.1.1 PORT A. In all operating modes port A may be configured for: three input capture functions (IC1, IC2, IC3), four output compare functions (OC2, OC3, OC4, OC5), and a pulse accumulator input (PAI) or a fifth output compare function (OC1). Refer to **8.1 PROGRAMMABLE TIMER** *for additional information.*

Each port A pin that is not used for its alternate timer function may be used as a general-purpose input or output line.

2.3.1.2 PORT B. All of the port B pins are general-purpose output pins. During MCU reads of this port, the level sensed at the input side of the port B output drivers is read. Port B may also be used in a simple strobed output mode where the STRB pulses each time port B is written.

2.3.1.3 PORT C. All port C pins are general-purpose input/output pins. Port C inputs can be latched by the STRA input. Port C may also be used in full handshake modes of parallel I/O where the STRA input and STRB output act as handshake control lines.

2.3.1.4 PORT D. Port D bits 0-5 may be used for general I/O or with the serial communications interface (SCI) and serial peripheral interface (SPI) subsystems. Bits 6 and 7 are used as handshake control signals for ports B and C.

Bit 0 is the receive data input (RxD) for the serial communication interface (SCI).

Bit 1 is the transmit data output (TxD) for the SCI.

Bits 2 through 5 are dedicated to the serial peripheral interface (SPI). Bit 2 is the master-in-slave-out (MISO) line. Bit 3 is the master-out-of-slave-in (MOSI) line. Bit 4 is the serial clock (SCK) and bit 5 is the slave select ($\overline{SS}$) input.

Bit 6 is STRA.

Bit 7 is STRB.

2.3.1.5 PORT E. In all operating modes, port E is used for general-purpose inputs and/or analog-to-digital (A/D) channel inputs. Port E should not be read while an A/D conversion is actually taking place.

NOTE

On 48-pin packaged versions of the MC68HC11A8, the four most significant bits of port E are not connected to pins.

2.3.2 Expanded Multiplexed Mode

In the expanded multiplexed mode, the MC68HC11A8 has the capability of accessing a 64K byte address space. The total address space includes the same on-chip memory address as for single-chip mode plus external peripheral and memory devices.

2.3.2.1 PORT A. This port has the same functions as in the single-chip mode (refer to 2.3.1.1 PORT A).

2.3.2.2 PORT B. All of the port B pins act as high order address output pins. During each MCU cycle, bits 8 through 15 of the address are output on the PB0-PB7 lines respectively.

2.3.2.3 PORT C. All port C pins are configured as multiplexed address/data pins. During the address portion of each MCU cycle, bits 0 through 7 of the address are output on the PC0-PC7 lines. During the data portion of each MCU cycle (E high), bits 0 through 7 (D0-D7) are bidirectional data pins controlled by the R/$\overline{W}$ signal.

2.3.2.4 PORT D. This port functions the same way as in the single-chip mode (refer to **2.3.1.4 PORT D**) except bits 6 and 7 which act as expansion bus control lines AS and R/$\overline{W}$ respectively.

2.3.2.5 PORT E. This port has the same function as in the single-chip mode (refer to **2.3.1.5 PORT E**).

2.3.3 Bootstrap Mode

The bootstrap mode is considered a special mode as distinguished from the normal operating single-chip mode. This is a very versatile mode since there are essentially no limitations on the special purpose program that is boot loaded into the internal RAM. The boot loader is contained in 192 bytes of ROM which is enabled as internal memory space at $BF40-$BFFF. The boot loader contains a small program which reads a 256 byte program into on-chip RAM ($0000-$00FF) via the SCI. After the character for address $00FF is received, control is automatically passed to that program at memory address $0000 and the MCU starts operating.

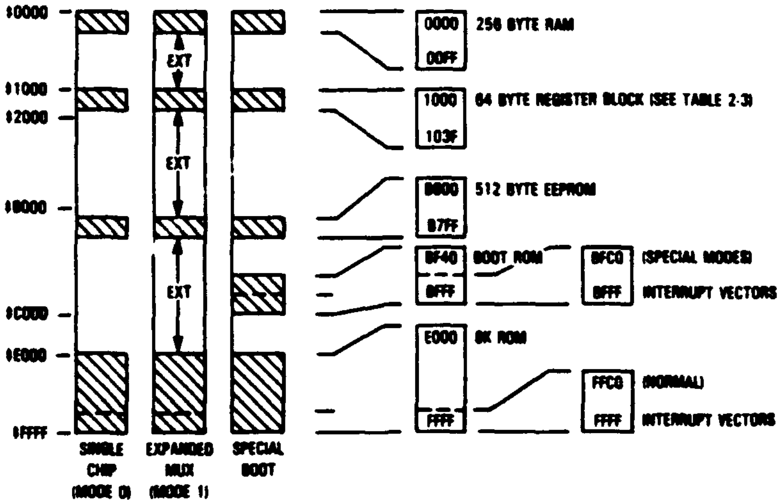
In the bootstrap mode, the serial receive logic is initialized by software in the boot loader ROM to be 1200 baud for a 8.0 MHz crystal or 600 baud for a 4.0 MHz crystal and a data format of one start bit, 8-bit data, and one stop bit. An opening character should be $FF. The character following that will be placed at $0000 and each subsequent character is put in the next higher address until the entire 256 bytes are filled. Note that the entire 256-byte space must be filled.

2.3.4 Test Mode

The test mode is used for factory testing.

2.4 MEMORY

Composite memory maps for each MC68HC11A8 mode of operation are shown in Figure 2-3. These modes include single-chip, expanded multiplexed, and special boot.



NOTES:

1. Either or both the internal RAM and registers can be remapped to any 4K boundary by software.
2. Either or both the ROM and EEPROM can be disabled using a control register (CONFIG) which is implemented with EEPROM cells.

Figure 2-3. Memory Maps

1-477

In the single-chip mode (mode 0) of Figure 2-3, the MC68HC11A8 does not generate external addresses. The actual internal memory locations are shown in the shaded areas of Figure 2-3 and the contents of these shaded areas are shown on the right side of the diagram. Refer to Table 2-3 found on a foldout page at the back of this document for a full list of the registers.

The expanded multiplexed mode (mode 1) memory locations shown in Figure 2-3 are basically the same as for the single-chip mode; however, the memory locations between the shaded areas (designated EXT) are for externally addressed memory and I/O.

The special bootstrap mode memory locations are similar to the single-chip memory locations except that a special bootstrap program is addressed at memory locations $BF40 through $BFFF.

2.4.1 ROM

The internal 8K ROM occupies the highest 8K of the memory map ($E000-$FFFF). This ROM can be disabled when the ROMON bit in the CONFIG register is clear. This register bit is implemented with an EEPROM cell and should be programmed using the same procedures for programming the on-chip EEPROM.

2.4.2 EEPROM

The MC68HC11A8 includes 512 bytes of EEPROM located in the area $B600 through $B7FF which has the same read cycle time as the internal ROM. The write (or programming) mechanism for the EEPROM is controlled by the PPROG register. The 512-byte EEPROM is disabled when the EEON bit in the CONFIG register is clear. This register bit is implemented with an EEPROM cell.

2.4.3 Programming/Erasing Internal EEPROM

The EEPROM programming and erasure process is controlled by the PPROG register. The operating modes for the 512-byte EEPROM are as follows:

NORMAL READ – In this mode, the ERASE bit in the PPROG register must be clear (not erase mode) and the EELAT bit must be clear (not programming mode). While these two bits are cleared, the ROW and EEPGM bits in the PPROG register have no meaning or effect, and the 512-byte EEPROM may be read as if it were a normal ROM.

PROGRAMMING – During EEPROM programming, the ROW bit is not used. If the E clock frequency is less than 1 MHz the CSEL bit in the OPTION register must be set. The normal sequence of events in programming the EEPROM is as follows:

1) Write xxxx x010 to the PPROG register. This specifies program normal mode (ERASE bit = 0), address/data buses configured to latch address and data information (EELAT bit = 1), and erase voltage turned off (EEPGM bit = 0).

2) Write data to be programmed to the desired EEPROM address. This write causes the address and data to be latched in a parallel internal latch.

3) Write EEPGM bit to one (xxxx x011). This couples the EEPROM programming supply voltage to the EEPROM array, to program the specified data into the specified address in EEPROM.

4) Delay for 10 milliseconds.

5) Write xxxx x0 0 to the PPROG register to turn off the programming voltage.

6) Repeat steps 2) through 5) until all desired locations have been programmed.

7) Write EELAT bit back to zero to allow the programmed data to be verified.

ERASE — If the E clock frequency is less than 1 MHz, the CSEL bit in the OPTION register must be set when erasing the EEPROM. The EEPROM has three erase modes:

1) full, 512-byte simultaneous "bulk" erase,

2) "row" erase where only one row (16 bytes) is erased at a time, and

3) "byte" erase where a single specified byte is erased.

NOTE

The erased state of all EEPROM cells is logic one. On early parts, byte and row erase are not implemented.

The normal procedure for erasure of the entire EEPROM is:

1) Write xxxx 0110 to the PPROG register. This specifies the "all" erase mode (ROW bit = 0), erase mode (ERASE bit = 1), EEPROM configured for address/data latching (EELAT bit = 1), and erase voltage turned off (EEPGM bit = 0).

1a) A write must be done to any EEPROM address after Step. 1.

1b) Optionally if the CONFIG register is also to be erased, a write to the address of the CONFIG register must be performed after "bulk" erase was specified by the write, in step 1 above, and before the programming voltage is turned on in step 2 below.

In the case of erasure, the data involved in this write operation is unimportant and the write is needed only for the addressing information it provides.

2) Write xxxx 0111 to the PPROG register to turn on the erase voltage to the EEPROM array.

3) Wait for 10 milliseconds to allow the erasure to complete.

4) Write xxxx 0110 to the PPROG register to turn off the erase voltage.

5) Write xxxx 0000 to the PPROG register to return the EEPROM to the normal read configuration.

The normal procedure for erasure of a row of EEPROM is:

1) Write xxxx 1110 to the PPROG register. This specifies the "row" erase mode (ROW bit = 1), erase mode (ERASE bit = 1), address/data buses configured to latch row address information (EELAT bit = 1), and erase voltage turned off (EEPGM bit = 0).

2) Write to an address in the EEPROM row to be erased (each row is 16 bytes). This latches the row addressing information for the row to be erased.

3) Write xxxx 1111 to the PPROG register to turn on the erase voltage to the EEPROM array.

4) Wait for 10 milliseconds to allow the erasure to complete.

5) Write xxxx 1110 to the PPROG register to turn off the erase voltage.

6) Write xxxx 0000 to the PPROG register to return the EEPROM to the normal read configuration.

The normal procedure for erasure of a single byte of EEPROM is:

1) Write xxx1 x110 to the PPROG register. This specifies the byte erase mode (BYTE = 1; ROW = x), erase mode (ERASE bit = 1), address/data buses configured to latch address information (EELAT bit = 1), and erase voltage turned off (EEPGM bit = 0).

2) Write to the address in the EEPROM to be erased (data is ignored). This latches the address of the byte to be erased.

3) Write xxx1 x111 to the PPROG ·egister. This turns on the erase voltage to the EEPROM array. EEPGM was not changed to one in the same write operation as the write that configured ROW, ERASE, and EELAT because of the possibility of enabling the erase voltage before the erase mode specification was stable.

4) Wait for 10 milliseconds to allow the erasure to complete.

5) Write xxx1 x110 to the PPROG ·egister to turn off the erase voltage.

6) Write xxx0 0000 to the PPROG register to return the EEPROM to the normal read configuration.

2.4.4 PPROG Register (EEPROM Programming Control)

This 8-bit register (see Figure 2-4) is used to control programming and erasure of the 512-byte internal EEPROM. Reset clears this register to $00 so EEPROM is configured for normal reads.

B7	B6	B5	B4	B3	B2	B1	B0	
ODD	EVEN	–	BYTE	ROW	ERASE	EELAT	EEPGM	$__03B

1-470

Figure 2-4. EEPROM Programming Control Register (PPROG)

Bit 7, ODD   Used to Program Odd Rows (TEST)

Bit 6, EVEN   -Used to Program Even Rows (TEST)

Bit 5 _   Not Implemented

Bit 4, BYTE   Used for Erasing Bytes—Overrides Bit 3
0 = Row or Bulk Erase
1 = Erase Only One Byte

Bit 3, ROW   Used for Row Erasing
0 = Bulk Erase
1 = Row Erase

Bit 2, ERASE Enables the Erase
0 = Normal Read or Program
1 = Erase Mode

Bit 1, EELAT EEPROM Latch Control
0 = EEPROM Address and Data Configured for Read
1 = EEPROM Address and Data Configured for Programming

Bit 0, EEPGM Program Command
0 = Switched Off
1 = Turned On

2.4.5 RAM

The 256 byte internal RAM may be positioned in the memory map during initialization by writing to the INIT control register. The reset default position is $0000 through $00FF. RAM is implemented with static cells and retains its contents during the WAIT and STOP modes.

2.4.6 Internal Registers

There are 64 internal registers which are used to control the operation of the MC68HC11A8. These registers can be remapped in the memory space on 4K boundaries using the INIT register. Refer to Table 2-3 (found on a foldout page at the back of this document) for a complete list of the registers. Most of the registers are explained throughout the text.

2.4.7 INIT Register (RAM and I/O Mapping)

This special purpose 8-bit register (see Figure 2-5) is used (optionally) during initialization to change the default locations of RAM and internal registers in the MCU memory map. It may be written to only once within the initial 64 E cycles after a reset and thereafter becomes a read-only register.

B7	B6	B5	B4	B3	B2	B1	B0	
RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0	$__030

Figure 2-5. RAM and I/O Mapping Register (INIT)

The default starting address for internal RAM is $0000 and the default starting address of the 64 byte internal register space is $1000 (the INIT register is initialized to $01 by reset). The upper four bits of the INIT register specify the starting address for the internal 256 byte RAM and the lower four bits of INIT specify the starting address for the 64 byte internal register space. The four bits reflect the upper nibble of the 16-bit address.

Note that if the RAM is repositioned to $E000 or $F000 so that it conflicts with the internal ROM (no conflict if in "No ROM" mode), then the RAM takes higher priority and the conflicting ROM becomes inaccessible. Also, if the 64-byte internal register space is repositioned so that it conflicts with the RAM and/or ROM, then the register space takes highest priority and the RAM and/or ROM become inaccessible.

SECTION 3
RESETS, INTERRUPTS, AND LOW POWER MODES

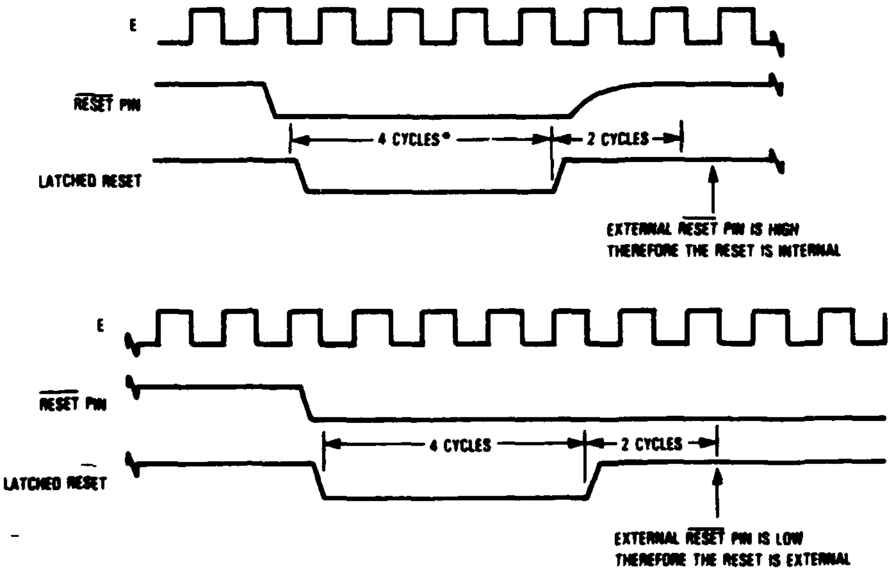
This section provides a description of the resets, interrupts, and low power modes for the MC68HC11A8.

3.1 RESETS

The MC68HC11A8 has four possible types of reset: an active low external reset pin ($\overline{\text{RESET}}$), a power-on reset function, a computer operating properly (COP) watchdog timer reset, and a clock monitor reset.

3.1.1 $\overline{\text{RESET}}$ Pin

The $\overline{\text{RESET}}$ pin is used to reset the MCU to provide an orderly software startup procedure. When the $\overline{\text{RESET}}$ pin goes low, it is held low by an internal device for four E cycles, then released, and two E cycles later it is sampled. If the pin is low, it means that an external reset has occurred. If the pin is high, it means that the reset was initiated internally by the watchdog timer (COP) or the clock monitor (refer to Figure 3-1).


E
$\overline{\text{RESET}}$ PIN
4 CYCLES*
2 CYCLES
LATCHED RESET
EXTERNAL $\overline{\text{RESET}}$ PIN IS HIGH
THEREFORE THE RESET IS INTERNAL
E
$\overline{\text{RESET}}$ PIN
4 CYCLES
2 CYCLES
LATCHED $\overline{\text{RESET}}$
EXTERNAL $\overline{\text{RESET}}$ PIN IS LOW
THEREFORE THE RESET IS EXTERNAL

*The $\overline{\text{RESET}}$ pin will never be low for less than four cycles because an internal device will hold it low even if it is only driven for a short time.

Figure 3-1. Reset Timing

3.1.2 Power-On Reset

The power-on reset occurs when a positive transition is detected on V_{DD}. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drops in power supply voltage. The power-on circuitry provides for a 4064 cycle time delay from the time of the first oscillator operation. In a system where E = 2 MHz, POR lasts about 2 milliseconds. If the system power supply rise time is more than 2 milliseconds, an external reset circuit should be used. If the external $\overline{\text{RESET}}$ pin is low at the end of the power-on delay time, the processor remains in the reset condition until the $\overline{\text{RESET}}$ pin goes high.

CPU

After reset, the CPU fetches the restart vector from $FFFE and $FFFF ($BFFE and $BFFF if in special bootstrap mode) during the first three cycles after reset, and begins executing instructions. The stack pointer and other CPU registers are indeterminate immediately after reset; however, the X and I interrupt mask bits in the condition code register are set so interrupt requests are masked. Also, the S bit in the condition code register is set so that the stop mode is disabled.

Memory Map

After reset, the INIT register is initialized to $01, putting the 256 bytes of RAM at $0000 and the internal registers at $1000. The 8K-byte ROM and/or the 512-byte EEPROM may or may not be present in the memory map because the two bits that enable them in the CONFIG register are EEPROM cells and are not affected by reset or power down.

Parallel I/O

When reset occurs in expanded multiplexed mode, the 18 pins used by the parallel I/O functions are dedicated to the expansion bus. If reset in single-chip mode, the STAF, STAI, and HNDS bits in the parallel input/output control (PIOC) register are initialized to zeros so that no interrupt is pending or enabled, and the simple strobed mode (rather than full handshake mode) of parallel I/O is selected. The CWOM bit in the PIOC is initialized to zero (port C not in wired-OR mode). Port C is initialized as an input port (DDRC = $00), port B is a general purpose output port with all bits initialized to logic zeros. Port D bit 6 is the STRA edge-sensitive strobe input and the active edge is initially configured to detect rising edges (EGA bit in the PIOC set to one by reset), and port D bit 7 is the STRB strobe output and is initially a logic zero (the INVB bit in the PIOC is initialized to logic one). Port C, port D bits 0 through 5, port A bits 0, 1, 2, and 7, and port E are configured as general purpose high-impedance inputs. Port B and bits 3 through 6 of port A have their directions fixed as outputs, when used as general purpose I/O pins, and their reset state is a logic zero.

Timer

During reset, the timer system is initialized to a count of $0000. The prescaler bits are set to 0:0, and all output compare registers are initialized to $FFFF. All input capture registers are indeterminate after reset. The output compare 1 mask (OC1M) register is cleared so that successful OC1 compares do not affect any I/O pins. The other four output compares are configured so as not to affect any I/O pins on successful compares. All three input capture edge-detector circuits are configured for "capture disabled" operation. The timer overflow interrupt flag and all eight timer function interrupt flags are cleared and all nine timer interrupts are disabled since their mask bits are cleared.

Real Time Interrupt

The real time interrupt flag is cleared and automatic hardware interrupts are masked. The rate control bits are cleared after reset and may be initialized by software before the real time interrupt system is used.

Pulse Accumulator

The pulse accumulator system is disabled at reset so that the PAI input pin defaults to being a general purpose input pin.

COP The COP watchdog system is enabled if the NOCOP control bit in the system configuration control register (EEPROM cell) is clear, and disabled if NOCOP is set. The COP rate is set for the shortest duration timeout.

SCI Serial I/O The reset condition of the SCI system is independent of the operating mode. At reset, the SCI baud rate is indeterminate and must be established by a software write to the BAUD register. All transmit and receive interrupts are masked and both the transmitter and receiver are disabled so the port pins default to being general purpose I/O lines. The SCI frame format is initialized to 8-bit word size. The send break and receiver wake up functions are disabled. The TDRE and TC status bits in the SCI status register are both set, indicating that there is no transmit data in either the transmit data register or the transmit serial shift register. The RDRF, IDLE, OR, NF, and FE receive-related status bits are all cleared.

SPI Serial I/O The SPI system is disabled by reset. The port pins associated with this function default to being general purpose I/O lines.

A to D The A/D system configuration at reset is indeterminate.

System The EEPROM programming controls are all disabled so the memory system is configured for normal read operation. The highest priority I interrupt defaults to being the external $\overline{\text{IRQ}}$ pin by PSEL3-PSEL0 equal to 0:1:0:1. The $\overline{\text{IRQ}}$ interrupt pin is configured for level sensitive operation (for wire-OR systems). The RBOOT, SMOD, and MDA bits in the HPRIO register reflect the status of the MODB and MODA inputs at the rising edge of reset. The DLY control bit is set to specify that an oscillator start-up delay is imposed upon recovery from STOP mode. The clock monitor system is disabled by CME equal zero.

3.1.3 Computer Operating Properly (COP) Reset

The watchdog timer, if not reset within a specific time by a COP reset sequence, will generate an MCU reset and drive the $\overline{\text{RESET}}$ pin low to reset the external system.

3.1.4 Clock Monitor Reset

The clock monitor circuit, if enabled, measures the E-clock frequency. If the E-clock signal is lost, or its frequency falls below about 200 kHz, then an MCU reset is generated, and the $\overline{\text{RESET}}$ pin is driven low to reset the external system.

3.2 INTERRUPTS

When an external or internal (hardware) interrupt occurs, the interrupt is not serviced until the current instruction being executed is completed. Until the current instruction is complete, the interrupt is considered pending. After completion of current instruction execution, unmasked interrupts may be serviced in accordance with an established fixed hardware priority circuit; however, one I bit related interrupt source may be dynamically elevated to the highest I bit priority position in the circuit.

Seventeen hardware interrupts and one software interrupt (excluding reset type interrupts) can be generated from all of the possible sources. The interrupts can be divided into two basic categories, maskable and non-maskable. In the MC68HC11A8 fifteen of the interrupts can be masked using the condition code register I bit. In addition to being maskable by the I bit in the condition code register, all of the on-chip interrupt sources are individually maskable by local control bits. The software interrupt (SWI instruction) is a non-maskable instruction rather than a maskable interrupt source. The last interrupt (external input to the $\overline{\text{XIRQ}}$ pin) is considered as a non-maskable interrupt because once enabled, it cannot be masked by software; however, it is masked during reset and upon receipt of an interrupt at the $\overline{\text{XIRQ}}$ pin. Tables 3-1, 3-2, and 3-3 provide a list of each interrupt, its vector location in ROM, and the actual condition code and control bits that mask it. A discussion of the various interrupts is provided below.

Table 3-1. Interrupt Vector Assignments

Vector Address	Interrupt Source	CC Register Mask	Local Mask
FFC0, C1	Reserved	–	–
●	●		
●	●		
FFD4, D5	Reserved	–	–
FFD6, D7	SCI Serial System	I Bit	See Table 3-2
FFD8, D9	SPI Serial Transfer Complete	I Bit	SPIE
FFDA, DB	Pulse Accumulator Input Edge	I Bit	PAII
FFDC, DD	Pulse Accumulator Overflow	I Bit	PAOVI
FFDE, DF	Timer Overflow	I Bit	TOI
FFE0, E1	Timer Output Compare 5	I Bit	OC5I
FFE2, E3	Timer Output Compare 4	I Bit	OC4I
FFE4, E5	Timer Output Compare 3	I Bit	OC3I
FFE6, E7	Timer Output Compare 2	I Bit	OC2I
FFE8, E9	Timer Output Compare 1	I Bit	OC1I
FFEA, EB	Timer Input Capture 3	I Bit	OC3I
FFEC, ED	Timer Input Capture 2	I Bit	OC2I
FFEE, EF	Timer Input Capture 1	I Bit	OC1I
FFF0, F1	Real Time Interrupt	I Bit	RTII
FFF2, F3	$\overline{\text{IRQ}}$ (External Pin or Parallel I/O)	I Bit	See Table 3-3
FFF4, F5	$\overline{\text{XIRQ}}$ Pin (Pseudo Non-Maskable Interrupt)	X Bit	None
FFF6, F7	SWI	None	None
FFF8, F9	Illegal Opcode Trap	None	None
FFFA, FB	COP Failure (Reset)	None	NOCOP
FFFC, FD	COP Clock Monitor Fail (Reset)	None	CME
FFFE, FF	$\overline{\text{RESET}}$	None	None

1-480

Table 3-2. SCI Serial System Interrupts

Interrupt Cause	Local Mask
Receive Data Register Full	RIE
Receiver Overrun	RIE
Idle Line Detect	ILIE
Transmit Data Register Empty	TIE
Transmit Complete	TCIE

1-481

Table 3-3. IRQ Vector Interrupts

Interrupt Cause	Local Mask
External Pin	None
Parallel I/O Handshake	STAI

1-482

3.2.1 Software Interrupt (SWI)

The software interrupt is executed the same as any other instruction and will take precedence over interrupts only if the other interrupts are masked (I and X bits in the condition code register set). The SWI instruction is executed similar to other maskable interrupts in that it sets the I bit, CPU registers are stacked, etc.

> **NOTE**
>
> The SWI instruction cannot be fetched as long as another interrupt is pending execution. However, once it is fetched no other interrupt can be honored until the first instruction in the SWI service routine is completed.

3.2.2 Illegal Opcode Trap

Since not all possible opcodes or opcode sequences are defined, an illegal opcode detection circuit has been included in the MC68HC11A8. When an illegal opcode is detected, an interrupt is requested to the illegal opcode vector.

3.2.3 Interrupt Mask Bits in Condition Code Register

Upon reset, both the X bit and the I bit are set to mask all interrupts. After minimum system initialization, software may clear the X bit by a TAP instruction, thus enabling $\overline{\text{XIRQ}}$ interrupts. Thereafter software cannot set the X bit so an $\overline{\text{XIRQ}}$ interrupt is effectively a non-maskable interrupt. Since the operation of the I bit related interrupt structure has no effect on the X bit, the external $\overline{\text{XIRQ}}$ pin remains effectively non-masked. In the interrupt priority logic, the $\overline{\text{XIRQ}}$ interrupt would be a higher priority than any source that is maskable by the I bit. All I bit related interrupts would operate normally with their own priority relationship. When an I bit related interrupt occurs, the I bit is automatically set by hardware after stacking the condition code register byte, but the X bit is not affected. When an X bit related interrupt occurs, both the X bit and the I bit are automatically set by hardware after stacking the condition code register. An RTI (return from interrupt) instruction restores the X and I bits to their pre-interrupt request state.

3.2.4 Priority Structure

Interrupts in the MC68HC11A8 obey a fixed hardware priority circuit to resolve simultaneous requests; however, one I bit related interrupt source may be elevated to the highest I bit priority position in the resolution circuit. The first six interrupt sources are not masked by the I bit in the condition code register and have the fixed priority interrupt relationship of: reset, clock monitor fail, COP fail, illegal opcode, and $\overline{\text{XIRQ}}$. (SWI is actually an instruction and has highest priority other than reset in the sense that once the SWI opcode is fetched, no other interrupt can be honored until the SWI vector has been fetched). Each of these sources is an input to the priority resolution circuit. The highest I bit masked priority input to the resolution circuit is assigned under software control (of the HPRIO register) to be connected to any one of the remaining I bit related interrupt sources. In order to avoid timing races, the HPRIO register may only be written while the I bit related interrupts are inhibited (I bit in condition code register is a logic one). An interrupt that is assigned to this high priority position is still subject to masking by any associated control bits or the I bit in the condition code register. The interrupt vector address is not affected by assigning a source to this higher priority position.

Figure 3-2 summarizes the priority structure and additional mask conditions that lead to recognition of interrupt requests in the MC68HC11A8.

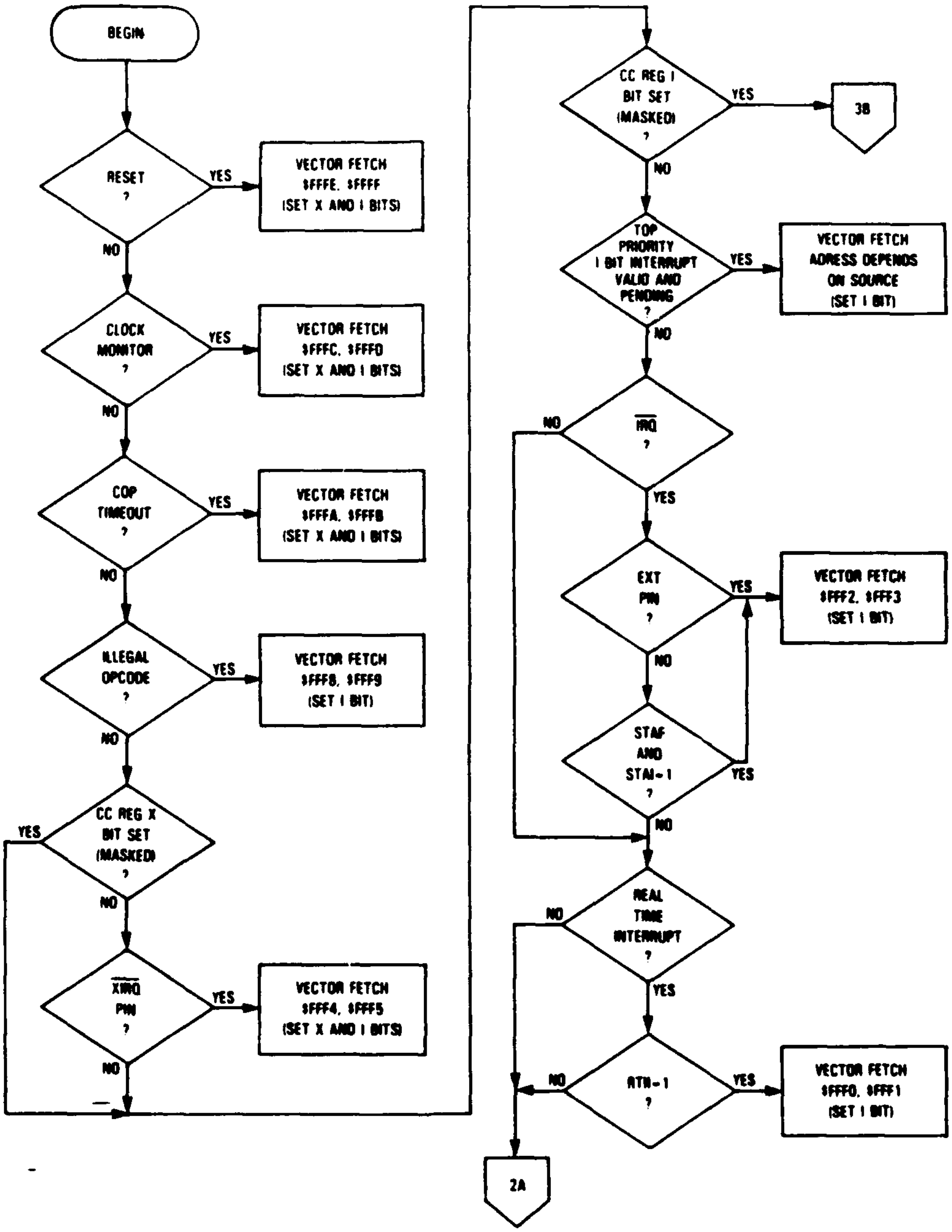

BEGIN
RESET ?
YES
VECTOR FETCH $FFFE, $FFFF (SET X AND I BITS)
NO
CLOCK MONITOR ?
YES
VECTOR FETCH $FFFC, $FFFD (SET X AND I BITS)
NO
COP TIMEOUT ?
YES
VECTOR FETCH $FFFA, $FFFB (SET X AND I BITS)
NO
ILLEGAL OPCODE ?
YES
VECTOR FETCH $FFF8, $FFF9 (SET I BIT)
NO
CC REG X BIT SET (MASKED) ?
YES
NO
XIRQ PIN ?
YES
VECTOR FETCH $FFF4, $FFF5 (SET X AND I BITS)
NO
CC REG I BIT SET (MASKED) ?
YES
3B
NO
TOP PRIORITY I BIT INTERRUPT VALID AND PENDING ?
YES
VECTOR FETCH ADRESS DEPENDS ON SOURCE (SET I BIT)
NO
IRQ ?
NO
YES
EXT PIN ?
YES
VECTOR FETCH $FFF2, $FFF3 (SET I BIT)
NO
STAF AND STAI=1 ?
YES
NO
REAL TIME INTERRUPT ?
NO
YES
RTII=1 ?
NO
YES
VECTOR FETCH $FFF0, $FFF1 (SET I BIT)
2A
1-4834

Figure 3-2. MC68HC11A8 Interrupt Structure Flowchart (Sheet 1 of 3)

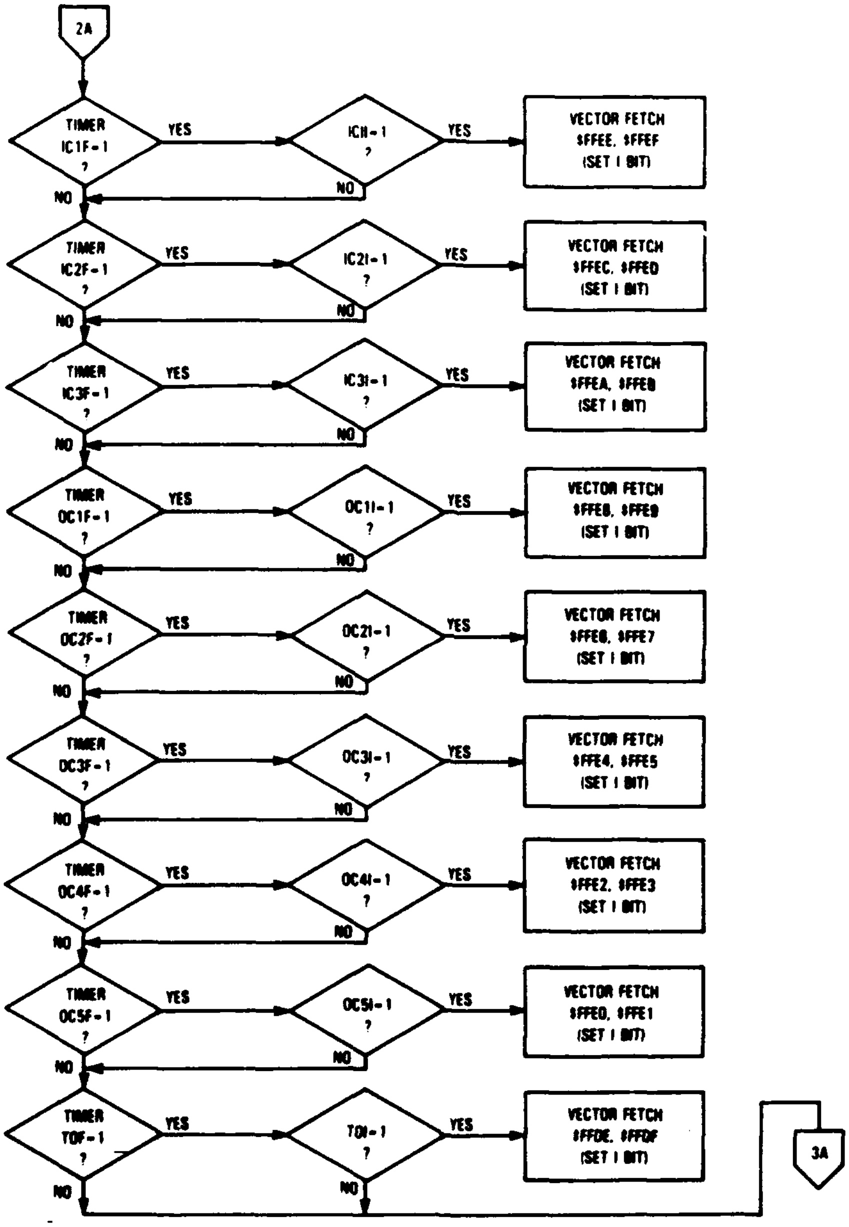
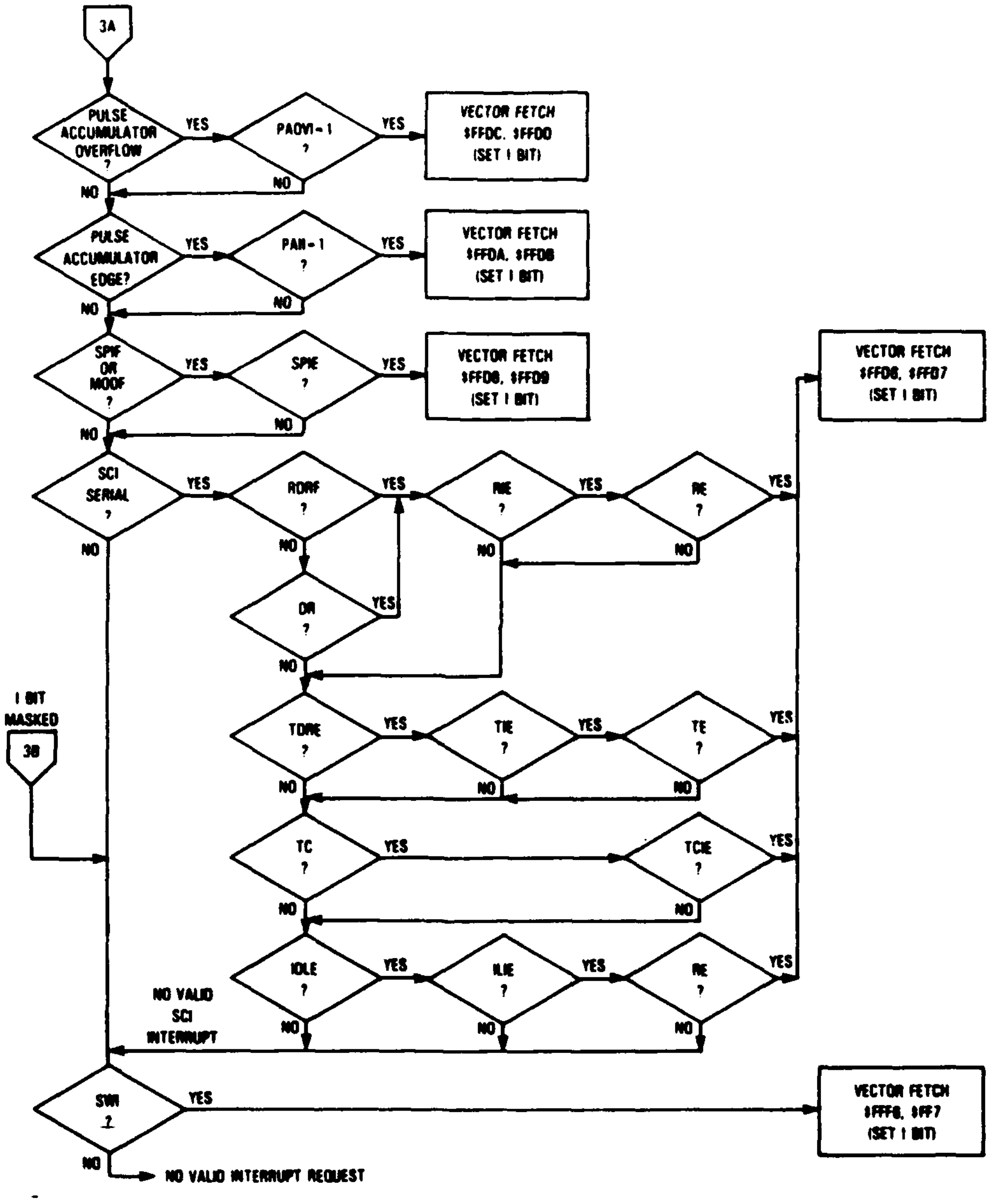

2A
TIMER IC1F = 1 ?
YES
NO
ICII = 1 ?
YES
NO
VECTOR FETCH $FFEE, $FFEF (SET I BIT)
TIMER IC2F = 1 ?
IC2I = 1 ?
VECTOR FETCH $FFEC, $FFED (SET I BIT)
TIMER IC3F = 1 ?
IC3I = 1 ?
VECTOR FETCH $FFEA, $FFEB (SET I BIT)
TIMER OC1F = 1 ?
OC1I = 1 ?
VECTOR FETCH $FFE8, $FFE9 (SET I BIT)
TIMER OC2F = 1 ?
OC2I = 1 ?
VECTOR FETCH $FFE6, $FFE7 (SET I BIT)
TIMER OC3F = 1 ?
OC3I = 1 ?
VECTOR FETCH $FFE4, $FFE5 (SET I BIT)
TIMER OC4F = 1 ?
OC4I = 1 ?
VECTOR FETCH $FFE2, $FFE3 (SET I BIT)
TIMER OC5F = 1 ?
OC5I = 1 ?
VECTOR FETCH $FFE0, $FFE1 (SET I BIT)
TIMER TOF = 1 ?
TOI = 1 ?
VECTOR FETCH $FFDE, $FFDF (SET I BIT)
3A

Figure 3-2. MC68HC11A8 Interrupt Structure Flowchart (Sheet 2 of 3)



/ 4BC

Figure 3-2. MC68HC11A8 Interrupt Structure Flowchart (Sheet 3 of 3)

3.2.5 Highest Priority I Interrupt Register (HPRIO)

This 8-bit register (Figure 3-3) is used to select one of the I bit related interrupt sources to be elevated to the highest I bit masked position in the priority resolution circuit. In addition, four miscellaneous system control bits are included in this register.

B7	B6	B5	B4	B3	B2	B1	B0	
RBOOT	SMOD	MDA	IRV	PSEL3	PSEL2	PSEL1	PSEL0	$_03C

Figure 3-3. Highest Priority I Interrupt Register (HPRIO)

1-484

B7, RBOOT The read bootstrap ROM bit only has meaning when the SMOD bit is a logic one (special bootstrap mode or special test mode). At all other times, this bit reverts to its logic zero disabled state and may not be written.

When set, upon reset in bootstrap mode only, the small bootstrap loader program is enabled. When clear, by reset in the other three modes, this ROM is disabled and accesses to this area are treated as external accesses.

B6, SMOD The special mode write-only bit reflects the status of the MODB input pin at the rising edge of reset. It is set if the MODB pin is at or above 1.8 times V_{DD} volts during reset. Otherwise, it is cleared or under software control from the special modes.

B5, MDA The mode select A bit reflects the status of the MODA input pin at the rising edge of reset. While the SMOD bit is a logic one (special test or special bootstrap mode in effect), the MDA bit may be written, thus, changing the operating mode of the MCU. When the SMOD bit is a logic zero, the MODA bit is a read-only bit and the operating mode cannot be changed without going through a reset sequence.

Table 3-4 summarizes the relationship between the SMOD and MDA bits and the MODB and MODA input pins at the rising edge of reset.

Table 3-4. Mode Bits Relationship

Inputs		Mode Description	Latched at Reset	
MODB	MODA		SMOD	MDA
1	0	Single Chip (Mode 0)	0	0
1	1	Expanded Multiplexed (Mode 1)	0	1
*	0	Special Bootstrap	1	0
*	1	Special Test	1	1

1 = logic high 0 = logic low * = 1.8 times V_{DD} (or higher)

1-485

B4, IRV The internal read visibility bit is used in the special modes (SMOD = 1) to affect visibility of internal reads on the expansion data bus. IRV is writable only if SMOD = 1 and returns to zero of SMOD = 0. If the bit is zero, visibility of internal reads are blocked. If the bit is one, internal reads are visible on the external bus.

BS, PSEL3 These four priority select bits are used to specify one I bit related interrupt source which becomes the highest priority I bit related source (Table 3-5).
B2, PSEL2
B1, PSEL1
B0, PSEL0

Table 3-5. Highest Priority I Interrupt versus PSEL3-PSEL0

PSEL3	PSEL2	PSEL1	PSEL0	Interrupt Source Promoted
0	0	0	0	Timer Overflow
0	0	0	1	Pulse Accumulator Overflow
0	0	1	0	Pulse Accumulator Input Edge
0	0	1	1	SPI Serial Transfer Complete
0	1	0	0	SCI Serial System
0	1	0	1	Reserved (*Default to* $\overline{\text{IRQ}}$)
0	1	1	0	$\overline{\text{IRQ}}$ (External Pin or Parallel I/O)
0	1	1	1	Real Time Interrupt
1	0	0	0	Timer Input Capture 1
1	0	0	1	Timer Input Capture 2
1	0	1	0	Timer Input Capture 3
1	0	1	1	Timer Output Compare 1
1	1	0	0	Timer Output Compare 2
1	1	0	1	Timer Output Compare 3
1	1	1	0	Timer Output Compare 4
1	1	1	1	Timer Output Compare 5

NOTE: During reset, PSEL3, PSEL2, PSEL1, and PSEL0 are initialized to 0:1:0:1 which corresponds to "Reserved (default to $\overline{\text{IRQ}}$)" being the highest priority I bit related interrupt soruce.

1-485

3.3 LOW POWER MODES

The MC68HC11A8 MCU contains two programmable low power operating modes; stop and wait. These two instructions are discussed below.

3.3.1 WAIT Instruction

The WAI instruction puts the MC68HC11A8 in a low-power mode, keeping the oscillator running. Upon execution of WAI, the machine state is stacked and program execution stops. The wait state can be exited only by an unmasked interrupt or $\overline{\text{RESET}}$. If the I bit is set (interrupts masked) and the COP is disabled, the timer system will be turned off to reduce power consumption. The amount of power savings is application dependent and depends upon circuitry connected to the MCU pins as well as which subsystems (i.e., timer, SPI, SCI) are active when the WAIT mode is entered.

3.3.2 STOP Instruction

The STOP instruction places the MC68HC11A8 MCU in its lowest power consumption mode provided the S bit in the condition code register is clear. In the stop mode, all clocks including the internal oscillator are stopped causing all internal processing to be halted. Recovery from STOP may be accomplished by $\overline{\text{RESET}}$, $\overline{\text{XIRQ}}$, or an unmasked $\overline{\text{IRQ}}$. When the $\overline{\text{XIRQ}}$ is used, the MCU exits from the stop mode regardless of the state of the X bit in the condition code register; however, the actual recovery sequence differs depending on the state of the X bit. If the X bit is a logic zero, the MCU starts up with the stacking sequence leading to normal service of the $\overline{\text{XIRQ}}$ request. If the X bit is a logic one, then processing will continue with the instruction immediately following the STOP instruction and no $\overline{\text{XIRQ}}$ interrupt service routine is requested. A $\overline{\text{RESET}}$ will always result in an exit from the stop mode, and the start of MCU operation is determined by the reset vector.

Since the oscillator is stopped in the stop mode, a restart delay of 4064 cycle times may be required to allow for oscillator stabilization when exiting from the stop mode. If the internal oscillator is being used, this delay is required; however, if a stable external oscillator is being used, a control bit in the OPTION register may be used (DLY = 0) to give a delay of four cycles.

SECTION 4
PARALLEL I/O AND SYSTEM CONFIGURATION

4.1 PARALLEL I/O

The MC68HC11A8 includes 40 I/O pins in five 8-bit ports. All of these pins serve multiple functions depending on the operating mode and several internal control registers. This section explains the operation of these pins only when they are used as parallel I/O pins.

Ports C and D may be used as general purpose input and/or output pins, as specified by the data direction registers DDRC and DDRD. Ports A, B, and E, with the exception of port A bit 7, have fixed data direction and do not require a DDR control register. Ports B and C, and bits 6 and 7 of port D, may be used for special strobed and handshake modes of parallel I/O, as well as for general purpose I/O.

4.1.1 General Purpose I/O (Ports C and D)

As general-purpose I/O lines, each bit has an associated bit in a PORTx data register and a bit in the corresponding position in a DDRx register. The DDRx is used to specify the primary direction of data on the I/O pin. When a bit which is configured for output is read, the value returned is the value at the input to the pin driver. When a line is configured as an input, by clearing the DDRx bit, the pin becomes a high impedance input. If a write is executed to a line that is configured as an input, the value does not affect the I/O pin. but the bit is stored in an internal latch so that if the line is later reconfigured as an output, then this value appears at the I/O pin.

Note that bits 6 and 7 of port D are dedicated to bus control (AS and R/$\overline{W}$) while in expanded mode, or parallel I/O strobes (STRA and STRB) while in single chip modes. For this reason, bits 6 and 7 of port D are not available as general purpose I/O lines and the associated bits in the DDRD and PORTD registers are not implemented.

4.1.2 Fixed Direction I/O (Ports A, B, and E)

The pins for ports A, B, and E, except for port A bit 7, have fixed data directions and do not need data direction registers. When port B is being used for general purpose outputs, it is configured for output-only and reads return the levels sensed at the input of the pin drivers. When port A is being used for general purpose I/O, bits 0, 1, and 2 are configured for input-only and writes to these bits have no meaning or effect. Bits 3, 4, 5, and 6 of port A are configured for output-only when used for general purpose I/O, and reads of these bits return the levels sensed at the inputs to the pin drivers. Port A bit 7 (PA7) can be configured as a general-purpose I/O using the DDRA7 bit in the PACTL register. Port E contains the eight A/D channel inputs, but these pins may also be used as general purpose digital inputs. Writes to the PORTE address have no meaning or effect.

4.1.3 Simple Strobed I/O

The simple strobed mode of parallel I/O is invoked and controlled by the PIOC control register. This mode is selected when the HNDS bit in the PIOC control register is clear. It forces port C to be a strobed input port with port D bit 6 as the edge-detecting latch command input (STRA pin). Also, port B becomes a strobed output port with port D bit 7 as the output strobe (STRB pin). The logic sense of the STRB output is selected by the INVB control bit.

4.1.3.1 STROBED INPUT PORT C. In this mode, there are two addresses where port C may be read, PORTC data register and PORTCL latch register. The DDRC register still controls the data direction of all port C pins. Even when the strobed input mode is selected, any or all of the bits in port C may still be used as general purpose I/O lines.

STRA (port D bit 6) is used as an edge-detecting input, and either falling or rising edges may be specified as the significant edge by use of the EGA bit in PIOC. Whenever the selected active edge is detected at the STRA pin, the current logic levels at port C are latched into the PORTCL register and the strobe A flag (STAF) bit in PIOC is set. If the STAI bit in PIOC is also set, an internal interrupt sequence is requested to the IRQ vector. The STAF flag is automatically cleared by reading the PIOC register (with STAF set) followed by a read of the PORTCL register. Data is latched in the PORTCL register whether or not the STAF flag was previously clear.

4.1.3.2 STROBED OUTPUT PORT B. In this mode, port D bit 7 (STRB) is a strobe output which is pulsed for two E periods each time there is a write to port B. The INVB bit in PIOC controls the polarity of the pulse out of the STRB pin.

4.1.4 Full Handshake I/O

The full handshake modes of parallel I/O involve port C and bits 6 and 7 of port D. There are two basic modes (input and output) and an additional variation on the output handshake mode that allows for three-stated operation of port C. In all handshake modes, port D bit 6 (STRA) is an edge-detecting input, and port D bit 7 (STRB) is a handshake output line.

When full input handshake protocol is specified, both general purpose input and/or general purpose output can coexist at port C. When full output handshake protocol is specified, general purpose output can coexist with the handshake outputs at port C, but the three-state feature of the output handshake mode interferes with general purpose input in two ways. First, in full output handshake, the port C pins are forced to be driven outputs whenever STRA is at its active level regardless of the DDRC bits. This potentially conflicts with any device trying to drive port C unless the external device has an open-drain type output driver. Second, the value returned on reads of port C is the state of the outputs of an internal port C output latch regardless of the states of the DDRC bits, so that the data written for output handshake can be read even if the pins are in a three-state condition.

4.1.4.1 INPUT HANDSHAKE PROTOCOL. In the input handshake scheme, port C is a latching input port, port D bit 6 (STRA) is an edge-sensitive latch command from the external system that is driving port C, and port D bit 7 (STRB) is a "READY" output line controlled by logic in the MCU.

When a ready condition is recognized, the external device places data on the port C inputs, then pulses the STRA input to the MC68HC11A8. The active edge on the STRA line latches the port C data into the PORTCL register, sets the STAF flag (optionally causing an interrupt), and deasserts the STRB line. Deassertion of the STRB line automatically inhibits the external device form strobing new data into port C.

Reading the PORTCL latch register (independent of clearing the STAF flag) causes the STRB line to be asserted indicating that new data may now be strobed into port C.

The STRB line can be configured (with the PLS control bit) to be a pulse output (pulse mode) or a static output (interlocked mode).

The port C data direction register bits should be cleared (input) for each bit that is to be used as a latched input bit. However, some port C bits can be used as latched inputs with the input handshake protocol while, at the same time, using some port C bits as static inputs, and some port C bits as static output bits. The input handshake protocol has no effect on the use of port C bits as static inputs or as static outputs. Reads of the PORTC register always return the static logic level at the port C pins (for lines configured as input by DDRC bit = 0). Writes to either the PORTC address or the PORTCL address send information to the port C output register without affecting the input handshake strobes.

4.1.4.2 OUTPUT HANDSHAKE PROTOCOL. In the output handshake scheme, port C is an output port, port D bit 7 (STRB) is a "READY" output, and port D bit 6 (STRA) is an edge-sensitive acknowledge input signal, indicating that port C output data has been accepted by the external device. In a variation of this output handshake operation, port D bit 6 (STRA) is also used as an output enable input, as well as an edge-sensitive acknowledge input.

The MC68HC11A8 places data on the port C output lines and then indicates stable data is available by automatically asserting the STRB line. The external device then processes the available data and pulses the STRA input to indicate that new data may be placed on the port C output lines. The active edge on STRA causes the STRB line to be automatically deasserted and the STAF status flag to be set (optionally causing an interrupt). In response to STAF being set, the program transfers new data out on port C as required. Placing the data in PORTCL asserts the STRB.

There is a variation to the output handshake protocol that allows three-state operation of port C. It is possible to directly interconnect this 8-bit parallel port to other 8-bit three-state devices with no extra external parts.

While the STRA input pin is inactive, all port C bits obey the data direction specified by DDRC so that bits which are configured as inputs are high impedance. When the STRA input is activated, all port C lines are forced to outputs regardless of the data in DDRC. Note that in output handshake mode, reads of port C always return the value sensed at the input to the output buffer regardless of the state of the DDRC bits because the pins would not necessarily have meaningful data on them in the three-state variation of this mode. This operation makes it impossible to use some port C bits as static inputs, while using others as handshake outputs, but does not interfere with the use of some port C bits as static outputs. Port C bits intended as static outputs or normal handshake outputs should have their corresponding DDRC bits set, and bits intended as three-state handshake outputs should have their corresponding DDRC bits clear.

4.1.4.3 PARALLEL I/O CONTROL REGISTER (PIOC)

The parallel handshake I/O functions are available only in the single-chip mode. PIOC is a read/write register except bit 7 which is read only (see Figure 4-1).

B7	B6	B5	B4	B3	B2	B1	B0	
STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB	$_002

1497

Figure 4-1. Parallel I/O Control Register (PIOC)

Bit 7, STAF — Strobe A Interrupt Status Flag. This bit is set when a selected edge of strobe A occurs. Clearing it depends on the state of HNDS and OIN bits (Table 4-1). STAF is cleared by reset.

Table 4-1. STAF Bit Clearing Conditions

HNDS	OIN	Clearing Mechanism
0	X	Reading PIOC (with STAF Set) Followed by a Read of PORTCL
1	0	Reading PIOC (with STAF Set) Followed by a Read of PORTCL
1	1	Reading PIOC (with STAF Set) Followed by a Write to PORTCL

Bit 6, STAI — Strobe A Interrupt Enable Mask. When this bit is set and the I bit in the condition code register is clear, STAF (when set) will request an interrupt. STAI is cleared by reset.

Bit 5, CWOM — Port C Wire-OR Mode. When clear, port C operates normally. When set, port C behaves as open-drain outputs. CWOM is cleared by reset.

Bit 4, HNDS — Handshake Mode. When clear, strobe A acts as a simple input strobe to latch data into PORTCL, and strobe B acts as a simple output strobe which pulses after a write to port B. When set, a handshake protocol involving port C, STRA, and STRB is selected (see the definition for the OIN bit).

Bit 3, OIN — Output or Input Handshaking. This bit has no meaning when HNDS = 0. When clear, input handshake mode is selected. When set, output handshake mode is selected. OIN is cleared by reset.

Bit 2, PLS — Pulse/Interlocked Handshake Operation. This bit has no meaning if HNDS = 0. When clear, interlocked handshake operation is selected. In this mode strobe B, once activated, stays active until the selected edge of strobe A is detected. When set, strobe B is pulsed for two E cycles. This bit is undefined coming out of reset.

Bit 1, EGA — Active Edge for Strobe A. When clear, falling edge of STRA is selected. When output handshake is selected, port C bits obey the DDRC while STRA is low, but port C is forced to output when STRA is high.

When set, rising edge of STRA is selected. When output handshake is selected, port C bits obey the DDRC while STRA is high, but port C is forced to output when STRA is low. This bit is set by reset.

Bit 0, INVB — Invert Strobe B. When clear, the active level on strobe B is a logic zero. When set, the active level on strobe B is a logic one. It is set by reset.

4.2 SYSTEM CONFIGURATION

The MC68HC11A8 allows an end user to configure the MCU system to his specific requirements through the use of hardwired options such as the mode select pins, semi-permanent EEPROM control bit specifications (CONFIG register), or by use of internal software control registers. The CONFIG control register (see Figure 4-2) is implemented in EEPROM cells and controls the presence of ROM and EEPROM in the memory map, as well as the COPON COP watchdog system enable. An optional security feature is available intended to allow user protection of data in MC68HC11A8 EEPROM and RAM.

B7	B6	B5	B4	B3	B2	B1	B0	
–	–	–	–	NOSEC	NOCOP	ROMON	EEON	$__03F

Figure 4-2. System Configuration Control Register (CONFIG)

Bits 7, 6, 5, and 4—Not Implemented	These bits are not implemented. They read as logic zeros.
Bit 3—NOSEC	Security Mode Option Bit. When the security mask option is specified, this bit can be used to enable a software antitheft mechanism. When cleared, this bit forces the MDA mode control bit to zero so that only single-chip modes of operation can be selected. If the bit is cleared when the MCU is reset in the special bootstrap mode, EEPROM and RAM are erased before the boot loading process continues.
Bit 2—NOCOP	COP System OFF. When this bit is clear, the COP watchdog forced reset function is enabled. When this bit is set, the COP watchdog circuit is disabled.
Bit 1—ROMON	Enable On-Chip ROM Select Bit. When this bit is clear, the 8K internal ROM is disabled, and that memory space becomes externally accessed space.
Bit 0—EEON	Enable On-Chip EEPROM Select Bit. When this bit is clear, the 512-byte internal EEROM is disabled, and that memory space becomes externally accessed space.

4.3 PROGRAMMING AND ERASURE OF THE CONFIG REGISTER

Since the CONFIG register is implemented with EEPROM cells, special provisions must be made to erase and program this register. The normal EEPROM control bits in the PPROG register are used for this purpose. The programming/erasure procedures for the CONFIG register are described in **2.4.3 Programming/Erasing Internal EEPROM.**

SECTION 5
SERIAL COMMUNICATIONS INTERFACE (SCI)

This section contains a description of the serial communications interface (SCI).

5.1 OVERVIEW AND FEATURES

A full-duplex asynchronous Serial Communications Interface (SCI) is provided with a standard NRZ format (one start bit, eight or nine data bits, and one stop bit) and a variety of baud rates. The SCI transmitter and receiver are functionally independent, but use the same data format and bit rate. "Baud" and "bit rate" are used synonymously in the following description.

SCI Two-Wire System Features

- Standard NRZ (mark/space) format.
- Advanced error detection method includes noise detection for noise duration of up to 1/16 bit time.
- Full-duplex operation.
- Software programmable for one of 32 different baud rates.
- Software selectable word length (eight or nine bit words).
- Separate transmitter and receiver enable bits.
- Capable of being interrupt driven.
- Four separate enable bits available for interrupt control.

SCI Receiver Features

- Receiver wake-up function (idle or address bit).
- Idle line detect.
- Framing error detect.
- Noise detect.
- Overrun detect.
- Receiver data register full flag.

SCI Transmitter Features

- Transmit data register empty flag.
- Transmit complete flag.
- Send break.

5.2 DATA FORMAT

Receive data (RxD) or transmit data (TxD) is the serial data which is transferred to the internal data bus from the input pin (RxD), and from the internal bus to the output pin (TxD). Data format is as shown for the NRZ in Figure 5-1 and must meet the following criteria:

1) The idle line is in a high (logic one) state prior to transmission/reception of a message.
2) A start bit (logic zero) is transmitted/receiver indicating the start of a message.
3) The data is transmitted and received least-significant-bit first.
4) A stop bit (high in the tenth or eleventh bit position) indicates the byte is complete.
5) A break is defined as the transmission or reception of a low (logic zero) for at least one complete frame time.

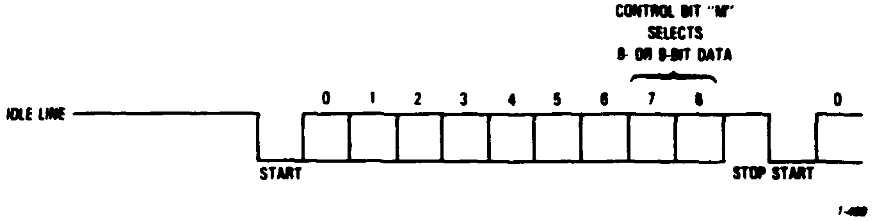


Figure 5-1. Data Format

5.3 WAKE-UP FEATURE

An inactive SCI may be re-enabled by two different methods. In the first method, an SCI receiver is re-enabled by an idle string of at least ten (or eleven) consecutive ones. The second wake-up method allows the user to insert a logic one as the most significant data bit (eighth or ninth bit) of the transmit data word which automatically wakes up all "sleeping" SCIs.

5.4 RECEIVE DATA (RxD)

Receive data (RxD) is the serial data which is presented from the input pin via the SCI to the internal bus. The receiver clocks the input at a rate equal to 16 times the baud rate. This 16 times higher-than-baud rate is referred to as the RT rate.

Once a valid start bit is detected, the start bit, each data bit, and the stop bit are sampled three times at RT intervals 8 RT, 9 RT, and 10 RT (1 RT is the position where the bit is expected to start), as shown in Figure 5-2. The value of the bit is determined by voting logic which takes the value of the majority of samples.

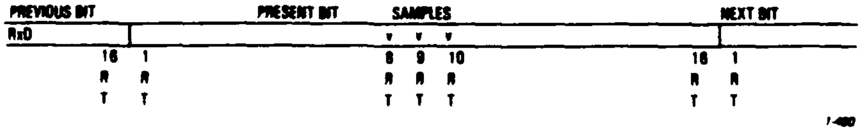


Figure 5-2. Sampling Technique Used on All Bits

5.5 START BIT DETECTION

When the RxD input is detected low, it is tested for three more sample times (referred to as the start edge verification samples in Figure 5-3). If at least two of these three verification samples detect a logic zero, a valid start bit has been detected, otherwise the line is assumed to be idle. A noise flag is set if all three verification samples do not detect a logic zero. A valid start bit could be assumed with a set noise flag present.

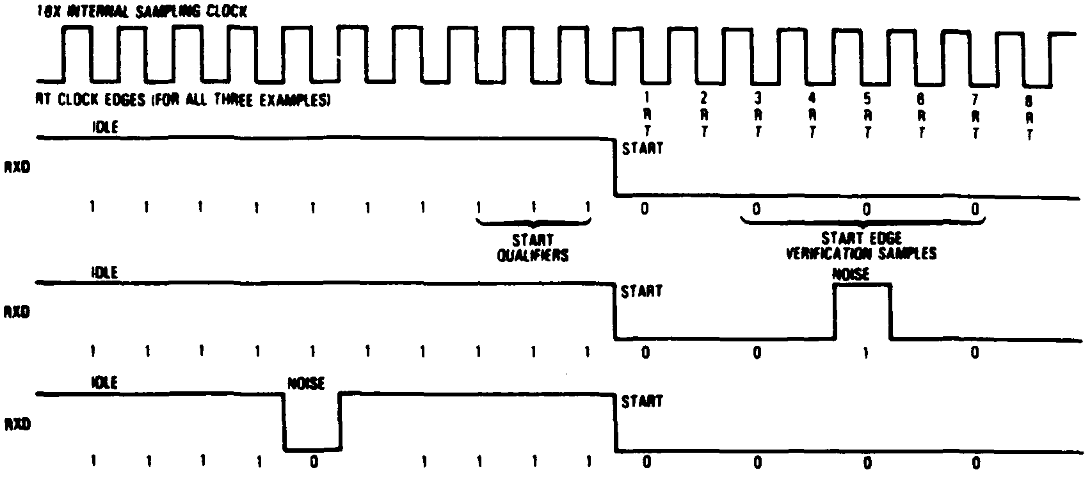
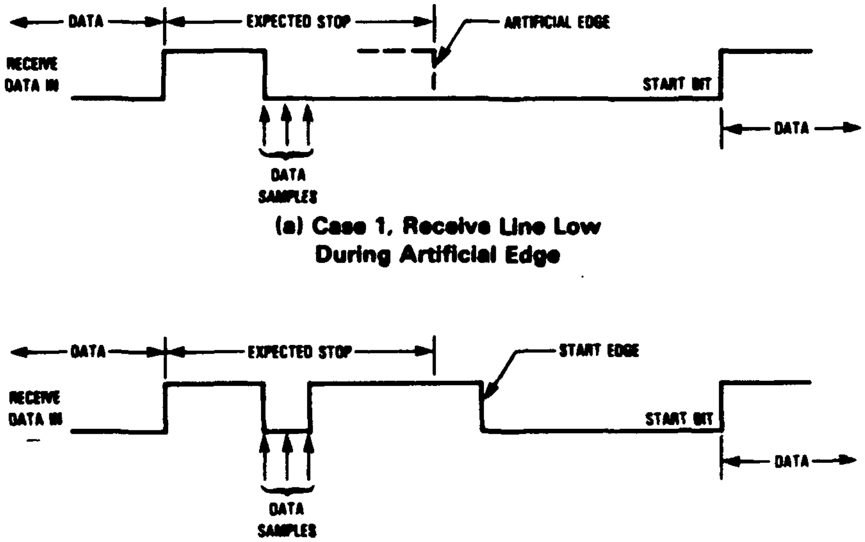


Figure 5-3. Examples of Start Bit Sampling Techniques

If there has been a framing error without detection of a break (10 zeros for 8-bit format or 11 zeros for 9-bit format), the circuit continues to operate as if there actually was a stop bit and the start edge will be placed artificially. The last bit received in the data shift register is inverted to a logic one, and the three logic one start qualifiers (shown in Figure 5-3) are forced into the sample shift register during the interval when detection of a start bit is anticipated (see Figure 5-4); therefore, the start bit will be accepted no sooner than it is anticipated.



(a) Case 1, Receive Line Low During Artificial Edge

(b) Case 2, Receive Line High During Expected Start Edge

Figure 5-4. SCI Artificial Start Following a Framing Error

If the receiver detects that a break produced the framing error, the start bit will not be artificially induced and the receiver must actually receive a logic one bit before start. See Figure 5-5.

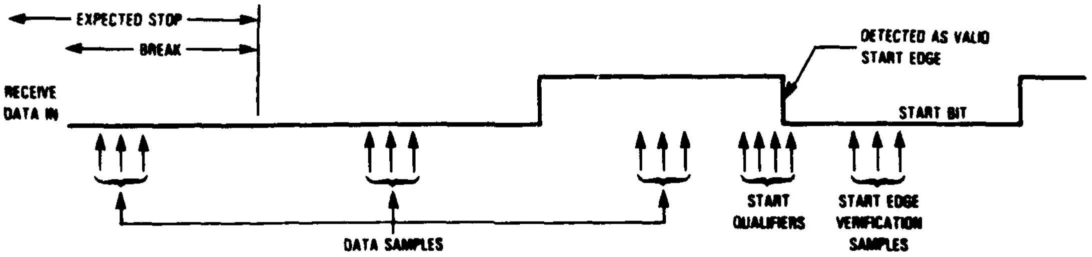


Figure 5-5. SCI Start Bit Following a Break

5.6 TRANSMIT DATA (TxD)

Transmit data (TxD) is the serial data which is presented from the internal data bus via the SCI and then to the output pin. The transmitter generates a bit time by using a derivative of the RT clock, thus producing a transmission rate equal to 1/16 that of the receiver sample clock.

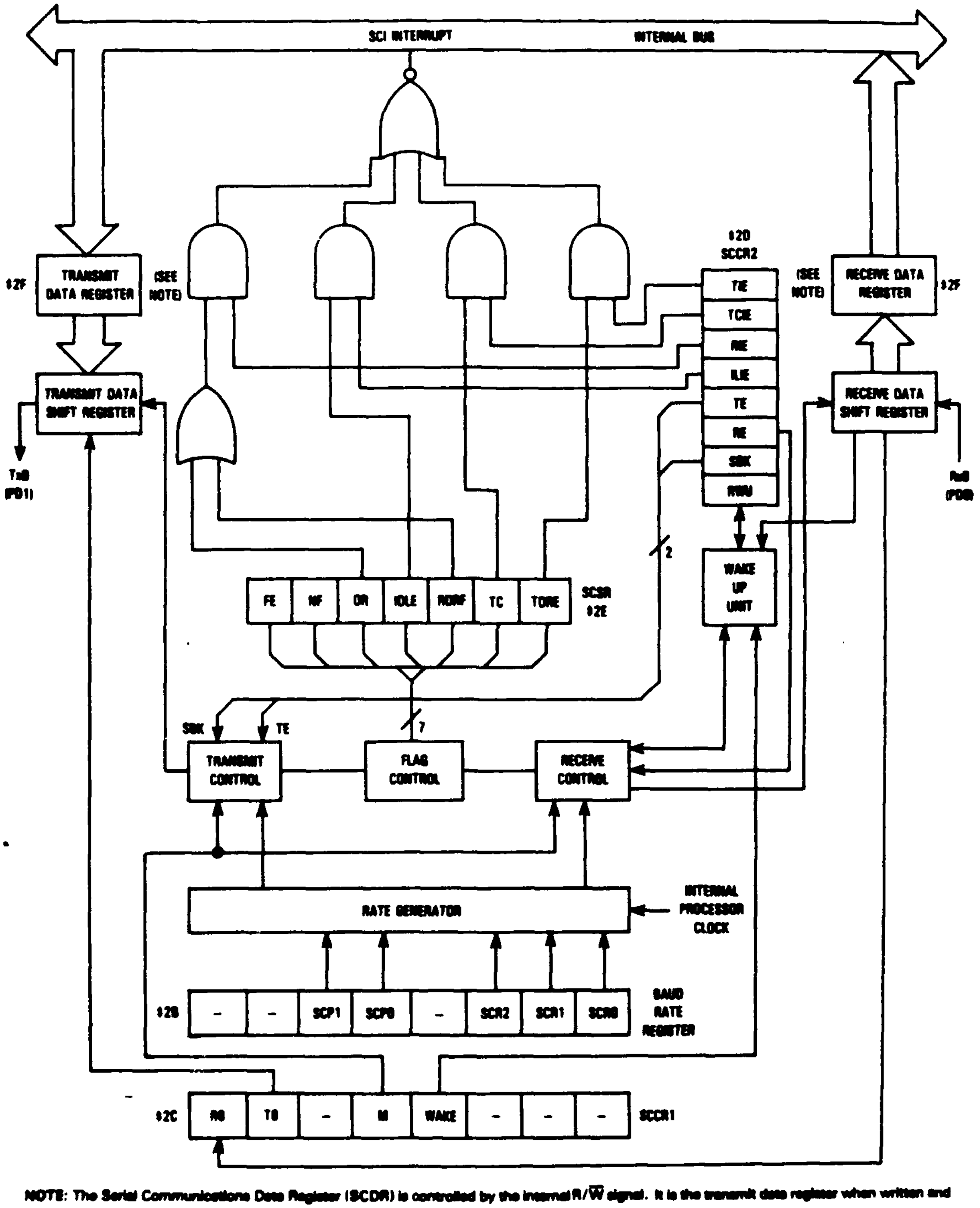
5.7 FUNCTIONAL DESCRIPTION

A block diagram of the SCI is shown in Figure 5-6. The user has option bits in serial communications control register 1 (SCCR1) to determine the "wake-up" method (WAKE bit) and data word length (M bit) of the SCI. Serial communications control register 2 (SCCR2) provides control bits which individually enable/disable the transmitter or receiver (TE and RE, respectively), enable system interrupts (TIE, TCIE, ILIE) and provide the wake-up enable bit (RWU) and the send break code bit (SBK). The baud rate register bits allow the user to select different baud rates which may be used as the rate control for the transmitter and receiver.

Data transmission is initiated by a write to the serial communications data register (SCDR). Provided the transmitter is enabled, data stored in the SCDR is transferred to the transmit serial shift register. This transfer of data sets the TDRE bit of the SCI status register (SCSR) and may generate an interrupt if the transmit interrupt is enabled. The transfer of data to the transmit shift register is synchronized with the bit rate clock (Figure 5-7). All data is transmitted bit zero first. Upon completion of data transmission, the TC (transmission complete) bit of the SCSR is set (provided no pending data, preamble, or break is to be sent), and an interrupt may be generated if the tranmit complete interrupt is enabled. If the the transmitter is disabled, and the data, preamble, or break (in the transmit shift register) has been sent, the TC bit will also be set. This will also generate an interrupt if the TCIE bit is set.

When the SCDR is read, it contains the last data byte received, provided that the receiver is enabled. The RDRF bit of the SCSR is set to indicate that a data byte has been transferred from the input serial shift register to the SCDR, which can cause an interrupt if the receiver interrupt is enabled. The data transfer from the input serial shift register to the SCDR is synchronized by the receiver bit rate clock. The OR (overrun), NF (noise), or FE (framing) error bits of the SCSR may be set if data reception errors occurred.

An idle line interrupt is generated if the idle line interrupt is enabled and the IDLE bit (which detects idle line transmission) of SCSR is set. This allows a receiver that is not in the wake-up mode to detect the end of a message, the preamble of a new message, or to resynchronize with the transmitter.



NOTE: The Serial Communications Data Register (SCDR) is controlled by the internal R/$\overline{W}$ signal. It is the transmit data register when written and receive data register when read.

Figure 5-6. Serial Communications Interface Block Diagram

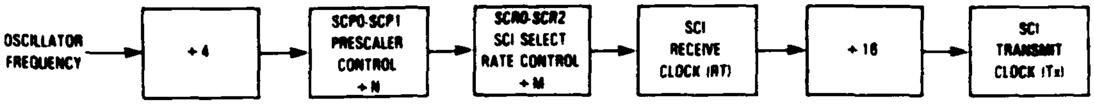


Figure 5-7. Rate Generator Division

5.8 REGISTERS

There are five different registers used in the serial communications interface (SCI) and the internal configuration of these registers is discussed in the following paragraphs. Refer to the block diagram shown in Figure 5-6.

5.8.1 SERIAL COMMUNICATIONS DATA REGISTER (SCDR)

The serial communications data register (Figure 5-8) performs two functions; i.e. it acts as the receive data register when it is read and as the transmit data register when it is written. Figure 5-6 shows this register as two separate registers, namely: the receive data register (RDR) and the transmit data register (TDR).

7	6	5	4	3	2	1	0	
SERIAL COMMUNICATIONS DATA REGISTER								$__02F

1-496

Figure 5-8. Serial Communications Data Register (SCDR)

5.8.2 Serial Communications Control Register 1 (SCCR1)

The serial communications control register 1 (SCCR1) (Figure 5-9) provides the control bits which: (1) determine the word length, and (2) selects the method used for the wake-up feature.

7	6	5	4	3	2	1	0	
R8	T8	–	M	WAKE	–	–	–	$__02C

1-497

Figure 5-9. Serial Communications Control Register 1 (SCCR1)

Bit 7, R8 — If the M bit is set, then this bit provides a storage location for the ninth bit in the receive data word. Reset does not affect this bit.

Bit 6, T8 — If the M bit is set, then this bit provides a storage location for the ninth bit in the transmit data word. Reset does not affect this bit.

Bit 5 — This bit is not implemented and reads as zero.

Bit 4, M — This bit selects the word length. Reset clears this bit.
0 = 1 start bit, 8 data bits, 1 stop bit
1 = 1 start bit, 9 data bits, 1 stop bit

Bit 3—WAKE This bit allows the user to select the method for receiver "wake up".

When clear, an idle line condition (10 consecutive ones if M = 0 or 11 consecutive ones if M = 1) will wake-up the receiver.

When set, detection of a one in last data bit (eighth data bit if M = 0, ninth data bit if M = 1) will wake-up the receiver.

Bit 2-0 These bits are not implemented and read as zeros.

5.8.3 Serial Communications Control Register 2 (SCCR2)

The serial communications control register 2 (SCCR2) (Figure 5-10) provides the control bits which: individually enable/disable the SCI functions.

7	6	5	4	3	2	1	0	
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	$_02D

Figure 5-10. Serial Communications Control Register 2 (SCCR2)

Bit 7, TIE When the transmit interrupt enable bit is set, the SCI interrupt occurs when TDRE is set. When TIE is clear, the TDRE interrupt is disabled. Cleared by reset.

Bit 6, TCIE When the transmission complete interrupt enable bit is set, the SCI interrupt occurs when TC is set. When TCIE is clear, the TC interrupt is disabled. Cleared by reset.

Bit 5, RIE When the receive interrupt enable bit is set, the SCI interrupt occurs when OR or RDRF are set. When RIE is clear, the OR and RDRF interrupts are disabled. Cleared by reset.

Bit 4, ILIE When the idle line interrupt enable bit is set, the SCI interrupt occurs when IDLE is set. When ILIE is clear, the IDLE interrupt is disabled. Cleared by reset.

Bit 3, TE When the transmit enable bit is set, the transmit shift register output is applied to the TxD line. Depending on the state of control bit M (SCCR1), a preamble of 10 (M = 0) or 11 (M = 1) consecutive ones is transmitted when software sets the TE bit from a cleared state. After loading the last byte in the serial communications data registor and receiving the interrupt from TDRE, the user can clear TE. Transmission of the last byte will then be completed before the transmitter gives up control of the TxD pin. Cleared by reset.

Bit 2, RE When the receive enable bit is set, the receiver is enabled. When RE is clear, the receiver is disabled and all of the status bits associated with the receiver (RDRF, IDLE, OR, NF, and FE) are inhibited. Cleared by reset.

Bit 1, RWU When the receiver wake-up bit is set, it enables the "wake up" function. If the WAKE bit is cleared, RWU is cleared after receiving 10 (M = 0) or 11 (M = 1) consecutive ones. If the WAKE bit is set, RWU is cleared after receiving a data word whose MSB is set. Cleared by reset.

Bit 0, SBK If the send break bit is toggled set and cleared, the transmitter sends 10 (M = 0) or 11 (M = 1) zeros and then reverts to idle or sending data. If SBK remains set, the transmitter will continually send whole blocks (sets of 10 or 11) zeros until cleared. At the completion of the break code, the transmitter sends at least one high bit to guarantee recognition of a valid start bit. Reset clears the SBK bit.

5.8.4 Serial Communications Status Register (SCSR)

The serial communications status register (SCSR) (Figure 5-11) provides inputs to the interrupt logic circuits for generation of the SCI system interrupt.

7	6	5	4	3	2	1	0	
TDRE	TC	RDRF	IDLE	OR	NF	FE	–	$__02E

1-4[illegible]

Figure 5-11. Serial Communications Status Register (SCSR)

Bit 7, TDRE The transmit data register empty bit is set to indicate that the content of the serial communications data register have been transferred to the transmit serial shift register. This bit is cleared by reading the SCSR (with TDRE = 1) followed by a write to the SCDR. Reset sets the TDRE bit.

Bit 6, TC The transmit complete bit is set at the end of a data frame, preamble, or break condition if:
1) TE = 1, TDRE = 1, and no pending data, preamble, or break is to be transmitted; or
2) TE = 0, and the data, preamble, or break (in the transmit shift register) has been transmitted.

The TC bit is a status flag which indicates that one of the above conditions have occurred. The TC bit is cleared by reading the SCSR (with TC set) followed by a write to the SCDR. Reset sets the TC bit.

Bit 5, RDRF The receive data register full bit is set when the receiver serial shift register is transferred to the SCDR. The RDRF bit is cleared when the SCSR is read (with RDRF set) followed by a read of the SCDR. Reset clears the RDRF bit.

Bit 4, IDLE The idle line detect bit, when set, indicates a receiver idle line is selected. The IDLE bit is cleared by reading the SCSR with IDLE set followed by a read of the SCDR. The IDLE bit is inhibited when the RWU bit is set. Reset clears the IDLE bit.

Bit 3, OR The overrun error bit is set when the next byte is ready to be transferred from the receive shift register to the SCDR which is already full (RDRF bit is set). The only valid data is located in SCDR when OR is set. The OR bit is cleared when the SCSR is read (with OR set), followed by a read of the SCDR. Reset clears the OR bit.

Bit 2, NF The noise flag bit is set if there is noise on any of the received bits, including the start and stop bits. The NF bit is not set until the RDRF flag is set. The NF bit is cleared when the $\overline{S}$CSR to read (with NF set), followed by a read of the SCDR. Reset clears the NF bit.

Bit 1, FE~ The framing error bit is set when no stop bit was detected in the data string received. The FE bit is set at the same time as the RDRF is set. If the byte received causes both framing and overrun errors, the processor will only recognize the overrun error. The framing error flag inhibits further transfer of data into the SCDR until it is cleared. The FE bit is cleared when the SCSR is read (with FE equal to one) followed by a read of the SCDR. Reset clears the FE bit.

Bit 0 Not implemented. Reads as zero.

5.8.5 Baud Rate Register

The baud rate register (Figure 5-12) provides the means for selecting different baud rates which may be used as the rate control for the transmitter and receiver. The SCP0-SCP1 bits function as a prescaler for the SCR0-SCR2 bits. Together, these five bits provide multiple baud rate combinations for a given crystal frequency.

7	6	5	4	3	2	1	0	
-	-	SCP1	SCP0	-	SCR2	SCR1	SCR0	$_028

1 500

Figure 5-12. Baud Rate Register

Bit 5, SCP1 Table 5-1 shows the prescale values attained from the E clock.
Bit 4, SCP0 Reset clears SCP1-SCP0 bits (divide-by-one).

Table 5-1. First Prescaler Stage

SCP1	SCP0	Internal Processor Clock Divide By
0	0	1
0	1	3
1	0	4
1	1	13

1-501

Bit 2, SCR2 These three bits select the baud rates of both the transmitter and the receiver. Table 5-2
Bit 1, SCR1 shows the prescaler value that divides the output of the first stage. Reset does not affect
Bit 0, SCR0 the SCR2-SCR0 bits.

Table 5-2. Second Prescaler Stage

SCR2	SCR1	SCR0	Prescaler Output Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

1 502

The diagram of Figure 5-7 and Tables 5-3 and 5-4 illustrate the divider chain used to obtain the baud rate clock. Note that there is a fixed rate divide-by-16 between the receive clock (RT) and the transmit clock (Tx). The actual divider chain is controlled by the combined SCP0-SCP1 and SCR0-SCR2 bits in the baud rate register as illustrated.

Table 5-3. Prescaler Highest Baud Rate Frequency Output

SCP Bit	Clk.* Divided By	Crystal Frequency (MHz)				
1 0		8.3886	8.0	4.9152	4.0	3.6864
0 0	1	131.072 K Baud	125.000 K Baud	76.80 K Baud	62.50 K Baud	57.60 K Baud
0 1	3	43.690 K Baud	41.666 K Baud	25.60 K Baud	20.833 K Baud	19.20 K Baud
1 0	4	32.768 K Baud	31.250 K Baud	19.20 K Baud	15.625 K Baud	14.40 K Baud
1 1	13	10.082 K Baud	9600 Baud	5.907 K Baud	4800 Baud	4430 Baud

*The clock in the "Clock Divided By" column is the internal processor clock.

1 503

NOTE

The divided frequences shown in Table 5-3 represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown below for some representative prescaler outputs.

Table 5-4. Transmit Baud Rate Output For a Given Prescaler Output

SCR Bits	Divided By	Representative Highest Prescaler Baud Rate Output				
2 1 0		131.072 K Baud	32.768 K Baud	76.80 K Baud	19.20 K Baud	9600 Baud
0 0 0	1	131.072 K Baud	32.768 K Baud	76.80 K Baud	19.20 K Baud	9600 Baud
0 0 1	2	65.536 K Baud	16.384 K Baud	38.40 K Baud	9600 Baud	4800 Baud
0 1 0	4	32.768 K Baud	8.192 K Baud	19.20 K Baud	4800 Baud	2400 Baud
0 1 1	8	16.384 K Baud	4.096 K Baud	9600 Baud	2400 Baud	1200 Baud
1 0 0	16	8.192 K Baud	2.048 K Baud	4800 Baud	1200 Baud	600 Baud
1 0 1	32	4.096 K Baud	1.024 K Baud	2400 Baud	600 Baud	300 Baud
1 1 0	64	2.048 K Baud	512 Baud	1200 Baud	300 Baud	150 Baud
1 1 1	128	1.024 K Baud	256 Baud	600 Baud	150 Baud	75 Baud

1 504

NOTE

Table 5-4 illustrates how the SCI select bits can be used to provide lower transmitter baud rates by further dividing the prescaler output frequency. The five examples are only representative samples. In all cases, the baud rates shown are transmit baud rates (transmit clock) and the receiver clock is 16 times higher in frequency than the actual baud rate.

SECTION 6
SERIAL PERIPHERAL INTERFACE (SPI)

This section contains a description of the serial peripheral interface (SPI).

6.1 OVERVIEW AND FEATURES

The serial peripheral interface (SPI) is a synchronous interface built into the MC68HC11A8 MCU which allows several MC68HC11A8 MCUs, or an MC68HC11A8 plus peripheral devices, to be interconnected. In an SPI, seperate wires (signals) are required for data and clock as the clock is not included in the data stream. An SPI system may be configured as a master or as a slave.

Features include:

- Full Duplex, Three-Wire Synchronous Transfers
- Master or Slave Operation
- 1.05 MHz (Maximum) Master Bit Frequency
- 2.1 MHz (Maximum) Slave Bit Frequency
- Four Programmable Master Bit Rates
- Programmable Clock Polarity and Phase
- End-of-Transmission Interrupt Flag
- Write Collision Flag Protection
- Master-Master Mode Fault Protection Capacity
- Easily Interfaces to Simple Expansion Parts (PLLs, D-A, Latches, Display Drivers, etc.)

6.2 SIGNAL DESCRIPTION

The four basic signals (MISO, MOSI, SCK, and $\overline{\text{SS}}$) used to transmit data by the serial peripheral interface are discussed in the following paragraphs. Each signal is described for both the master and slave modes.

Any SPI output has to have its corresponding data direction bit in DDRD set. If this bit is clear, the pin is disconnected from the SPI logic and becomes a general-purpose input.

6.2.1 Master In Slave Out (MISO)

The MISO pin is configured as an input in a master device and as an output in a slave device. It is one of the two wires that carry data in one direction, with the most significant bit sent first. The MISO pin of a slave device is placed in the high-impedance state if the slave is not selected.

6.2.2 Master Out Slave In (MOSI)

The MOSI pin is configured as a data output in a master device and as a data input in a slave device. It is one of the two lines that transfer serial data in one direction with the most significant bit sent first.

6.2.3 Serial Clock (SCK)

The serial clock is used to synchronize data movement both in and out of the device through its MOSI and MISO pins. The master and slave devices are capable of exchanging a data byte of information during a sequence of eight clock cycles. Since SCK is generated by the master device, this line becomes an input on a slave device.

As shown in Figures 6-1 and 11-17, four possible timing relationships may be chosen by using control bits CPOL and CPHA. Both master and slave devices must be operated in the same timing mode. The master device always allows data to be applied on the MOSI line a half-cycle before the clock edge (SCK), in order for the slave device to latch the data.

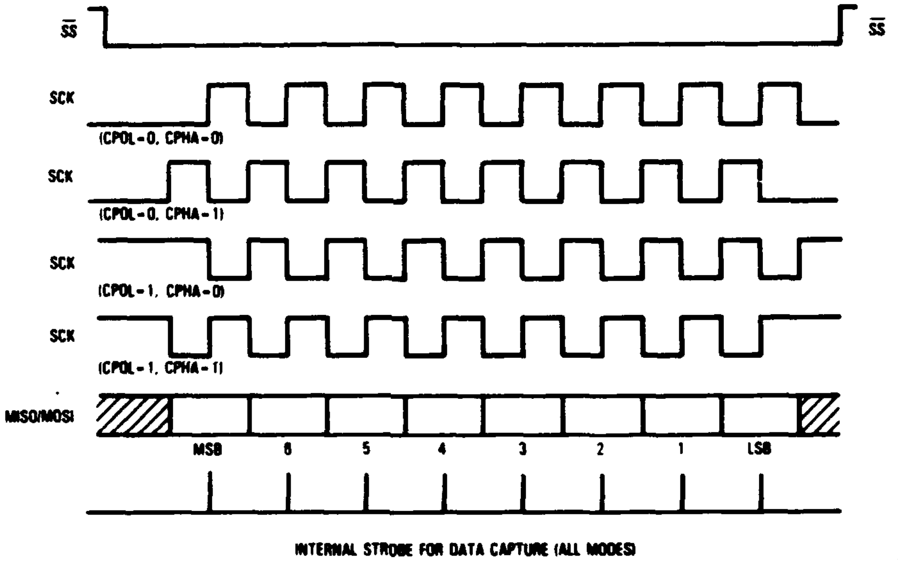


Figure 6-1. Data Clock Timing Diagram

Two bits (SPR0 and SPR1) in the SPCR of the master device select the clock rate. In the slave device, SPR0 and SPR1 have no effect on the operation of the SPI.

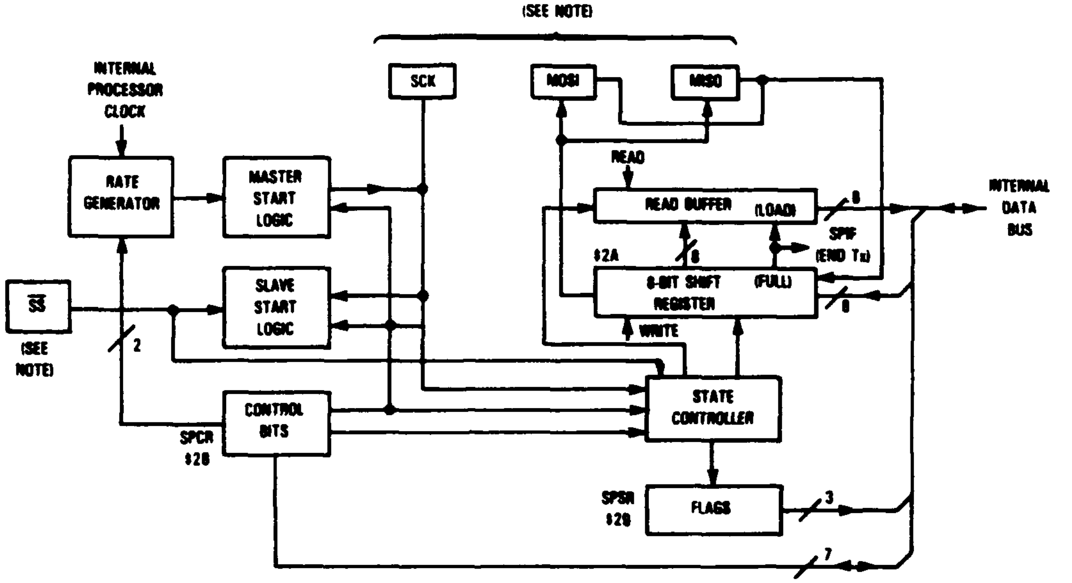
6.2.4 Slave Select ($\overline{\text{SS}}$)

The slave select ($\overline{\text{SS}}$) input is used to chip select a slave when it goes low. It has to be low prior to data transactions and must stay low for the duration of the transaction.

The $\overline{\text{SS}}$ pin on the master must be tied high. If it goes low for any reason, a mode fault error flag (MODF) is set in the SPSR. The $\overline{\text{SS}}$ pin can be selected to be a general-purpose output by writing a one in the port D data direction register bit 5, thus disabling the mode fault circuit.

6.3 FUNCTIONAL DESCRIPTION

Figure 6-2 shows a block diagram of the SPI. When the master device transmits data to a second (slave) device via the MOSI line, the slave device responds by sending data to the master device via the MISO line. This implies full duplex transmission with both data out and data in synchronized with the same clock signal. Thus, the byte transmitted is replaced by the byte received and eliminates the need for separate transmit-empty and receiver-full status bits. A single status bit (SPIF) is used to signify that the I/O operation is completed.



NOTES: The $\overline{SS}$, SCK, MOSI, and MISO are external pins which provide the following functions:

a. MOSI — Provides serial output to slave unit(s) when device is configured as a master. Receives serial input from master unit when device is configured as a slave unit.

b. MISO — Receives serial input from slave unit(s) when device is configured as a master. Provides serial output to master when device is configured as a slave unit.

c. SCK — Provides system clock when device is configured as a master unit. Receives system clock when device is configured as a slave unit.

d. $\overline{SS}$ — Provides a logic low to select a slave device for a transfer with a master device.

7-588

Figure 6-2. Serial Peripheral Interface Block Diagram

The SPI is double buffered on the read, but not the write. If a write is performed during data transfer, the transfer occurs uninterrupted, and the write will be unsuccessful. This condition will cause the write collision (WCOL) status bit of the SPSR to be set. After a data byte is shifted, the SPIF flag of the SPSR is set.

In the master mode, the SCK pin is an output. It idles high or low, depending on the CPOL bit in the SPCR, until data is written to the shift register, at which point eight clocks are generated to shift the eight bits of data and then SCK goes idle again.

In a slave mode, the slave start logic receives a logic low at the $\overline{SS}$ pin and a system clock input at the SCK pin. Thus, the slave is synchronized with the master. Data from the master is received serially at the slave MOSI pin and loads the 8-bit shift register. After the 8-bit shift register is loaded, its data is parallel transferred to the read buffer. During a write cycle, data is written into the shift register, then the slave waits for a clock train from the master to shift the data out on the MISO pin.

Figure 6-3 illustrates the MOSI, MISO, and SCK master-slave interconnections.

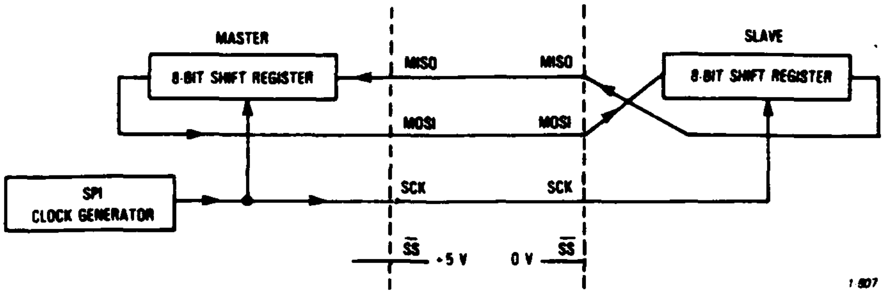


Figure 6-3. Serial Peripheral Interface Master-Slave Interconnection

6.4 REGISTERS

There are three registers in the serial parallel interface which provide control, status, and data storage functions.

6.4.1 Serial Peripheral Control Register (SPCR), Figure 6-4

5	6	5	4	3	2	1	0	
SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	$__028

1-472

Figure 6-4. Serial Peripheral Control Register (SPCR)

Bit 7, SPIE — When the serial peripheral interrupt enable bit is set, SPI interrupts are allowed. Interrupts are masked when this bit is cleared. The SPIE bit is cleared by reset.

Bit 6, SPE — When the serial peripheral enable bit is set, it enables the SPI system by connecting it to the external pins. Because the SPE bit is cleared by reset, the SPI system is not connected to the external pins upon reset.

Bit 5, DWOM — If the DWOM bit is set, port D output pins function as open-drain outputs, and when the DWOM bit is clear, port D output pins function normally. The DWOM bit is cleared by reset.

Bit 4, MSTR — If the MSTR bit is set, the SPI is a master device. If cleared, the SPI is a slave device.

Bit 3, CPOL — When the clock polarity bit is cleared and data is not being transferred, a steady state low value is produced at the SCK pin of the master device. Conversely, if this bit is set, the SCK pin will idle high. This bit is also used in conjunction with the clock phase control bit to produce the desired clock-data relationship between master and slave. The CPOL bit is cleared by reset. See Figure 6-1.

Bit 2, CPHA — The clock phase bit, in conjunction with the CPOL bit, controls the clock-data relationship between master and slave. In general, the CPHA bit selects which clock edge captures data and allows it to change states. The CPHA bit is set by reset. Refer to Figure 6-1.

Bit 1, SPR1
Bit 0, SPR0 These two serial peripheral rate bits select one of four baud rates (Table 6-1) to be used as SCK if the device is a master; however, they have no effect in the slave mode. The SPR1 and SPR0 bits are not affected by reset.

Table 6-1. Serial Peripheral Rate Selection

SPR1	SPR0	Internal Processor Clock Divide By
0	0	2
0	1	4
1	0	16
1	1	32

1-669

6.4.2 Serial Peripheral Status Register (SPSR), Figure 6-5

7	6	5	4	3	2	1	0	
SPIF	WCOL	–	MODF	–	–	–	–	$__029

1-470

Figure 6-5. Serial Peripheral Status Register (SPSR)

Bit 7, SPIF The serial peripheral data transfer flag bit is set upon completion of data transfer between the processor and external device. If SPIF goes high, and if SPIE is set, a serial peripheral interrupt is generated. While SPIF is set, all writes to the serial peripheral data register are inhibited. Clearing the SPIF bit is accomplished by reading the SPSR (with SPIF set) followed by an access of the SPDR. The SPIF bit is cleared by reset.

Bit 6, WCOL The write collision bit is set when an attempt is made to write to the serial peripheral data register while data transfer is taking place. Clearing the WCOL bit is accomplished by reading the SPSR (with WCOL set) followed by an access to SPDR. The WCOL bit is cleared by reset.

Bit 5 Not implemented and reads as zero.

Bit 4, MODF The mode fault flag indicates that there may have been a multi-master conflict for system control and allows a proper exit from system operation to a reset or default system state. The MODF bit is normally clear, and is set only when the master device has its $\overline{SS}$ pin pulled low. Setting the MODF bit affects the internal serial peripheral interface system in the following ways:

1) An SPI interrupt is generated if SPIE = 1.
2) The SPE bit is cleared. This disables the SPI.
3) The MSTR bit is cleared, thus forcing the device into the slave mode.

Clearing the MODF bit is accomplished by reading the SPSR (with MODF set), followed by a write to the SPCR. Control bits SPE and MSTR may be restored to their original set state after the MODF bit has been cleared. The MODF bit is cleared by reset.

Bits 3-0 Bits 3, 2, 1, and 0 are not implemented and read as zeros.

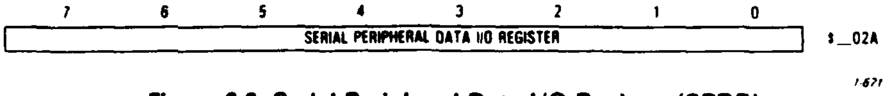


Figure 6-6. Serial Peripheral Data I/O Register (SPDR)

The serial peripheral data I/O register is used to transmit and receive data on the serial bus. Only a write to this register will initiate transmission/reception of another byte, and this will only occur in the master device. At the completion of transmitting a byte of data, the SPIF status bit is set in both the master and slave devices.

When the user reads the serial peripheral data I/O register, a buffer is actually being read. The first SPIF must be cleared by the time a second transfer of data from the shift register to the read buffer is initiated or an overrun condition will exist.

A write to the serial peripheral data I/O register is not buffered and places data directly into the shift register for transmission.

SECTION 7
ANALOG-TO-DIGITAL CONVERTER

The MC68HC11A8 includes an 8-channel multiplexed-input successive approximation analog-to-digital (A/D) converter with sample and hold to minimize conversion errors caused by rapidly changing input signals. Two dedicated pins (V_{RL}, V_{RH}) are provided for the reference supply voltage inputs. These pins may be connected to a lower noise power supply to assure full accuracy of the A/D conversion. The 8-bit A/D converter has a linearity error of ± ½ LSB and accepts analog inputs which range from V_{RL} to V_{RH}. Smaller analog input ranges can also be obtained by adjusting V_{RH} and V_{RL} to the desired upper and lower limits. Each conversion is accomplished in 32 MCU E clock cycles, provided the E rate is equal to or greater than 1 MHz. For systems which operate at clock rates less than 1.0 MHz, an internal R-C oscillator must be used to clock the A/D system by setting the CSEL bit in the OPTION register.

NOTE

Only four A/D inputs are available in the 48-pin package version of the MC68HC11A8.

7.1 CONVERSION PROCESS

The A/D converter is ratiometric. An input voltage equal to V_{RL} converts to $00 and an input voltage equal to V_{RH} converts to $FF (full scale), with no overflow indication. For ratiometric conversions, the source of each analog input should use V_{RH} as the supply voltage and be referenced to V_{RL}.

7.2 CHANNEL ASSIGNMENTS

A multiplexer allows the single A/D converter to select one of sixteen analog signals. Eight of these channels correspond to port E input pins to the MCU, four of the channels are for internal reference points or test functions, and four channels are reserved for future use. Table 7-1 shows the signals selected by the four channel select control bits.

Table 7-1. Analog-to-Digital Channel Assignments

CD	CC	CB	CA	Channel Signal	Result in ADRx If MULT = 1
0	0	0	0	AN0	ADR1
0	0	0	1	AN1	ADR2
0	0	1	0	AN2	ADR3
0	0	1	1	AN3	ADR4
0	1	0	0	AN4*	ADR1
0	1	0	1	AN5*	ADR2
0	1	1	0	AN6*	ADR3
0	1	1	1	AN7*	ADR4
1	0	0	0	Reserved	ADR1
1	0	0	1	Reserved	ADR2
1	0	1	0	Reserved	ADR3
1	0	1	1	Reserved	ADR4
1	1	0	0	V_{RH} Pin	ADR1
1	1	0	1	V_{RL} Pin	ADR2
1	1	1	0	(V_{RH})/2	ADR3
1	1	1	1	Reserved	ADR4

*Not available in 48-pin package versions.

1-509

7.3 SINGLE-CHANNEL OPERATION

There are two variations of single-channel operation. In the first variation (SCAN = 0), the single-selected channel is converted four consecutive times with the first result being stored in the ADR1 result register and the fourth result being stored in the ADR4 register. After the fourth conversion is complete, all conversion activity is halted until a new conversion command is written to the ADCTL control register. In the second variation (SCAN = 1), conversions continue to be performed on the selected channel with the fifth conversion being stored to the ADR1 register (overwriting the first conversion result), the sixth conversion overwrites ADR2, and so on continuously.

7.4 FOUR-CHANNEL OPERATION

There are two variations in multiple-channel operation. In the first variation (SCAN = 0), the selected group of four channels are converted, one time each, with the first result being stored in the ADR1 result register and the fourth result being stored to the ADR4 register. After the fourth conversion is complete, all conversion activity is halted until a new conversion command is written to the ADCTL control register. In the second variation (SCAN = 1), conversions continue to be performed on the selected group of channels with the fifth conversion being stored in the ADR1 register (replacing the earlier conversion result for the first channel in the group), the sixth conversion overwrites ADR2, and so on continuously.

7.5 OPERATION IN STOP AND WAIT MODES

If a conversion sequence is still in process when the MC68HC11A8 enters the STOP or WAIT mode, the conversion of the current channel is suspended. When the MCU resumes normal operation, that channel will be re-sampled and the conversion sequence resumes. As the MCU exits the WAIT mode, the A/D circuits are stable and valid results can be obtained on the first conversion. However, in STOP mode, all analog bias currents are disabled and it becomes necessary to allow a stabilization period when leaving the STOP mode. If the MC68HC11A8 exits the STOP mode with a delay, there will automatically be enough time for these circuits to stabilize before the first conversion. If the MC68HC11A8 exits the STOP mode with no delay (DLY bit in OPTION register equal to zero), the user must allow sufficient time for the A/D circuitry to stabilize to avoid invalid results.

7.6 A/D CONTROL/STATUS REGISTER (ADCTL)

All bits in this register may be read or written, except bit 7 which is a read-only status indicator and bit 6 which always reads as a zero. Bit 7 is cleared at reset but the other bits are not affected by reset. Figure 7-1 and the following paragraphs describe the function of each of the bits.

B7	B6	B5	B4	B3	B2	B1	B0	
CCF	–	SCAN	MULT	CD	CC	CB	CA	$_030

Figure 7-1. A/D Control/Status Register (ADCTL)

Bit 7, CCF — Conversions Complete Flag — This read-only status indicator is set when all four A/D result registers contain valid conversion results. Each time the ADCTL register is written, this bit is automatically cleared to zero and a conversion sequence is started. In the continuous modes, conversions continue in a round-robin fashion and the result registers continue to be updated with current data even though the CCF bit remains set.

NOTE

The user must write to register ADCTL to initiate conversion.

Bit 6 Not implemented. Reads as zero.

Bit 5, SCAN Continuous Scan Control—When this control bit is cleared, the four requested conversions are performed once to fill the four result registers. When this control bit is set, conversions continue in a round-robin fashion with the result registers being updated as data becomes available.

Bit 4, MULT Multiple-Channel/Single Channel Control—When this bit is cleared, the A/D system is configured to perform four consecutive conversions on the single channel specified by the four channel select bits CD thru CA (bits 3-0 of ADCTL). When this bit is set, the A/D system is configured to perform a conversion on each of four channels where each result register corresponds to one channel.

Bit 3, CD Channel Select D
Bit 2, CC Channel Select C
Bit 1, CB Channel Select B
Bit 0, CA Channel Select A—These four bits are used to select one of 16 A/D channels (see Table 7-1). When a multiple channel mode is selected (MULT = 1), the two least-significant channel select bits (CB and CA) have no meaning and the CD and CC bits specify which group of four channels are to be converted. The signals selected by the four channel select control bits are shown in Table 7-1.

7.7 A/D RESULT REGISTERS 1, 2, 3, AND 4 (ADR1, ADR2, ADR3, and ADR4)

The A/D result registers are read-only registers used to hold an 8-bit conversion result. Writes to these registers have no effect. Data in the A/D result registers is not valid unless the CCF flag bit in ADCTL is set, indicating conversion complete. Refer to the A/D channel assignments in Table 7-1 for the relationship between the channels and the result registers.

7.8 A/D POWER UP AND CLOCK SELECT

A/D power up is controlled by bit 7 (ADPU) of the OPTION register. When ADPU is cleared, power to the A/D system is disabled. When ADPU is set, the A/D system is enabled. A delay of typically 100 microseconds is required after turning on the A/D converter to allow the analog bias voltages to stabilize.

Clock select is controlled by bit 6 (CSEL) of the OPTION register. When CSEL is cleared, the A/D uses the system E clock. When CSEL is set, the A/D uses an internal R-C clock source, which runs at about 1.5 MHz. The MCU E clock is not suitable to drive the A/D system if it is operating below 1 MHz, in which case the R-C internal clock should be selected. Refer to 9.2 CONFIGURATION OPTIONS REGISTER (OPTION) for more detailed information.

SECTION 8
PROGRAMMABLE TIMER, REAL-TIME INTERRUPT, AND PULSE ACCUMULATOR

This section describes the 16-bit programmable timer, the real-time interrupt, and the pulse accumulator system features of the MC68HC11A8.

8.1 PROGRAMMABLE TIMER

The timer has a single 16-bit free-running counter which is clocked by the output of a four-stage prescaler (divide by 1, 4, 8, or 16), which is in turn driven by the MCU E clock. Input functions are called input captures. These input captures record the count from the free-running counter in response to a detected edge on an input pin. Output functions, called output compares, cause an output action when there is a match between a 16-bit output-compare register and the free-running counter. This timer system has three input capture registers and five output compare registers.

In order to reduce the overhead required to service interrupts, timer overflow, and the other eight timer functions, each has a separate interrupt vector, and each of the nine interrupts is separately maskable. A change in the way interrupt flag bits are cleared has been incorporated to enhance timer interrupt operation.

8.1.1 Counter

The key element in the timer system is a 16-bit free-running counter, or timer counter register. After reset, the MC68HC11A8 is configured to use the E clock as the input to the free-running counter. Initialization software may optionally reconfigure the system to use one of the three prescale taps. Software can read the counter at any time without affecting its value because it is clocked and read during opposite half cycles of the MPU E clock.

A counter read should first address the most significant byte. An MPU read of this address causes the least significant byte to be transferred to a buffer. This buffer is not affected by reset and is accessed when reading the least significant byte of the counter. For double byte read instructions, the two accesses occur on consecutive bus cycles.

The counter is cleared to $0000 during reset and is a read-only register with one exception. In test modes only, any MPU write to the most significant byte always presets the counter to $FFF8 regardless of the value involved in the write.

When the count changes from $FFFF to $0000, the timer overflow flag (TOF) bit is set in TFLG2. An interrupt can be enabled by setting the interrupt enable bit (TOI) in TMSK2.

8.1.2 Input Capture

The input capture registers are 16-bit read-only registers which are not affected by reset and are used to latch the value of the counter when a defined transition is sensed by the corresponding input capture edge detector. The level transition which triggers counter transfer is defined by the corresponding input edge bits (EDGxB, EDGxA) in TCTL2.

The result obtained by an input capture corresponds to the value of the counter one cycle after the transition which triggered the edge-detection logic. The selected edge transition sets the ICxF in TFLG1 and can cause an interrupt if the corresponding ICxI bit(s) is (are) set in the TMSK1 register. A read of the Input Capture Register's MSB inhibits captures for one E cycle to allow a double-byte read of the full 16-bit register.

8.1.3 Output Compare

The output compare registers are 16-bit read/write registers which are initialized to $FFFF by reset. They can be used as output waveform controls and as elapsed time indicators. If an output compare is not utilized, the unused registers may be used as storage locations.

All output compare registers have a separate dedicated comparator for comparing against the free-running counter. If a match is found, the corresponding output compare flag (OCxF) bit in TFLG1 is set and a specified action is automatically taken. For output compare functions two through five the automatic action is controlled by pairs of bits in the TCTL1 control register (OMX and OLX). Each pair of control bits are encoded to specify the output action to be taken as a result of a successful OCx compare.

An interrupt can also accompany a successful output compare, provided that the corresponding interrupt enable bit (OCxI) is set in TMSK1.

After an MPU write cycle to the most significant byte, output compares are inhibited for one E cycle in order to allow writing two consecutive bytes before making the next comparison. If both bytes of the register are to be changed, a double-byte write instruction should be used in order to take advantage of the compare inhibit feature.

MPU writes can be made to either byte of the output compare register without affecting the other byte.

A write-only register (CFORC), allows forced compares. Five of the bit positions in the CFORC register correspond to the five output compares. To force a compare, or compares, a write is done to CFORC with the associated bits set for each output compare that is to be forced. The action taken as a result of a forced compare is the same as if there was a match between the OCx register and the free-running counter, except that the corresponding interrupt flag status bits are not set.

8.1.4 Output Compare 1 I/O Pin Control

Unlike the other four output compares, output compare 1 can automatically affect any or all of the five output pins (bits 3-7) in port A as a result of a successful compare between the OC1 register and the 16-bit free-running counter. The two 5-bit registers used in conjunction with this function are the output compare 1 mask register (OC1M) and the output compare 1 data register (OC1D).

Register OC1M is used to specify the bits of port A (I/O and timer port) which are to be affected as a result of a successful OC1 compare. Register OC1D is used to specify the data which is to be stored to the affected bits of port A as the result of a successful OC1 compare. If an OC1 compare and another output compare occur during the same E cycle and both attempt to alter the same port A bit, the OC1 compare overrides.

This function allows control of multiple I/O pins automatically with a single output compare.

Another intended use for the special I/O pin control on output compare 1 is to allow more than one output compare to control a single I/O pin.

8.1.5 Timer Compare Force Register (CFORC)

The compare force reg'ster is used to force early output compare actions. The CFORC register is an 8-bit write-only register except that bits 2, 1, and 0 are not implemented. Reads of this location have no meaning or effect and always return logic zeros ($00). Note that the compare force function is not generally recommended for use with the output toggle function because a normal compare occuring immediately before or after the force may result in undesirable operation. Figure 8-1 and the following paragraphs describe the function of each of the bits.

B7	B6	B5	B4	B3	B2	B1	B0	
FOC1	FOC2	FOC3	FOC4	FOC5	–	–	–	$_00B

1-510

Figure 8-1. Timer Compare Force Register (CFORC)

FOC1-FOC5 Force Output Compare x Action
0—Has no meaning
1—Causes the action which is programmed for output compare except that the OCxF is not set.

Bits 2-0 Not Implemented. Read as a logic zero.

8.1.6 Output Compare 1 Mask Register (OC1M)

This 8-bit read/write register (Figure 8-2) is cleared by reset and is used in conjunction with output compare 1 to specify the bits of port A which are to be affected as a result of a successful OC1 compare. Bits zero through two are not implemented and always return zero.

B7	B6	B5	B4	B3	B2	B1	B0	
OC1M7	OC1M6	OC1M5	OC1M4	OC1M3	–	–	–	$_00C

1-511

Figure 8-2. Output Compare 1 Mask Register (OC1M)

The bits of OC1M register correspond bit-for-bit with the bits of port A (bits 7 through 3 only). For each bit that is affected by the successful compare, the corresponding bit in OC1M should be set to one.

Note that the pulse accumulator function shares bit 7 of port A. If the DDRA7 control bit in the PACTL register is set, then port A bit 7 is configured as an output and OC1 can obtain access by setting OC1M bit 7. In this condition if the PAEN control bit in the PACTL register is set, enabling the pulse accumulator input, then OC1 compares cause a write of OC1D bit 7 to an internal latch, and the output of that latch drives the pin and the pulse accumulator input. This action can then cause the pulse accumulator to take the appropriate action (pulse count or gate modes).

8.1.7 Output Compare 1 Data Register (OC1D)

This 8-bit, read/write register (Figure 8-3) is cleared by reset and is used in conjunction with output compare 1 to specify the data which is to be stored to the affected bits of port A as the result of a successful OC1 compare. This register is not affected by reset. Bits zero through two are not implemented and always read as zeros.

B7	B6	B5	B4	B3	B2	B1	B0	
OC1D7	OC1D6	OC1D5	OC1D4	OC1D3	–	–	–	$__000

I-512

Figure 8-3. Output Compare 1 Data Register (OC1D)

The bits of OC1D correspond bit-for-bit with the bits of port A (bits 7 thru 3 only). When a successful OC1 compare occurs, for each bit that is set in OC1M, the corresponding data bit in OC1D is stored in the corresponding bit of port A. If there is a conflicting situation where an OC1 compare and another output compare function occur during th3 same E cycle with both attempting to alter the same port A bit, the OC1 action overrides.

8.1.8 Timer Control Register 1 (TCTL1)

Timer control register 1 is an 8-bit read/write register. All bits in this register are cleared to zero during reset. Figure 8-4 and the following paragraphs describe the function of each of the bits.

B7	B6	B5	B4	B3	B2	B1	B0	
OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5	$__020

I-513

Figure 8-4. Timer Control Register 1 (TCTL1)

Bits 7, 5, 3, 1 OMx — Output Mode

Bits 6, 4, 2, 0 OLx — Output Level. These two control bits (OMx and OLx) are encoded to specify the output action to be taken as a result of a successful OCx compare.

OMx	OLx	Action Taken Upon Successful Compare
0	0	Timer disconnected from output pin logic
0	1	Toggle OCx output line
1	0	Clear OCx output line to zero
1	1	Set OCx output line to one

8.1.9 Timer Control Register 2 (TCTL2)

Timer control register 2 is an 8-bit read/write register except for bits 6 and 7 which are not implemented. Figure 8-5 and the following paragraphs describe the function of each of the bits.

B7	B6	B5	B4	B3	B2	B1	B0	
–	–	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A	$__021

I-514

Figure 8-5. Timer Control Register 2 (TCTL2)

Bits 7-6 Not Implemented. Read as a logic zeros.

Bits 5, 3, 1 EDGxB
Bits 4, 2, 0 EDGxA

Input Capture x Edge Control. These two bits (EDGxB and EDGxA) are cleared to zero by reset and are encoded to configure the input sensing !ogic for input capture x as follows:

EDGxB	EDGxA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any (rising or falling) edge

8.1.10 Main Timer Interrupt Mask Register 1 (TMSK1)

The timer interrupt mask register (see Figure 8-6) *is a read/write register and the bits are described in the* following paragraphs.

B7	B6	B5	B4	B3	B2	B1	B0
OC1I	OC2I	OC3I	OC4I	OC5I	IC1I	IC2I	IC3I

$_022

I 515

Figure 8-6. Timer Interrupt Mask Register 1 (TMSK1)

Bits 7-3, OCxI Output Compare x Interrupt. If the OCxI mask bit is set when the OCxF flag bit is set, a hardware interrupt sequence is requested.

Bits 2-0, ICxI Input Capture x Interrupt. If the ICxI mask bit is set when the ICxF flag bit is set, a hardware interrupt sequence is requested.

8.1.11 Main Timer Interrupt Flag Register 1 (TFLG1)

The timer system flag register 1 (TFLG1), shown in Figure 8-7, is used to indicate the occurrence of timer system events, and together with the TMSK1 register allows the timer sub-system to operate in a polled or interrupt driven system. For each bit in the timer flag register 1 (TFLG1), there is a corresponding bit in the timer mask register 1 (TMSK1) in the same bit position. If, and only if, the mask bit is set each time the conditions for the corresponding flag are met, a hardware interrupt sequence is requested as well as the flag bit being set.

Bit manipulation instructions would be inappropriate for flag clearing because they are read-modify-write instructions. Even though the instruction mask implies that the programmer is only interested in some of the bits in the manipulated location, the entire 8-bit location is actually read and rewritten which may clear other bits in the register.

B7	B6	B5	B4	B3	B2	B1	B0
OC1F	OC2F	OC3F	OC4F	OC5F	IC1F	IC2F	IC3F

$_023

I 516

Figure 8-7. Timer Interrupt Flag Register 1 (TFLG1)

Bits 7-3, OCxF Output Compare x Flag. This flag bit is set each time the timer counter matches the output compare register x value. A write of a zero does not affect the state of this bit. A write of a one causes this bit to be cleared.

Bits 2-0, ICxF Input Capture x Flag. This flag bit is set each time a selected active edge is detected on the ICx input line. A write of a zero, does not affect this bit. A write of a one causes this bit to be cleared.

8.1.12 Timer Interrupt Mask Register 2 (TMSK2)

The timer system mask register 2 is used to control whether or not a hardware interrupt sequence is requested as a result of a status bit being set in timer system flag register 2. In addition, two timer prescaler bits are included in this register. For each of the four most significant bits in timer flag register 2 (TFLG2) there is a corresponding bit in the timer mask register 2 (TMSK2) in the same bit position.

The timer interrupt mask register is a read/write register and the bits are described in Figure 8-8 and the following paragraphs.

B7	B6	B5	B4	B3	B2	B1	B0
TOI	RTII	PAOVI	PAII	–	–	PR1	PR0

$_024

1517

Figure 8-8. Timer Interrupt Mask Register 2 (TMSK2)

Bit 7, TOI Timer Overflow Interrupt Enable. This read/write bit is cleared by reset. If this bit is set, a timer overflow interrupt request is initiated each time the TOF bit (in TFLG2) is set as a result of a timer counter overflow.

Bit 6, RTII RTI Interrupt Enable. This read/write bit is cleared to zero by reset and is used to enable or inhibit RTIF interrupt flags from causing hardware interrupt sequences. When RTII is clear, the RTIF flag is masked (inhibited). When RTII is set, the RTIF flag is enabled to causes a hardware interrupt.

Bit 5, PAOVI Pulse Accumulator Overflow Interrupt Enable. This read/write bit is cleared to zero by reset and is used to enable or inhibit PAOVF interrupt flags from causing hardware interrupt sequences. When it is set, a hardware interrupt results when the PAOVF bit is set.

Bit 4, PAII Pulse Accumulator Input Interrupt Enable. This read/write bit is cleared to zero by reset and is used to enable or inhibit PAIF interrupt flags from causing hardware interrupt sequences. When it is set, a hardware interrupt results if the PAIF bit is set.

Bits 3, 2 These bits are not implemented. Reads of these bits will always return a logic zero.

Bit 1, PR1
Bit 0, PRO Timer Prescaler Selects. These two bits may be read at any time but may only be written during initialization. Writes are disabled after the first write or after 64 E cycles out of reset. If the MCU is in special test or special bootstrap mode, then these two bits may be written any time.

These two bits specify the timer prescaler divide factor.

PR1	PRO	Divide-by-Factor
0	0	1
0	1	4
1	0	8
1	1	16

8.1.13 Timer Interrupt Flag Register 2 (TFLG2)

Timer system flag register 2 (TFLG2) is used to indicate the occurrence of timer system events and, together with the TMSK2 register, allows the timer sub-systems to operate in a polled or interrupt driven system. For each bit in timer flag register 2 (TFLG2), there is a corresponding bit in timer mask register 2 (TMSK2) in the same bit position. If the mask bit is set each time the conditions for the corresponding flag are met, a hardware interrupt sequence is requested, as well as the flag bit being set.

The timer system status register indicates when interrupt conditions have occurred. To clear a bit, or bits, a logic one is written to it, or them.

Bit manipulation instructions would be inappropriate for flag clearing because they are read-modify-write instructions. Even though the instruction mask implies that the programmer is only interested in some of the bits in the manipulated location, the entire 8-bit location is actually read and rewritten which may clear other bits in the register.

The timer system flag register 2 is shown in Figure 8-9 and the bits are described in the following paragraphs.

B7	B6	B5	B4	B3	B2	B1	B0	
TOF	RTIF	PAOVF	PAIF	–	–	–	–	$__025

1-518

Figure 8-9. Timer Interrupt Flag Register 2 (TFLG2)

Bit 7, TOF — Timer Overflow. This bit is cleared by reset. It is set to one each time the 16-bit free-running counter advances from a value of $FFFF to $0000. In order to acknowledge (clear) the TOF flag the user must perform a write operation to TFLG2 with bit 7 set.

Bit 6, RTIF — Real Time Interrupt Flag. This bit is cleared by reset. RTIF is set at each rising edge of the selected tap point. To clear the RTIF flag, a software write is performed to the TFLG2 register with bit 6 set to one.

Bit 5, PAOVF — Pulse Accumulator Overflow Interrupt Flag. This bit is cleared by reset and becomes set when the count in the pulse accumulator rolls over from $FF to $00. This bit is cleared by a write to the TFLG2 register with bit 5 set.

Bit 4, PAIF — Pulse Accumulator Input Edge Interrupt Flag. This bit is cleared by reset, and becomes set when an active edge is detected on the PAI input pin. This bit is cleared by a write to the TFLG2 register with bit 4 set.

Bits 3-0 — These bits are not implemented and they read as a logic zero.

8.2 REAL TIME INTERRUPT

The real time interrupt feature on the MC68HC11A8 is configured and controlled by two bits in the PACTL control register(RTR1 and RTR0) to select one of four interrupt rates. The RTII bit in TMSK2 enables the interrupt capability. Every timeout causes the RTIF bit to be set in TFLG2, and if RTII is set, an interrupt request is generated. After reset, one entire real time interrupt period elapses before the RTIF flag is set for the first time.

8.3 PULSE ACCUMULATOR

The pulse accumulator is an 8-bit counter which can operate in either of two modes depending on the state of the PAMOD control bit in PACTL. In the event counting mode, the 8-bit counter is clocked to increasing values by an external pin. In the gated time accumulation mode, an E/64 clock drives the 8-bit counter, but only while the external PAI input pin is in a selected state.

The pulse accumulator uses port A bit 7 as its PAI input, but this pin also shares function as a general purpose I/O pin and as a timer output compare 1 output. Although the port A bit 7 pin would normally be configured as an input when being used for the pulse accumulator, it still drives the pulse accumulator system even when it is configured for use in its alternate capacities.

8.4 PULSE ACCUMULATOR CONTROL REGISTER (PACTL)

Four bits in this register are used to control an 8-bit pulse accumulator system and two other bits are used to select the rate for the real time interrupt system. Figure 8-10 and the following paragraphs describe the function of each of the bits.

B7	B6	B5	B4	B3	B2	B1	B0	
DDRA7	PAEN	PAMOD	PEDGE	-	-	RTR1	RTR0	$_026

I-519

Figure 8-10. Pulse Accumulator Control Register (PACTL)

Bit 7, DDRA7 Data Direction for Port A Bit 7. This read/write bit is cleared by reset and is used to enable or disable the output driver for the port A bit 7 pin. When DDRA7 is zero the port A bit 7 pin is configured for input only and when DDRA7 is one port A bit 7 is configured for output. Note that even when port A bit 7 is configured for output, this pin still acts as the input to the pulse accumulator system.

Bit 6, PAEN Pulse Accumulator System Enable. This read/write bit is cleared by reset and is used to enable or disable the pulse accumulator system. When it is set, the pulse accumulator is enabled, and when clear, the pulse accumulator system is disabled.

Bit 5, PAMOD Pulse Accumulator Mode. This read/write bit controls whether the pulse accumulator is to operate in the external event counting mode or the gated time accumulation mode. When it is zero, the pulse acumulator counts pulses on the PAI input pin (port A bit 7). When it is set, the pulse accumulator operates in the gated time accumulation mode and the PAI input allows the pulse accumulator counter to count. The PAMOD bit is cleared to zero by reset.

Bit 4, PEDGE Pulse Accumulator Edge Control. This read/write bit has different meanings depending on the state of the PAMOD control bit. This bit is cleared by reset.

PAMOD	PEDGE	Action on Clock
0	0	PAI Falling Edge Increments the Counter
0	1	PAI Rising Edge Increments the Counter
1	0	A '0' on PAI Inhibits Counting
1	1	A '1' on PAI Inhibits Counting

Bits 3-2 These bits are not implemented and read as a logic zero.

Bit 1, RTR1
Bit 0, RTR0 RTI Interrupt Rate Selects. These two read/write bits select one of four rates for the real time periodic interrupt circuit (see Table 8-1). Reset clears these two bits and after reset, a full RTI period elapses before the first RTI interrupt.

Table 8-1. Real Time Interrupt Rate versus RTR1 and RTR0

RTR1	RTR0	Divide E By	XTAL = 2^{23}	XTAL = 8.0 MHz	XTAL = 4.9152 MHz	XTAL = 4.0 MHz	XTAL = 3.6864 MHz
0	0	2^{13}	3.91 ms	4.10 ms	6.67 ms	8.19 ms	8.89 ms
0	1	2^{14}	7.81 ms	8.19 ms	13.33 ms	16.38 ms	17.78 ms
1	0	2^{15}	15.62 ms	16.38 ms	26.67 ms	32.77 ms	35.56 ms
1	1	2^{16}	31.25 ms	32.77 ms	53.33 ms	65.54 ms	71.11 ms
		E =	2.1 MHz	2.0 MHz	1.2288 MHz	1.0 MHz	921.6 KHz

1-520

SECTION 9
SYSTEM PROTECTION

This section describes the computer operating properly (COP) system feature and other system protection features.

9.1 COP WATCHDOG SYSTEM

The MC68HC11A8 includes a COP (computer operating properly) watchdog system to help protect against software failures. To use a COP watchdog timer, a watchdog reset sequence is executed on a regular periodic basis so that the watchdog timer is never allowed to time out.

The internal MC68HC11A8 COP function includes special control bits which permit specification of one of four time out periods and even allows the function to be disabled completely.

The NOCOP control bit, which determines whether or not a watchdog timeout causes a system reset, is implemented in an EEPROM cell in the CONFIG register. Once programmed, this bit remains set (or cleared) even when no power is applied to the MC68HC11A8, and the COP function is enabled or disabled independent of resident software. The NOCOP control bit may be overridden while in special test modes so the COP system can be inhibited from causing a hardware reset.

Two other control bits in the OPTION register select one of four timeout durations for the COP timer. The actual timeout period is dependent on the system E clock frequency, but for reference purposes, Table 9-1 shows the relationship between the CR1 and CR0 control bits and the COP timeout period for various system clock frequencies.

Table 9-1. COP Timeout Period versus CR1 and CR0

CR1	CR0	E/2^{15} Divided By	XTAL = 2^{23} Timeout −0/+15.6 ms	XTAL = 8.0 MHz Timeout −0/+16.4 ms	XTAL = 4.9152 MHz Timeout −0/+26.7 ms	XTAL = 4.0 MHz Timeout −0/+32.8 ms	XTAL = 3.6864 MHz Timeout −0/+35.6 ms
0	0	1	15.625 ms	16.384 ms	26.667 ms	32.768 ms	35.556 ms
0	1	4	62.5 ms	65.536 ms	106.67 ms	131.07 ms	142.22 ms
1	0	16	250 ms	262.14 ms	426.67 ms	524.29 ms	568.89 ms
1	1	64	1 s	1.049 s	1.707 s	2.1 s	2.276 s
		E =	2.1 MHz	2.0 MHz	1.2288 MHz	1.0 MHz	921.6 KHz

1-521

The default reset condition of CR1:CR0 is 0:0 which corresponds to the shortest timeout period.

The sequence required to reset the watchdog timer is: 1) write $55 to the COPRST register at $__03A, followed by 2) write $AA to the same address. Both writes must occur in correct order prior to timeout but, any number of instructions may be executed between the writes. The elapsed time between adjacent software reset sequences must never be greater than the COP time out period.

9.2 CONFIGURATION OPTIONS REGISTER (OPTION)

This is a special purpose 8-bit register that is used (optionally) during initialization to configure internal *system configuration* options. With the exception of bits 7, 6, and 3 (ADPU, CSEL, and CME) which may be read or written at any time, this register may be written to only once after a reset and thereafter is a read-only register. If no write is performed to this location within 64 E-clock cycles after reset, then bits 5, 4, 1, and 0 (IRQE, DLY, CR1, and CR0) will become read-only to minimize the possibility of any accidental changes to the system configuration. While in special test modes, the protection mechanism on this register is overridden and all bits in the OPTION register may be written.

Figure 9-1 and the following paragraphs describe the function of each of the bits.

B7	B6	B5	B4	B3	B2	B1	B0	
ADPU	CSEL	IRQE	DLY	CME	-	CR1	CR0	$_039

I 527

Figure 9-1. Configuration Options Register (OPTION)

Bit 7, ADPU A/D Powerup. This bit is cleared by reset and controls operation of the on-chip analog-to-digital converter. When ADPU is clear, the A/D system is powered down and conversion requests will not return meaningful information. To use the A/D system, this bit should be set.

Bit 6, CSEL A/D Clock Source Select. This bit is cleared by reset and determines the clocking source for the on-chip A/D. When this bit is zero, the MCU E clock drives the A/D system. When CSEL is one, an on-chip R-C oscillator is enabled and clocks the A/D system at about 1.5 MHz rate. When running with an E clock less than 1 MHz, CSEL must be high to program or erase the EEPROM.

Bit 5, IRQE IRQ Edge/Level Sensitive. This read/write bit is cleared at reset. When it is clear, the $\overline{IRQ}$ pin is configured for level sensitive wired-OR operation (low level) and when it is set, the IRQ is configured for edge-only sensitivity (falling edges) on the $\overline{IRQ}$ pin.

Bit 4, DLY STOP Exit Turn-On Delay. This bit may only be written under special circumstances as described above. This bit is set to one during reset and controls whether or not a relatively long turn-on delay will be imposed before processing can resume after a STOP period. If an external clock source is supplied this delay can be inhibited so that processing can resume within a few cycles of a wake up from STOP mode. When DLY is a one, delay is imposed to allow oscillator stabilization and when DLY is a zero, this delay is bypassed.

Bit 3, CME Clock Monitor Enable. This control bit may be read or written at any time and controls whether or not the internal clock monitor circuit will trigger a reset sequence when a slow or absent system clock is detected. When it is clear, the clock monitor circuit is disabled and when it is set, the clock monitor circuit is enabled. Systems operating at or below 200 KHz should not use the clock monitor function. Reset clears the CME bit.

Bit 2 _ Not Implemented. Reads as a logic zero.

Bit 1, CR1 COP Timer Rate Selects. Refer to Table 9-1 for the relationship between CR1:CR0 and the
Bit 0, CR0 COP timeout period.

9.3 CLOCK MONITOR

The clock monitor function is enabled by the CME control bit in the OPTION register. When CME is zero, the function is disabled and when CME is a one, the clock monitor function detects the absence of an E clock for more than a certain period of time. The timeout period is dependent on processing parameters and will be between 5 and 100 microseconds. This means that an E-clock rate of 200 kHz or more will never cause a clock monitor failure and an E-clock rate of 10 kHz or less will definitely cause a clock monitor failure. This implies that systems operating near or below an E-clock rate of 200 kHz should not use the clock monitor function.

Upon detection of a slow or absent clock, the clock monitor circuit (if enabled by CME = 1) will cause a system reset to be generated. This reset is issued to the external system via the bidirectional $\overline{\text{RESET}}$ pin.

Special considerations are needed when using a STOP function and clock monitor in the same system. Since the STOP function causes the clocks to be halted, the clock monitor function will generate a reset sequence if it is enabled at the time the STOP mode is entered.

SECTION 10
ADDRESSING MODES AND INSTRUCTION SET

This section provides a description of the addressing modes and instruction set.

10.1 ADDRESSING MODES

Six addressing modes can be used to reference memory; they include: immediate, direct, extended, indexed (with either of two 16-bit index registers and an 8-bit offset), inherent and relative. Some instructions require an additional byte before the opcode to accomodate a multi-page opcode map; this byte is called a prebyte.

The following paragraphs provide a description of each addressing mode plus a discussion of the prebyte. In these descriptions the term effective address is used to indicate the address in memory from which the argument is fetched or stored, or from which execution is to proceed.

10.1.1 Immediate Addressing

In the immediate addressing mode, the actual argument is contained in the byte(s) immediately following the instruction, where the number of bytes matches the size of the register. These are two, three, or four (if prebyte is required) byte instructions.

10.1.2 Direct Addressing

In the direct addressing mode, the least significant byte of the operand address is contained in a single byte following the opcode and the most significant byte is assumed to be $00. Direct addressing allows the user to access $0000 through $00FF using two byte instructions and execution time is reduced by eliminating the additional memory access. In most applications, this 256-byte area is reserved for frequently referenced data. In the MC68HC11A8, software can configure the memory map so that internal RAM, and/or internal registers, or external memory space can occupy these addresses.

10.1.3 Extended Addressing

In the extended addressing mode, the second and third bytes (following the opcode) contain the absolute address of the operand. These are three or four (if prebyte is required) byte instructions: one or two for the opcode, and two for the effective address.

10.1.4 Indexed Addressing

In the indexed addressing mode, one of the index registers (X or Y) is used in calculating the effective address. In this case, the effective address is variable and depends on two factors: 1) the current contents of the index register (X or Y) being used, and 2) the 8-bit unsigned offset contained in the instruction. This addressing mode allows referencing any memory location in the 64K byte address space. These are usually two or three (if prebyte is required) byte instructions, the opcode plus the 8-bit offset.

10.1.5 Inherent Addressing

In the inherent addressing mode, all of the information to execute the instruction is contained in the opcode. The operands (if any) are registers and no memory reference is required. These are usually one or two byte instructions.

10.1.6 Relative Addressing

The relative addressing mode is used for branch instructions. If the branch condition is true, the contents of the 8-bit signed byte following the opcode (the offset) is added to the contents of the program counter to form the effective branch address; otherwise, control proceeds to the next instruction. These are usually two byte instructions.

10.1.7 Prebyte

In order to expand the number of instructions used in the MC68HC11A8, a prebyte instruction has been added to certain instructions. The instructions affected are usually associated with index register Y. The opcode instructions which do not require a prebyte could be considered as page 1 of the overall opcode map. The remaining opcodes could be considered as pages 2, 3, and 4 of the opcode map and would require a prebyte; $18 for page 2, $1A for page 3, and $CD for page 4.

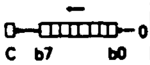
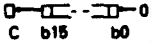
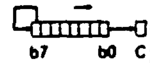
10.2 INSTRUCTION SET

The central processing unit (CPU) in the MC68HC11A8 is basically a proper extension of the MC6801 CPU. In addition to its ability to execute all M6800 and M6801 instructions, the MC68HC11A8 CPU has a paged operation code (opcode) map with a total of 91 new opcodes. Major functional additions include a second 16-bit index register (Y register), two types of 16-by-16 divide instructions, STOP and WAIT instructions, and bit manipulation instructions.

Table 10-1 shows all MC68HC11A8 instructions in all possible addressing modes. For each instruction, the operand construction is shown as well as the total number of machine code bytes and execution time in CPU E-clock cycles. Notes are provided at the end of Table 10-1 which explain the letters in the Operand and Execution Time columns for some instructions. Definition of "Special Ops" found in the Boolean Expression column is found in Figure 10-1.

Tables 10-2 through 10-8 provide a detailed description of the information present on the address bus, data bus, and the read/write (R/$\overline{W}$) line during each cycle of each instruction. The information is useful in comparing actual with expected results during debug of both software and hardware as the program is executed. The information is categorized in groups according to addressing mode and number of cycles per instruction. In general, instructions with the same addressing mode and number of cycles execute in the same manner. Exceptions are indicated in the table.

Table 10-1. MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 1 of 7)

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes							
				Opcode	Operand(s)				S	X	H	I	N	Z	V	C
ABA	Add Accumulators	A + B → A	INH	1B		1	2	2-1	-	-	↕	-	↕	↕	↕	↕
ABX	Add B to X	IX + 00:B → IX	INH	3A		1	3	2-2	-	-	-	-	-	-	-	-
ABY	Add B to Y	IY + 00:B → IY	INH	18 3A		2	4	2-4	-	-	-	-	-	-	-	-
ADCA (opr)	Add with Carry to A	A + M + C → A	A IMM	89	ii	2	2	3-1	-	-	↕	-	↕	↕	↕	↕
			A DIR	99	dd	2	3	4-1								
			A EXT	B9	hh ll	3	4	5-2								
			A IND,X	A9	ff	2	4	6-2								
			A IND,Y	18 A9	ff	3	5	7-2								
ADCB (opr)	Add with Carry to B	B + M + C → B	B IMM	C9	ii	2	2	3-1	-	-	↕	-	↕	↕	↕	↕
			B DIR	D9	dd	2	3	4-1								
			B EXT	F9	hh ll	3	4	5-2								
			B IND,X	E9	ff	2	4	6-2								
			B IND,Y	18 E9	ff	3	5	7-2								
ADDA (opr)	Add Memory to A	A + M → A	A IMM	8B	ii	2	2	3-1	-	-	↕	-	↕	↕	↕	↕
			A DIR	9B	dd	2	3	4-1								
			A EXT	BB	hh ll	3	4	5-2								
			A IND,X	AB	ff	2	4	6-2								
			A IND,Y	18 AB	ff	3	5	7-2								
ADDB (opr)	Add Memory to B	B + M → B	B IMM	CB	ii	2	2	3-1	-	-	↕	-	↕	↕	↕	↕
			B DIR	DB	dd	2	3	4-1								
			B EXT	FB	hh ll	3	4	5-2								
			B IND,X	EB	ff	2	4	6-2								
			B IND,Y	18 EB	ff	3	5	7-2								
ADDD (opr)	Add 16-Bit to D	D + M:M + 1 → D	IMM	C3	jj kk	3	4	3-3	-	-	-	-	↕	↕	↕	↕
			DIR	D3	dd	2	5	4-7								
			EXT	F3	hh ll	3	6	5-10								
			IND,X	E3	ff	*2*	*6*	*6-10*								
			IND, Y	18 E3	ff	3	7	7-8								
ANDA (opr)	AND A with Memory	A•M → A	A IMM	84	ii	2	2	3-1	-	-	-	-	↕	↕	0	-
			A DIR	94	dd	2	3	4-1								
			A EXT	B4	hh ll	3	4	5-2								
			A IND,X	A4	ff	2	4	6-2								
			A IND,Y	18 A4	ff	3	5	7-2								
ANDB (opr)	AND B with Memory	B•M → B	B IMM	C4	ii	2	2	3-1	-	-	-	-	↕	↕	0	-
			B DIR	D4	dd	2	3	4-1								
			B EXT	F4	hh ll	3	4	5-2								
			B IND,X	E4	ff	2	4	6-2								
			B IND,Y	18 E4	ff	3	5	7-2								
ASL (opr)	Arithmetic Shift Left		EXT	78	hh ll	3	6	5-8	-	-	-	-	↕	↕	↕	↕
			IND,X	68	ff	2	6	6-3								
			IND,Y	18 68	ff	3	7	7-3								
ASLA			A INH	48		1	2	2-1								
ASLB			B INH	58		1	2	2-1								
ASLD	Arithmetic Shift Left Double		INH	05		1	3	2-2	-	-	-	-	↕	↕	↕	↕
ASR (opr)	Arithmetic Shift Right		EXT	77	hh ll	3	6	5-8	-	-	-	-	↕	↕	↕	↕
			IND,X	67	ff	2	6	6-3								
			IND,Y	18 67	ff	3	7	7-3								
ASRA	-		A INH	47		1	2	2-1								
ASRB			B INH	57		1	2	2-1								
BCC (rel)	Branch if Carry Clear	? C = 0	REL	24	rr	2	3	8-1	-	-	-	-	-	-	-	-
BCLR (opr) (msk)	Clear Bit(s)	M•($\overline{mm}$) → M	DIR	15	dd mm	3	6	4-10	-	-	-	-	↕	↕	0	-
			IND,X	1D	ff mm	3	7	6-13								
			IND,Y	18 1D	ff mm	4	8	7-10								
BCS (rel)	Branch if Carry Set	? C = 1	REL	25	rr	2	3	8-1	-	-	-	-	-	-	-	-
BEQ (rel)	Branch if = Zero	? Z = 1	REL	27	rr	2	3	8-1	-	-	-	-	-	-	-	-

*Cycle-by-cycle number provides a reference to Tables 10-2 through 10-8 which detail cycle-by-cycle operation.
Example: Table 10-1 Cycle-by-Cycle column reference number 2-4 equals Table 10-2 line item 2-4.

1 5234

Table 10-1. MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 2 of 7)

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes							
				Opcode	Operand(s)				S	X	H	I	N	Z	V	C
BGE (rel)	Branch if ≥ Zero	? N ⊕ V = 0	REL	2C	rr	2	3	8-1	·	·	·	·	·	·	·	·
BGT (rel)	Branch if > Zero	? Z + (N ⊕ V) = 0	REL	2E	rr	2	3	8-1	·	·	·	·	·	·	·	·
BHI (rel)	Branch if Higher	? C + Z = 0	REL	22	rr	2	3	8-1	·	·	·	·	·	·	·	·
BHS (rel)	Branch if Higher or Same	? C = 0	REL	24	rr	2	3	8-1	·	·	·	·	·	·	·	·
BITA (opr)	Bit(s) Test A with Memory	A•M	A IMM A DIR A EXT A IND,X A IND,Y	85 95 B5 A5 18 A5	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	·	·	·	·	↕	↕	0	·
BITB (opr)	Bit(s) Test B with Memory	B•M	B IMM B DIR B EXT B IND,X B IND,Y	C5 D5 F5 E5 18 E5	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	·	·	·	·	↕	↕	0	·
BLE (rel)	Branch if ≤ Zero	? Z + (N ⊕ V) = 1	REL	2F	rr	2	3	8-1	·	·	·	·	·	·	·	·
BLO (rel)	Branch if Lower	? C = 1	REL	25	rr	2	3	8-1	·	·	·	·	·	·	·	·
BLS (rel)	Branch if Lower or Same	? C + Z = 1	REL	23	rr	2	3	8-1	·	·	·	·	·	·	·	·
BLT (rel)	Branch If < Zero	? N ⊕ V = 1	REL	2D	rr	2	3	8-1	·	·	·	·	·	·	·	·
BMI (rel)	Branch if Minus	? N = 1	REL	2B	rr	2	3	8-1	·	·	·	·	·	·	·	·
BNE (rel)	Branch if Not = Zero	? Z = 0	REL	26	rr	2	3	8-1	·	·	·	·	·	·	·	·
BPL (rel)	Branch if Plus	? N = 0	REL	2A	rr	2	3	8-1	·	·	·	·	·	·	·	·
BRA (rel)	Branch Always	? 1 = 1	REL	20	rr	2	3	8-1	·	·	·	·	·	·	·	·
BRCLR(opr) (msk) (rel)	Branch if Bit(s) Clear	? M• mm = 0	DIR IND,X IND,Y	13 1F 18 1F	dd mm rr ff mm rr ff mm rr	4 4 5	6 7 8	4-11 6-14 7-11	·	·	·	·	·	·	·	·
BRN (rel)	Branch Never	? 1 = 0	REL	21	rr	2	3	8-1	·	·	·	·	·	·	·	·
BRSET(opr) (msk) (rel)	Branch if Bit(s) Set	? ($\overline{M}$)•mm = 0	DIR IND,X IND,Y	12 1E 18 1E	dd mm rr ff mm rr ff mm rr	4 4 5	6 7 8	4-11 6-14 7-11	·	·	·	·	·	·	·	·
BSET(opr) (msk)	Set Bit(s)	M + mm → M	DIR IND,X IND,Y	14 1C 18 1C	dd mm ff mm ff mm	3 3 4	6 7 8	4-10 6-13 7-10	·	·	·	·	↕	↕	0	·
BSR (rel)	Branch to Subroutine	See Special Ops	REL	8D	rr	2	6	8-2	·	·	·	·	·	·	·	·
BVC (rel)	Branch if Overflow Clear	? V = 0	REL	28	rr	2	3	8-1	·	·	·	·	·	·	·	·
BVS (rel)	Branch if Overflow Set	? V = 1	REL	29	rr	2	3	8-1	·	·	·	·	·	·	·	·
CBA	Compare A to B	A-B	INH	11		1	2	2-1	·	·	·	·	↕	↕	↕	↕
CLC	Clear Carry Bit	0 → C	INH	0C		1	2	2-1	·	·	·	·	·	·	·	0
CLI	Clear Interrupt Mask	0 → I	INH	0E		1	2	2-1	·	·	·	0	·	·	·	·
CLR (opr)	Clear Memory Byte	0 → M	EXT IND,X IND,Y	7F 6F 18 6F	hh ll ff ff	3 2 3	6 6 7	5-8 6-3 7-3	·	·	·	·	0	1	0	0
CLRA	Clear Accumulator A	0 → A	A INH	4F		1	2	2-1	·	·	·	·	0	1	0	0
CLRB	Clear Accumulator B	0 → B	B INH	5F		1	2	2-1	·	·	·	·	0	1	0	0
CLV	Clear Overflow Flag	0 → V	INH	0A		1	2	2-1	·	·	·	·	·	·	0	·
CMPA (opr)	Compare A to Memory —	A − M	A IMM A DIR A EXT A IND,X A IND,Y	81 91 B1 A1 18 A1	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	·	·	·	·	↕	↕	↕	↕

*Cycle-by-cycle number provides a reference to Tables 10-2 through 10-8 which detail cycle-by-cycle operation.
Example: Table 10-1 Cycle-by-Cycle column reference number 2-4 equals Table 10-2 line item 2-4.

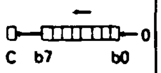
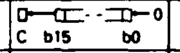
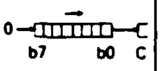
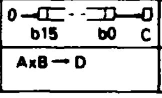
1-5238

Table 10-1. MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 3 of 7)

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes							
				Opcode	Operand(s)				S	X	H	I	N	Z	V	C
CMPB (opr)	Compare B to Memory	B – M	B IMM	C1	ii	2	2	3-1	-	-	-	-	↕	↕	↕	↕
			B DIR	D1	dd	2	3	4-1								
			B EXT	F1	hh ll	3	4	5-2								
			B IND,X	E1	ff	2	4	6-2								
			B IND,Y	18 E1	ff	3	5	7-2								
COM (opr)	1's Complement Memory Byte	$FF – M → M	EXT	73	hh ll	3	6	5-8	-	-	-	-	↕	↕	0	1
			IND,X	63	ff	2	6	6-3								
			IND,Y	18 63	ff	3	7	7-3								
COMA	1's Complement A	$FF - A → A	A INH	43		1	2	2-1	-	-	-	-	↕	↕	0	1
COMB	1's Complement B	$FF – B → B	B INH	53		1	2	2-1	-	-	-	-	↕	↕	0	1
CPD (opr)	Compare D to Memory 16-Bit	D – M:M + 1	IMM	1A 83	jj kk	4	5	3-5	-	-	-	-	↕	↕	↕	↕
			DIR	1A 93	dd	3	6	4-9								
			EXT	1A B3	hh ll	4	7	5-11								
			IND,X	1A A3	ff	3	7	6-11								
			IND,Y	CD A3	ff	3	7	7-8								
CPX (opr)	Compare X to Memory 16-Bit	IX – M:M + 1	IMM	8C	jj kk	3	4	3-3	-	-	-	-	↕	↕	↕	↕
			DIR	9C	dd	2	5	4-7								
			EXT	BC	hh ll	3	6	5-10								
			IND,X	AC	ff	2	6	6-10								
			IND,Y	CD AC	ff	3	7	7-8								
CPY (opr)	Compare Y to Memory 16-Bit	IY – M:M + 1	IMM	18 8C	jj kk	4	5	3-5	-	-	-	-	↕	↕	↕	↕
			DIR	18 9C	dd	3	6	4-9								
			EXT	18 BC	hh ll	4	7	5-11								
			IND,X	1A AC	ff	3	7	6-11								
			IND,Y	18 AC	ff	3	7	7-8								
DAA	Decimal Adjust A	Adjust Sum to BCD	INH	19		1	2	2-1	-	-	-	-	↕	↕	↕	↕
DEC (opr)	Decrement Memory Byte	M – 1 → M	EXT	7A	hh ll	3	6	5-8	-	-	-	-	↕	↕	↕	
			IND,X	6A	ff	2	6	6-3								
			IND,Y	18 6A	ff	3	7	7-3								
DECA	Decrement Accumulator A	A – 1 → A	A INH	4A		1	2	2-1	-	-	-	-	↕	↕	↕	
DECB	Decrement Accumulator B	B – 1 → B	B INH	5A		1	2	2-1	-	-	-	-	↕	↕	↕	
DES	Decrement Stack Pointer	SP - 1 → SP	INH	34		1	3	2-3	-	-	-	-	-	-	-	-
DEX	Decrement Index Register X	IX – 1 → IX	INH	09		1	3	2-2	-	-	-	-	-	↕	-	-
DEY	Decrement Index Register Y	IY – 1 → IY	INH	18 09		2	4	2-4	-	-	-	-	-	↕	-	-
EORA (opr)	Exclusive OR A with Memory	A ⊕ M → A	A IMM	88	ii	2	2	3-1	-	-	-	-	↕	↕	0	-
			A DIR	98	dd	2	3	4-1								
			A EXT	B8	hh ll	3	4	5-2								
			A IND,X	A8	ff	2	4	6-2								
			A IND,Y	18 A8	ff	3	5	7-2								
EORB (opr)	Exclusive OR B with Memory	B ⊕ M → B	B IMM	C8	ii	2	2	3-1	-	-	-	-	↕	↕	0	
			B DIR	D8	dd	2	3	4-1								
			B EXT	F8	hh ll	3	4	5-2								
			B IND,X	E8	ff	2	4	6-2								
			B IND,Y	18 E8	ff	3	5	7-2								
FDIV	Fractional Divide 16 by 16	D/IX → IX; r → D	INH	03		1	41	2-17	-	-	-	-	-	↕	↕	↕
IDIV	Integer Divide 16 by 16	D/IX → IX; r → D	INH	02		1	41	2-17	-	-	-	-	-	↕	0	↕
INC (opr)	Increment Memory Byte	M + 1 → M	EXT	7C	hh ll	3	6	5-8	-	-	-	-	↕	↕	↕	
			IND,X	6C	ff	2	6	6-3								
			IND,Y	18 6C	ff	3	7	7-3								
INCA	Increment Accumulator A	A + 1 → A	A INH	4C		1	2	2-1	-	-	-	-	↕	↕	↕	-
INCB	Increment Accumulator B	B + 1 → B	B INH	5C		1	2	2-1	-	-	-	-	↕	↕	↕	-
INS	Increment Stack Pointer	SP + 1 → SP	INH	31		1	3	2-3	-	-	-	-	-	-	-	-

*Cycle-by-cycle number provides a reference to Tables 10-2 through 10-8 which detail cycle-by-cycle operation.
Example: Table 10-1 Cycle-by-Cycle column reference number 2-4 equals Table 10-2 line item 2-4.

Table 10-1. MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 4 of 7)

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes							
				Opcode	Operand(s)				S	X	H	I	N	Z	V	C
INX	Increment Index Register X	IX + 1 → IX	INH	08		1	3	2-2	-	-	-	-	-	↕	-	-
INY	Increment Index Register Y	IY + 1 → IY	INH	18 08		2	4	2-4	-	-	-	-	-	↕	-	-
JHP (opr)	Jump	See Special Ops	EXT	7E	hh ll	3	3	5-1	-	-	-	-	-	-	-	-
			IND,X	6E	ff	2	3	6-1								
			IND,Y	18 6E	ff	3	4	7-1								
JSR (Opr)	Jump to Subroutine	See Special Ops	DIR	9D	dd	2	5	4-8	-	-	-	-	-	-	-	-
			EXT	BD	hh ll	3	6	5-12								
			IND,X	AD	ff	2	6	6-12								
			IND,Y	18 AD	ff	3	7	7-9								
LDAA (opr)	Load Accumulator A	M → A	A IMM	86	ii	2	2	3-1	-	-	-	-	↕	↕	0	-
			A DIR	96	dd	2	3	4-1								
			A EXT	B6	hh ll	3	4	5-2								
			A IND,X	A6	ff	2	4	6-2								
			A IND,Y	18 A6	ff	3	5	7-2								
LDAB (opr)	Load Accumulator B	M → B	B IMM	C6	ii	2	2	3-1	-	-	-	-	↕	↕	0	-
			B DIR	D6	dd	2	3	4-1								
			B EXT	F6	hh ll	3	4	5-2								
			B IND,X	E6	ff	2	4	6-2								
			B IND,Y	18 E6	ff	3	5	7-2								
LDD (opr)	Load Double Accumulator D	M → A,M + 1 → B	IMM	CC	jj kk	3	3	3-2	-	-	-	-	↕	↕	0	-
			DIR	DC	dd	2	4	4-3								
			EXT	FC	hh ll	3	5	5-4								
			IND,X	EC	ff	2	5	6-6								
			IND,Y	18 EC	ff	3	6	7-6								
LDS (opr)	Load Stack Pointer	M:M + 1 → SP	IMM	8E	jj kk	3	3	3-2	-	-	-	-	↕	↕	0	-
			DIR	9E	dd	2	4	4-3								
			EXT	BE	hh ll	3	5	5-4								
			IND,X	AE	ff	2	5	6-6								
			IND,Y	18 AE	ff	3	6	7-6								
LDX (opr)	Load Index Register X	M:M + 1 → IX	IMM	CE	jj kk	3	3	3-2	-	-	-	-	↕	↕	0	-
			DIR	DE	dd	2	4	4-3								
			EXT	FE	hh ll	3	5	5-4								
			IND,X	EE	ff	2	5	6-6								
			IND,Y	CD EE	ff	3	6	7-6								
LDY (opr)	Load Index Register Y	M:M + 1 → IY	IMM	18 CE	jj kk	4	4	3-4	-	-	-	-	↕	↕	0	-
			DIR	18 DE	dd	3	5	4-5								
			EXT	18 FE	hh ll	4	6	5-6								
			IND,X	1A EE	ff	3	6	6-7								
			IND,Y	18 EE	ff	3	6	7-6								
LSL (opr)	Logical Shift Left		EXT	78	hh ll	3	6	5-8	-	-	-	-	↕	↕	↕	↕
			IND,X	68	ff	2	6	6-3								
			IND,Y	18 68	ff	3	7	7-3								
LSLA			A INH	48		1	2	2-1								
LSLB			B INH	58		1	2	2-1								
LSLD	Logical Shift Left Double		INH	05		1	3	2-2	-	-	-	-	↕	↕	↕	↕
LSR (opr)	Logical Shift Right		EXT	74	hh ll	3	6	5-8	-	-	-	-	0	↕	↕	↕
			IND,X	64	ff	2	6	6-3								
			IND,Y	18 64	ff	3	7	7-3								
LSRA			A INH	44		1	2	2-1								
LSRB			B INH	54		1	2	2-1								
LSRD	Logical Shift Right Double		INH	04		1	3	2-2	-	-	-	-	0	↕	↕	↕
MUL	Multiply 8 by 8	AxB → D	INH	3D		1	10	2-13	-	-	-	-	-	-	-	↕

*Cycle-by-cycle number provides a reference to Tables 10-2 through 10-8 which detail cycle-by-cycle operation.
Example: Table 10-1 Cycle-by-Cycle column reference number 2-4 equals Table 10-2 line item 2-4.

1 5230

Table 10-1. MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 5 of 7)

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes							
				Opcode	Operand(s)				S	X	H	I	N	Z	V	C
NEG (opr)	2's Complement Memory Byte	0 – M → M	EXT	70	hh ll	3	6	5-8	·	·	·	·	↕	↕	↕	↕
			IND,X	60	ff	2	6	6-3								
			IND,Y	18 60	ff	3	7	7-3								
NEGA	2's Complement A	0 – A → A	A INH	40		1	2	2-1	·	·	·	·	↕	↕	↕	↕
NEGB	2's Complement B	0 – B → B	B INH	50		1	2	2-1	·	·	·	·	↕	↕	↕	↕
NOP	No Operation	No Operation	INH	01		1	2	2-1	·	·	·	·	·	·	·	·
ORAA (opr)	OR Accumulator A (Inclusive)	A + M → A	A IMM	8A	ii	2	2	3-1	·	·	·	·	↕	↕	0	·
			A DIR	9A	dd	2	3	4-1								
			A EXT	BA	hh ll	3	4	5-2								
			A IND,X	AA	ff	2	4	6-2								
			A IND,Y	18 AA	ff	3	5	7-2								
ORAB (opr)	OR Accumulator B (Inclusive)	B + M → B	B IMM	CA	ii	2	2	3-1	·	·	·	·	↕	↕	0	·
			B DIR	DA	dd	2	3	4-1								
			B EXT	FA	hh ll	3	4	5-2								
			B IND,X	EA	ff	2	4	6-2								
			B IND,Y	18 EA	ff	3	5	7-2								
PSHA	Push A onto Stack	A → Stk, SP = SP – 1	A INH	36		1	3	2-6	·	·	·	·	·	·	·	·
PSHB	Push B onto Stack	B → Stk, SP = SP – 1	B INH	37		1	3	2-6	·	·	·	·	·	·	·	·
PSHX	Push X onto Stack (Lo First)	IX → Stk, SP = SP – 2	INH	3C		1	4	2-7	·	·	·	·	·	·	·	·
PSHY	Push Y onto Stack (Lo First)	IY → Stk, SP = SP – 2	INH	18 3C		2	5	2-8	·	·	·	·	·	·	·	·
PULA	Pull A from Stack	SP = SP + 1, A ← Stk	A INH	32		1	4	2-9	·	·	·	·	·	·	·	·
PULB	Pull B from Stack	SP = SP + 1, B ← Stk	B INH	33		1	4	2-9	·	·	·	·	·	·	·	·
PULX	Pull X from Stack (Hi First)	SP = SP + 2, IX ← Stk	INH	38		1	5	2-10	·	·	·	·	·	·	·	·
PULY	Pull Y from Stack (Hi First)	SP = SP + 2, IY ← Stk	INH	18 38		2	6	2-11	·	·	·	·	·	·	·	·
ROL (opr)	Rotate Left	□—□□□□□□□□—□ C b7 ← b0 C	EXT	79	hh ll	3	6	5-8	·	·	·	·	↕	↕	↕	↕
			IND,X	69	ff	2	6	6-3								
			IND,Y	18 69	ff	3	7	7-3								
ROLA			A INH	49		1	2	2-1								
ROLB			B INH	59		1	2	2-1								
ROR (opr)	Rotate Right	□—□□□□□□□□—□ C b7 → b0 C	EXT	76	hh ll	3	6	5-8	·	·	·	·	↕	↕	↕	↕
			IND,X	66	ff	2	6	6-3								
			IND,Y	18 66	ff	3	7	7-3								
RORA			A INH	46		1	2	2-1								
RORB			B INH	56		1	2	2-1								
RTI	Return from Interrupt	See Special Ops	INH	3B		1	12	2-14	↕	↓	↕	↕	↕	↕	↕	↕
RTS	Return from Subroutine	See Special Ops	INH	39		1	5	2-12	·	·	·	·	·	·	·	·
SBA	Subtract B from A	A – B → A	INH	10		1	2	2-1	·	·	·	·	↕	↕	↕	↕
SBCA (opr)	Subtract with Carry from A	A – M – C → A	A IMM	82	ii	2	2	3-1	·	·	·	·	↕	↕	↕	↕
			A DIR	92	dd	2	3	4-1								
			A EXT	B2	hh ll	3	4	5-2								
			A IND,X	A2	ff	2	4	6-2								
			A IND,Y	18 A2	ff	3	5	7-2								
SBCB (opr)	Subtract from Carry from B	B – M – C → B	B IMM	C2	ii	2	2	3-1	·	·	·	·	↕	↕	↕	↕
			B DIR	D2	dd	2	3	4-1								
			B EXT	F2	hh ll	3	4	5-2								
			B IND,X	E2	ff	2	4	6-2								
			B IND,Y	18 E2	ff	3	5	7-2								
SEC	Set Carry	1 → C	INH	OD		1	2	2-1	·	·	·	·	·	·	·	1
SEI	Set Interrupt Mask	1 → I	INH	OF		1	2	2-1	·	·	·	1	·	·	·	·
SEV	Set Overflow Flag	1 → V	INH	OB		1	2	2-1	·	·	·	·	·	·	1	·

*Cycle-by-cycle number provides a reference to Tables 10-2 through 10-8 which detail cycle-by-cycle operation.
Example: Table 10-1 Cycle-by-Cycle column reference number 2-4 equals Table 10-2 line item 2-4.

r 523f

Table 10-1. MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 6 of 7)

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes							
				Opcode	Operand(s)				S	X	H	I	N	Z	V	C
STAA (opr)	Store Accumulator A	A → M	A DIR	97	dd	2	3	4-2	·	·	·	·	↕	↕	0	·
			A EXT	B7	hh ll	3	4	5-3								
			A IND,X	A7	ff	2	4	6-5								
			A IND,Y	18 A7	ff	3	5	7-5								
STAB (opr)	Store Accumulator B	B → M	B DIR	D7	dd	2	3	4-2	·	·	·	·	↕	↕	0	·
			B EXT	F7	hh ll	3	4	5-3								
			B IND,X	E7	ff	2	4	6-5								
			B IND,Y	18 E7	ff	3	5	7-5								
STD (opr)	Store Accumulator D	A → M, B → M + 1	DIR	DD	dd	2	4	4-4	·	·	·	·	↕	↕	0	·
			EXT	FD	hh ll	3	5	5-5								
			IND,X	ED	ff	2	5	6-8								
			IND,Y	18 ED	ff	3	6	7-7								
STOP	Stop Internal Clocks		INH	CF		1	2	2-1	·	·	·	·	·	·	·	·
STS (opr)	Store Stack Pointer	SP → M:M + 1	DIR	9F	dd	2	4	4-4	·	·	·	·	↕	↕	0	·
			EXT	BF	hh ll	3	5	5-5								
			IND,X	AF	ff	2	5	6-8								
			IND,Y	18 AF	ff	3	6	7-7								
STX (opr)	Store Index Register X	IX → M:M + 1	DIR	DF	dd	2	4	4-4	·	·	·	·	↕	↕	0	·
			EXT	FF	hh ll	3	5	5-5								
			IND,X	EF	ff	2	5	6-8								
			IND,Y	CD EF	ff	3	6	7-7								
STY (opr)	Store Index Register Y	IY → M:M + 1	DIR	18 DF	dd	3	5	4-6	·	·	·	·	↕	↕	0	·
			EXT	18 FF	hh ll	4	6	5-7								
			IND,X	1A EF	ff	3	6	6-9								
			IND,Y	18 EF	ff	3	6	7-7								
SUBA (opr)	Subtract Memory from A	A - M → A	A IMM	80	ii	2	2	3-1	·	·	·	·	↕	↕	↕	↕
			A DIR	90	dd	2	3	4-1								
			A EXT	B0	hh ll	3	4	5-2								
			A IND,X	A0	ff	2	4	6-2								
			A IND,Y	18 A0	ff	3	5	7-2								
SUBB (opr)	Subtract Memory from B	B - M → B	B IMM	C0	ii	2	2	3-1	·	·	·	·	↕	↕	↕	↕
			B DIR	D0	dd	2	3	4-1								
			B EXT	F0	hh ll	3	4	5-2								
			B IND,X	E0	ff	2	4	6-2								
			B IND,Y	18 E0	ff	3	5	7-2								
SUBD (opr)	Subtract Memory from D	D - M:M + 1 → D	IMM	83	jj kk	3	4	3-3	·	·	·	·	↕	↕	↕	↕
			DIR	93	dd	2	5	4-7								
			EXT	B3	hh ll	3	6	5-10								
			IND,X	A3	ff	2	6	6-10								
			IND,Y	18 A3	ff	3	7	7-8								
SWI	Software Interrupt	See Special Ops	INH	3F		1	14	2-15	·	·	·	1	·	·	·	·
TAB	Transfer A to B	A → B	INH	16		1	2	2-1	·	·	·	·	↕	↕	0	·
TAP	Transfer A to CC Register	A → CCR	INH	06		1	2	2-1	↕	↓	↕	↕	↕	↕	↕	↕
TBA	Transfer B to A	B → A	INH	17		1	2	2-1	·	·	·	·	↕	↕	0	·
TEST	TEST (Only in Test Modes)	Address Bus Counts	INH	00		1	**	2-20	·	·	·	·	·	·	·	·
TPA	Transfer CC Register to A	CCR → A	INH	07		1	2	2-1	·	·	·	·	·	·	·	·
TST (opr)	Test for Zero or Minus	M - 0	EXT	7D	hh ll	3	6	5-9	·	·	·	·	↕	↕	0	0
			IND,X	6D	ff	2	6	6-4								
			IND,Y	18 6D	ff	3	7	7-4								
TSTA	—	A - 0	A INH	4D		1	2	2-1	·	·	·	·	↕	↕	0	0
TSTB		B - 0	B INH	5D		1	2	2-1	·	·	·	·	↕	↕	0	0
TSX	Transfer Stack Pointer to X	SP + 1 → IX	INH	30		1	3	2-3	·	·	·	·	·	·	·	·
TSY	Transfer Stack Pointer to Y	SP + 1 → IY	INH	18 30		2	4	2-5	·	·	·	·	·	·	·	·

*Cycle-by-cycle number provides a reference to Tables 10-2 through 10-8 which detail cycle-by-cycle operation.
Example: Table 10-1 Cycle-by-Cycle column reference number 2-4 equals Table 10-2 line item 2-4.

1 523F

Table 10-1. MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 7 of 7)

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes							
				Opcode	Operand(s)				S	X	H	I	N	Z	V	C
TXS	Transfer X to Stack Pointer	IX − 1 → SP	INH	35		1	3	2-2	-	-	-	-	-	-	-	-
TYS	Transfer Y to Stack Pointer	IY − 1 → SP	INH	18 35		2	4	2-4	-	-	-	-	-	-	-	-
WAI	Wait for Interrupt	Stack Regs & WAIT	INH	3E		2	***	2-18	-	-	-	-	-	-	-	-
XGDX	Exchange D with X	IX → D, D → IX	INH	8F		1	3	2-2	-	-	-	-	-	-	-	-
XGDY	Exchange D with Y	IY → D, D → IY	INH	18 8F		2	4	2-4	-	-	-	-	-	-	-	-

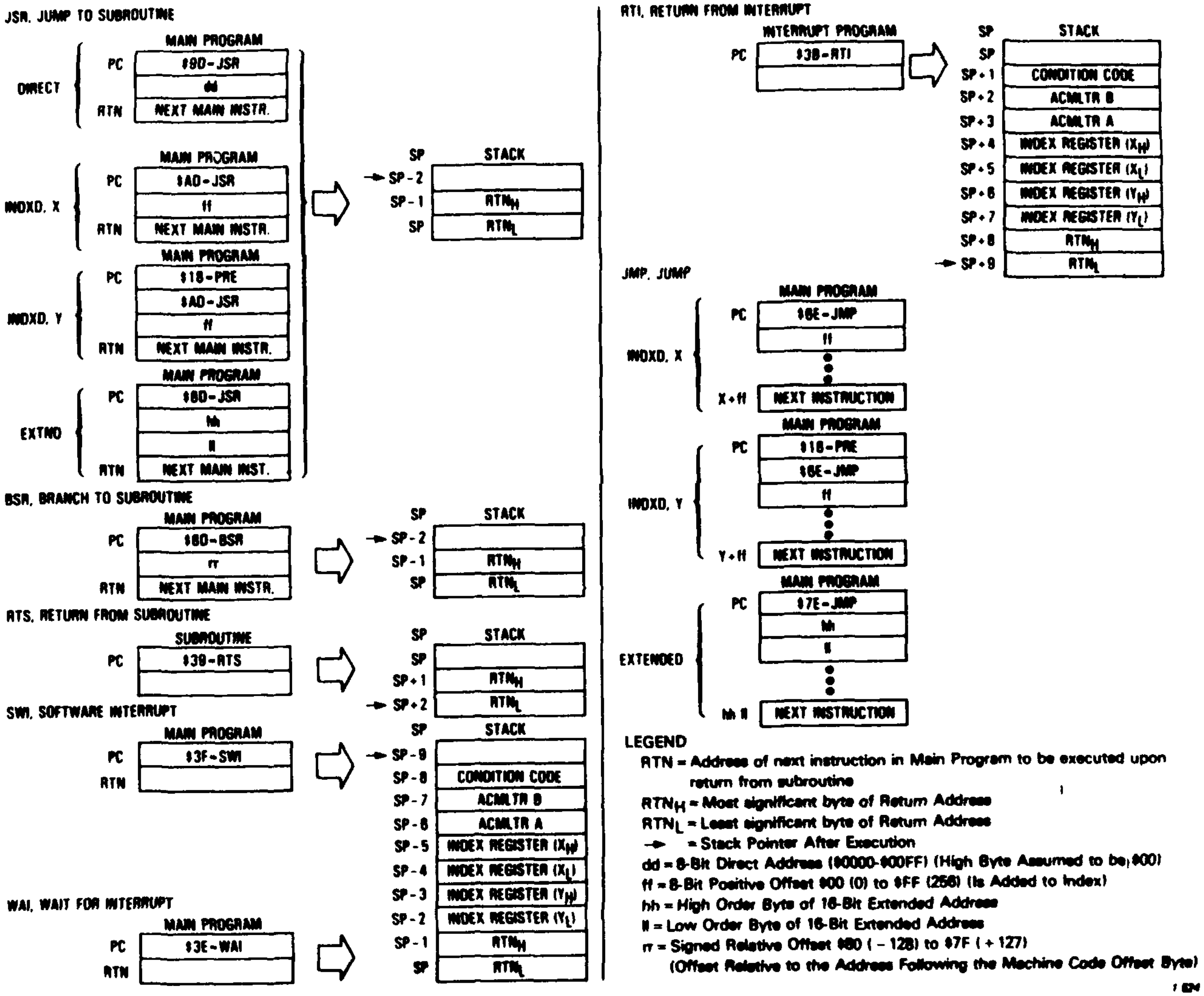
*Cycle-by-cycle number provides a reference to Tables 10-2 through 10-8 which detail cycle-by-cycle operation.
Example: Table 10-1 Cycle-by-Cycle column reference number 2-4 equals Table 10-2 line item 2-4.

**Infinity or Until Reset Occurs

***12 Cycles are used beginning with the opcode fetch. A wait state is entered which remains in effect for an integer number of MPU E-clock cycles (n) until an interrupt is recognized. Finally, two additional cycles are used to fetch the appropriate interrupt vector (14 + n total).

dd	= 8-Bit Direct Address ($0000 − $00FF) (High Byte Assumed to be $00)
ff	= 8-Bit Positive Offset $00 (0) to $FF (255) (Is Added to Index)
hh	= High Order Byte of 16-Bit Extended Address
ii	= One Byte of Immediate Data
jj	= High Order Byte of 16-Bit Immediate Data
kk	= Low Order Byte of 16-Bit Immediate Data
ll	= Low Order Byte of 16-Bit Extended Address
mm	= 8-Bit Bit Mask (Set Bits to be Affected)
rr	= Signed Relative Offset $80 (− 128) to $7F (+ 127) (Offset Relative to the Address Following the Machine Code Offset Byte)

1 523G



LEGEND

RTN = Address of next instruction in Main Program to be executed upon return from subroutine

RTN_H = Most significant byte of Return Address

RTN_L = Least significant byte of Return Address

→ = Stack Pointer After Execution

dd = 8-Bit Direct Address ($0000-$00FF) (High Byte Assumed to be $00)

ff = 8-Bit Positive Offset $00 (0) to $FF (256) (Is Added to Index)

hh = High Order Byte of 16-Bit Extended Address

ll = Low Order Byte of 16-Bit Extended Address

rr = Signed Relative Offset $80 (– 128) to $7F (+ 127) (Offset Relative to the Address Following the Machine Code Offset Byte)

1 824

Figure 10-1. Special Operations

Table 10-2. Cycle-by-Cycle Operation – Inherent Mode (Sheet 1 of 3)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-1	ABA, ASLA, ASLB, ASRA, ASRB, CBA, CLC, CLI, CLRA, CLRB, CLV, COMA, COMB, DAA, DECA, DECB, INCA, INCB, LSLA, LSLB, LSRA, LSRB, NEGA, NEGB, NOP, ROLA, ROLB, RORA, RORB, SBA, SEC, SEI, SEV, STOP, TAB, TAP, TBA, TPA, TSTA, TSTB	2	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
2-2	ABX, ASLD, DEX, INX, LSLD, LSRD, TXS, XGDX	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
			3	$FFFF	1	Irrelevant Data
2-3	DES, INS, TSX	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
			3	Previous SP Value	1	Irrelevant Data
2-4	ABY, DEY, INY, TYS, XGDY	4	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Irrelevant Data
			4	$FFFF	1	Irrelevant Data
2-5	TSY	4	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($30)
			3	Opcode Address + 2	1	Irrelevant Data
			4	Stack Pointer	1	Irrelevant Data
2-6	PSHA, PSHB	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	0	Accumulator Data
2-7	PSHX	4	1	Opcode Address	1	Opcode ($3C)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	0	IXL (Low Byte) to Stack
			4	Stack Pointer – 1	0	IXH (High Byte) to Stack
2-8	PSHY	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($3C)
			3	Opcode Address + 2	1	Irrelevant Data
			4	Stack Pointer	0	IXL (Low Byte) to Stack
			5	Stack Pointer – 1	0	IXH (High Byte) to Stack
2-9	PULA, PULB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Operand Data from Stack
2-10	PULX	5	1	Opcode Address	1	Opcode ($38)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	IXH (High Byte) from Stack
			5	Stack Pointer + 2	1	IXL (Low Byte) from Stack
2-11	PULY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($38)
			3	Opcode Address + 2	1	Irrelevant Data
			4	Stack Pointer	1	Irrelevant Data
			5	Stack Pointer + 1	1	IYH (High Byte) from Stack
			6	Stack Pointer + 2	1	IYL (Low Byte) from Stack

*The reference number is given to provide a cross-reference to Table 10-1.

/ 5284

Table 10-2. Cycle-by-Cycle Operation—Inherent Mode (Sheet 2 of 3)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-12	RTS	5	1	Opcode Address	1	Opcode ($39)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Address of Next Instruction (High Byte)
			5	Stack Pointer + 2	1	Address of Next Instruction (Low Byte)
2-13	MUL	10	1	Opcode Address	1	Opcode ($3D)
			2	Opcode Address + 1	1	Irrelevant Data
			3	$FFFF	1	Irrelevant Data
			4	$FFFF	1	Irrelevant Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
			7	$FFFF	1	Irrelevant Data
			8	$FFFF	1	Irrelevant Data
			9	$FFFF	1	Irrelevant Data
			10	$FFFF	1	Irrelevant Data
2-14	RTI	12	1	Opcode Address	1	Opcode ($3B)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Condition Code Register from Stack
			5	Stack Pointer + 2	1	B Accumulator from Stack
			6	Stack Pointer + 3	1	A Accumulator from Stack
			7	Stack Pointer + 4	1	IXH (High Byte) from Stack
			8	Stack Pointer + 5	1	IXL (Low Byte) from Stack
			9	Stack Pointer + 6	1	IYH (High Byte) from Stack
			10	Stack Pointer + 7	1	IYL (Low Byte) from Stack
			11	Stack Pointer + 8	1	Address of Next Instruction (High Byte)
			12	Stack Pointer + 9	1	Address of Next Instruction (Low Byte)
2-15	SWI	14	1	Opcode Address	1	Opcode ($3F)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	0	Return Address (Low Byte)
			4	Stack Pointer − 1	0	Return Address (High Byte)
			5	Stack Pointer − 2	0	IYL (Low Byte) to Stack
			6	Stack Pointer − 3	0	IYH (High Byte) to Stack
			7	Stack Pointer − 4	0	IXL (Low Byte) to Stack
			8	Stack Pointer − 5	0	IXH (High Byte) to Stack
			9	Stack Pointer − 6	0	A Accumulator to Stack
			10	Stack Pointer − 7	0	B Accumulator to Stack
			11	Stack Pointer − 8	0	Condition Code Register to Stack
			12	Stack Pointer − 8	1	Irrelevant Data
			13	Address of SWI Vector (First Location)	1	SWI Service Routine Address (High Byte)
			14	Address of Vector + 1 (Second Location)	1	SWI Service Routine Address (Low Byte)
2-16	WAI	14 + n	1	Opcode Address	1	Opcode ($3E)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	0	Return Address (Low Byte)
			4	Stack Pointer − 1	0	Return Address (High Byte)
			5	Stack Pointer − 2	0	IYL (Low Byte) to Stack
			6	Stack Pointer − 3	0	IYH (High Byte) to Stack
			7	Stack Pointer − 4	0	IXL (Low Byte) to Stack
			8	Stack Pointer − 5	0	IXH (High Byte) to Stack
			9	Stack Pointer − 6	0	A Accumulator to Stack

*The reference number is given to provide a cross-reference to Table 10-1

Table 10-2. Cycle-by-Cycle Operation – Inherent Mode (Sheet 3 of 3)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-16 (Continued)	WAI	14 + n	10	Stack Pointer – 7	0	B Accumulator to Stack
			11	Stack Pointer – 8	0	Condition Code Register to Stack
			12 to n + 12	Stack Pointer – 8	1	Irrelevant Data
			n + 13	Address of Vector (First Location)	1	Service Routine Address (High Byte)
			n + 14	Address of Vector (Second Location)	1	Service Routine Address (Low Byte)
2-17	FDIV, IDIV	41	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
			3 – 41	$FFFF	1	Irrelevant Data
2-18	Page 1 Illegal Opcodes	15	1	Opcode Address	1	Opcode (Illegal)
			2	Opcode Address + 1	1	Irrelevant Data
			3	$FFFF	1	Irrelevant Data
			4	Stack Pointer	0	Return Address (Low Byte)
			5	Stack Pointer – 1	0	Return Address (High Byte)
			6	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			7	Stack Pointer – 3	0	IYH (High Byte) to Stack
			8	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			9	Stack Pointer – 5	0	IXH (high Byte) to Stack
			10	Stack Pointer – 6	0	A Accumulator
			11	Stack Pointer – 7	0	B Accumulator
			12	Stack Pointer – 8	0	Condition Code Register to Stack
			13	Stack Pointer – 8	1	Irrelevant Data
			14	Address of Vector (First Location)	1	Service Routine Address (High Byte)
			15	Address of Vector + 1 (Second Location)	1	Service Routine Address (Low Byte)
2-19	Pages 2, 3, or 4 Illegal Opcodes	16	1	Opcode Address	1	Opcode (Legal Page Select)
			2	Opcode Address + 1	1	Opcode (Illegal Second Byte)
			3	Opcode Address + 2	1	Irrelevant Data
			4	$FFFF	1	Irrelevant Data
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)
			7	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			8	Stack Pointer – 3	0	IYH (High Byte) to Stack
			9	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			10	Stack Pointer – 5	0	IXH (High Byte) to Stack
			11	Stack Pointer – 6	0	A Accumulator
			12	Stack Pointer – 7	0	B Accumulator
			13	Stack Pointer – 8	0	Condition Code Register to Stack
			14	Stack Pointer – 8	1	Irrelevant Data
			15	Address of Vector (First Location)	1	Service Routine Address (High Byte)
			16	Address of Vector + 1 (Second Location)	1	Service Routine Address (Low Byte)
2-20	TEST	Infinite	1	Opcode Address	1	Opcode ($00)
			2	Opcode Address + 1	1	Irrelevant Data
	–		3	Opcode Address + 1	1	Irrelevant Data
			4	Opcode Address + 2	1	Irrelevant Data
			5 – n	Previous Address + 1	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10-1

1 525C

Table 10-3. Cycle-by-Cycle Operation – Immediate Mode

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
3-1	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	2	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Data
3-2	LDD, LDS, LDX	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Data (High Byte)
			3	Opcode Address + 2	1	Operand Data (Low Byte)
3-3	ADDD, CPX, SUBD	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Data (High Byte)
			3	Opcode Address + 2	1	Operand Data (Low Byte)
			4	$FFFF	1	Irrelevant Data
3-4	LDY	4	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($EC)
			3	Opcode Address + 2	1	Operand Data (High Byte)
			4	Opcode Address + 3	1	Operand Data (Low Byte)
3-5	CPD, CPY	5	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Operand Data (High Byte)
			4	Opcode Address + 3	1	Operand Data (Low Byte)
			5	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10-1

1526

Table 10-4. Cycle-by-Cycle Operation – Direct Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
4-1	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data
4 2	STAA, STAB	3	1	Opcode Address	1	Opcode
			2	Opcode Address +	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	0	Data from Accumulator
4-3	LDD, LDS, LDX	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data (High Byte)
			4	Operand Address + 1	1	Operand Data (Low Byte)
4-4	STD, STS, STX –	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be ($00)
			3	Operand Address	0	Register Data (High Byte)
			4	Operand Address + 1	0	Register Data (Low Byte)
4-5 –	LDY	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($DE)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)

*The reference number is given to provide a cross-reference to Table 10-1

1527A

Table 10-4. Cycle-by-Cycle Operation — Direct Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
4-6	STY	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($DF)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	0	Register Data (High Byte)
			5	Operand Address + 1	0	Register Data (Low Byte)
4-7	ADDD, CPX, SUBD	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data (High Byte)
			4	Operand Address + 1	1	Operand Data (Low Byte)
			5	$FFFF	1	Irrelevant Data
4-8	JSR	5	1	Opcode Address	1	Opcode ($9D)
			2	Opcode Address + 1	1	Subroutine Address (Low Byte) (High Byte Assumed to be $00)
			3	Subroutine Address	1	First Subroutine Opcode
			4	Stack Pointer	0	Return Address (Low Byte)
			5	Stack Pointer − 1	0	Return Address (High Byte)
4-9	CPD, CPY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
			6	$FFFF	1	Irrelevant Data
4-10	BCLR, BSET	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Original Operand Data
			4	Opcode Address + 2	1	Mask Byte
			5	$FFFF	1	Irrelevant Data
			6	Operand Address	0	Result Operand Data
4-11	BRCLR, BRSET	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Original Operand Data
			4	Opcode Address + 2	1	Mask Byte
			5	Opcode Address + 3	1	Branch Offset
			6	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10-1

1 5278

Table 10-5. Cycle-by-Cycle Operation — Extended Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
5-1	JMP	3	1	Opcode Address	1	Opcode ($7E)
			2	Opcode Address + 1	1	Jump Address (High Byte)
			3	Opcode Address + 2	1	Jump Address (Low Byte)
5-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data

*The reference number is given to provide a cross-reference to Table 10-1

1 5284

Table 10-5. Cycle-by-Cycle Operation – Extended Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
5-3	STAA, STAB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	0	Accumulator Data
5-4	LDD, LDS, LDX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
5-5	STD, STS, STX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	0	Register Data (High Byte)
			5	Operand Address + 1	0	Register Data (Low Byte)
5-6	LDY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($FE)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	1	Operand Data (High Byte)
			6	Operand Address + 1	1	Operand Data (Low Byte)
5-7	STY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($FF)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	0	Register Data (High Byte)
			6	Operand Address + 1	0	Register Data (Low Byte)
5-8	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	Operand Address	0	Result Operand Data
5-9	TST	6	1	Opcode Address	1	Opcode ($7D)
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
5-10	ADDD, CPX, SUBD	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
			6	$FFFF	1	Irrelevant Data
5-11	CPD, CPY	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	1	Operand Data (High Byte)
			6	Operand Address + 1	1	Operand Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
5-12	JSR	6	1	Opcode Address	1	Opcode ($BD)
			2	Opcode Address + 1	1	Subroutine Address (High Byte)
			3	Opcode Address + 2	1	Subroutine Address (Low Byte)
			4	Subroutine Address	1	First Opcode in Subroutine
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)

*The reference number is given to provide a cross-reference to Table 10-1

Table 10-6. Cycle-by-Cycle Operation — Indexed X Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
6-1	JMP	3	1	Opcode Address	1	Opcode ($6E)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
6-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Operand Data
6-3	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	(IX) + Offset	0	Result Operand Data
6-4	TST	6	1	Opcode Address	1	Opcode ($6D)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
6-5	STAA, STAB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	0	Accumulator Data
6-6	LDD, LDS, LDX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Operand Data (High Byte)
			5	(IX) + Offset + 1	1	Operand Data (Low Byte)
6-7	LDY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($1A)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($EE)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	1	Operand Data (High Byte)
			6	(IX) + Offset + 1	1	Operand Data (Low Byte)
6-8	STD, STS, STX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	0	Register Data (High Byte)
			5	(IX) + Offset + 1	0	Register Data (Low Byte)
6-9	STY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($1A)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($EF)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	0	Register Data (High Byte)
			6	(IX) + Offset + 1	0	Register Data (Low Byte)
6-10	ADDD, CPX, SUBD	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Register Data (High Byte)
			5	(IX) + Offset + 1	1	Register Data (Low Byte)
			6	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10-1

5294

Table 10-6. Cycle-by-Cycle Operation — Index X Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
6-11	CPD, CPY	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	1	Register Data (High Byte)
			6	(IX) + Offset + 1	1	Register Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
6-12	JSR	6	1	Opcode Address	1	Opcode ($AD)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	First Opcode in Subroutine
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer − 1	0	Return Address (High Byte)
6-13	BCLR, BSET	7	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	Opcode Address + 2	1	Mask Byte
			6	$FFFF	1	Irrelevant Data
			7	(IX) + Offset	0	Result Operand Data
6-14	BRCLR, BRSET	7	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	Opcode Address + 2	1	Mask Byte
			6	Opcode Address + 3	1	Branch Offset
			7	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10-1

5298

Table 10-7. Cycle-by-Cycle Operation — Indexed Y Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
7-1	JMP	4	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($6E)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
7-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data
7-3	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	$FFFF	1	Irrelevant Data
			7	(IY) + Offset	0	Result Operand Data

*The reference number is given to provide a cross-reference to Table 10-1

5304

Table 10-7. Cycle-by-Cycle Operation – Indexed Y Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
7-4	TST	7	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($6D)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	$FFFF	1	Irrelevant Data
			7	$FFFF	1	Irrelevant Data
7-5	STAA, STAB	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	0	Accumulator Data
7-6	LDD, LDS, LDX, LDY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data (High Byte)
			6	(IY) + Offset + 1	1	Operand Data (Low Byte)
7-7	STD, STS, STX, STY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	0	Register Data (High Byte)
			6	(IY) + Offset + 1	0	Register Data (Low Byte)
7-8	ADDD, CPD, CPX, CPY, SUBD	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data (High Byte)
			6	(IY) + Offset + 1	1	Operand Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
7-9	JSR	7	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($AD)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	First Opcode in Subroutine
			6	Stack Pointer	0	Return Address (Low Byte)
			7	Stack Pointer – 1	0	Return Address (High Byte)
7-10	BCLR, BSET	8	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	Opcode Address + 3	1	Mask Byte
			7	$FFFF	1	Irrelevant Data
			8	(IY) + Offset	0	Result Operand Data
7-11	BRCLR, BRSET	8	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	Opcode Address + 3	1	Mask Byte
			7	Opcode Address + 4	1	Branch Offset
			8	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10-1

Table 10-8. Cycle-by-Cycle Operation – Relative Mode

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
8-1	BCC, BCS, BEQ, BGE, BGT, BHI, BHS, BLE, BLO, BLS, BLT, BMI, BNE, BPL, BRA, BRN, BVC, BVS	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Branch Offset
			3	$FFFF	1	Irrelevant Data
8-2	BSR	6	1	Opcode Address	1	Opcode ($8D)
			2	Opcode Address + 1	1	Branch Offset
			3	$FFFF	1	Irrelevant Data
			4	Subroutine Address	1	Opcode of Next Instruction
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)

*The reference number is given to provide a cross-reference to Table 10-1

SECTION 11
ELECTRICAL SPECIFICATIONS

11.1 MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	−40 to 85	°C
Storage Temperature Range	Tstg	−55 to 150	°C

This device contains protective circuitry against damage due to high static voltages or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltages higher than maximum-rated voltages to this high-impedence circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either GND or V_{CC}).

11.2 THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance	θ_{JA}		°C/W
Plastic 52-Pin Quad Pack		50	
Plastic 48-Pin DIP		TBD	

11.3 POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \bullet \theta_{JA}) \qquad (1)$$

Where:

T_A = Ambient Temperature, °C

θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W

$P_D = P_{INT} + P_{I/O}$

$P_{INT} = I_{CC} \times V_{CC}$, Watts — Chip Internal Power

$P_{I/O}$ = Power Dissipation on Input and Output Pins — User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

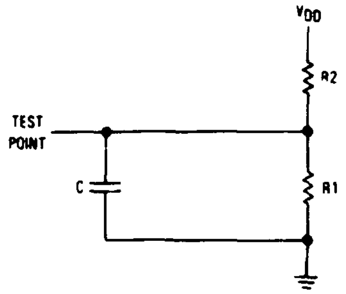
An approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected) is:

$$P_D = K + (T_J + 273°C) \qquad (2)$$

Solving equations-1 and 2 for K gives:

$$K = P_D \bullet (T_A + 273°C) + \theta_{JA} \bullet P_D^2 \qquad (3)$$

Where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.



V_{DD} = 4.5 V

PINS	R1	R2	C
PA0-PA7 PB0-PB7 PC0-PC7 PD0, PD5	3.26 kΩ	2.38 kΩ	90 pF
PD1-PD4	3.26 kΩ	2.38 kΩ	200 pF

1-532

Figure 11-1. Equivalent Test Load

11.4 DC ELECTRICAL CHARACTERISTICS (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = −40 to 85°C, unless otherwise noted)

Characteristics		Symbol	Min	Max	Unit
Output Voltage	All Outputs	V_{OL}	–	0.1	V
I_{Load} = ± 10.0 μA (See Note 1)	All Outputs Except $\overline{RESET}$	V_{OH}	V_{DD} − 0.1	–	
Output High Voltage I_{Load} = −0.8 mA, V_{DD} = 4.5 V (See Note 1)	All Outputs Except $\overline{RESET}$ and EXTAL	V_{OH}	V_{DD} − 0.8	–	V
Output Low Voltage I_{Load} = 1.6 mA	All Outputs Except XTAL	V_{OL}	–	0.4	V
Input High Voltage	All Inputs	V_{IH}	0.7 × V_{DD}	V_{DD}	V
Input Low Voltage	All Inputs	V_{IL}	V_{SS}	0.2 × V_{DD}	V
I/O Ports, 3-State Leakage V_{in} = V_{DD} or V_{IL} (See Note 2)	PA7, PC0-PC7, PD0-PD5, AS/STRA, MODA/$\overline{LIR}$, $\overline{RESET}$	I_{OZ}	–	± 10	μA
Input Current		I_{in}			μA
V_{in} = V_{DD} or V_{SS}	PA0-PA2, PE0-PE7, $\overline{IRQ}$, $\overline{XIRQ}$		–	± 1	
V_{in} = V_{DD}	MODB		–	TBD	
Total Supply Current (See Note 3)					
RUN: Single Chip		I_{DD}	–	20	mA
Expanded Multiplexed			–	TBD	
WAIT: All Peripheral Functions Shut Down		WI_{DD}	–	4	mA
STOP: No Clocks, Single-Chip Mode		SI_{DD}	–	300	μA
Input Capacitance		C_{in}			pF
	PA0-PA2, PE0-PE7, $\overline{IRQ}$, $\overline{XIRQ}$, EXTAL		–	8	
	PA7, PC0-PC7, PD0-PD5, AS/STRA, MODA/$\overline{LIR}$, $\overline{RESET}$		–	12	
Power Dissipation		P_D	–	110	mW

NOTES:

1. V_{OH} specification for reset is not applicable because it is an open-drain pin.
2. See A/D specification for leakage current for port E.
3. All ports configured as inputs, V_{IL} ≤ 0.2 V, V_{IH} ≥ V_{DD} − 0.2 V, no dc loads. EXTAL is driven by square wave. C_L = 20 pF on EXTAL. t_{cyc} = 500 ns.

11.5 CONTROL TIMING ($V_{DD} = 5.0$ Vdc ± 10%, $V_{SS} = 0$ Vdc, $T_A = -40$ to $+85°C$)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Operation External Clock Option	 f_{XTAL} $4f_o$	 – dc	 8.4 8.4	MHz
Internal Operating Frequency Crystal Option (f_{XTAL} ÷ 4) External Clock Option	 f_o f_o	 – dc	 2.1 2.1	MHz
Crystal Oscillator Startup Time (see Figure 11-2)	t_{rc}	–	100	ms
Crystal Oscillator Stop Recovery Startup Time (see Figure 11-3)	t_{ILCH}	–	100	ms
Wait Recovery Startup Time (see Figure 11-4)	t_{IVASH}	–	2	E Cyc
Processor Control Setup Time Before Falling Edge of E (see Figure 11-2)	t_{PCS}	TBD	–	ns
Reset Low Time (Output)	t_{RCCM}	3	4	E Cyc
Reset Rise Time from Internal Reset (see Figure 11-5 and Note 1)	t_{IRR}	–	2	E Cyc
Reset Input Pulse Width (see Figure 11-2 and Note 2)	PW_{RSTL}	2	–	E Cyc
Interrupt Pulse Width Low, $\overline{IRQ}$ Edge-Triggered Mode (see Figure 11-6)	PW_{IRQ}	125	–	ns
Timer Input Capture Pulse Width (see Figure 11-7)	PW_{TIM}	125	–	ns
Pulse Accumulator Input Pulse Width (see Figure 11-7)	PW_{TIM}	125	–	ns

NOTES:

1. This is the maximum time that external components should delay $\overline{RESET}$ rising so that internal COP and clock monitor interrupts can be recognized.
2. This is the minimum time that $\overline{RESET}$ must be held low for an external reset if not preempted by an internal reset (COP or clock monitor). To guarantee an external reset vector will be generated, $\overline{RESET}$ must be held low for a minimum of 8 E cycles.

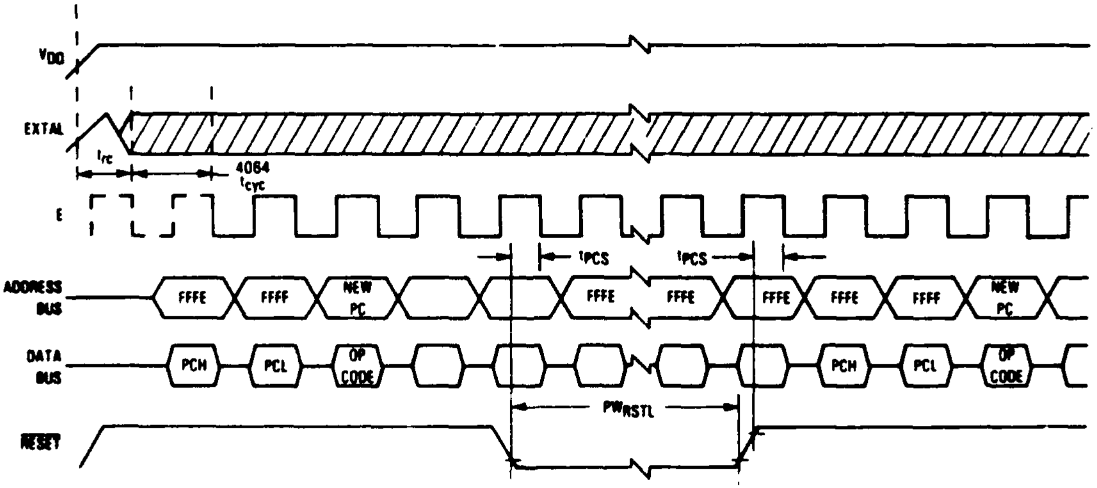
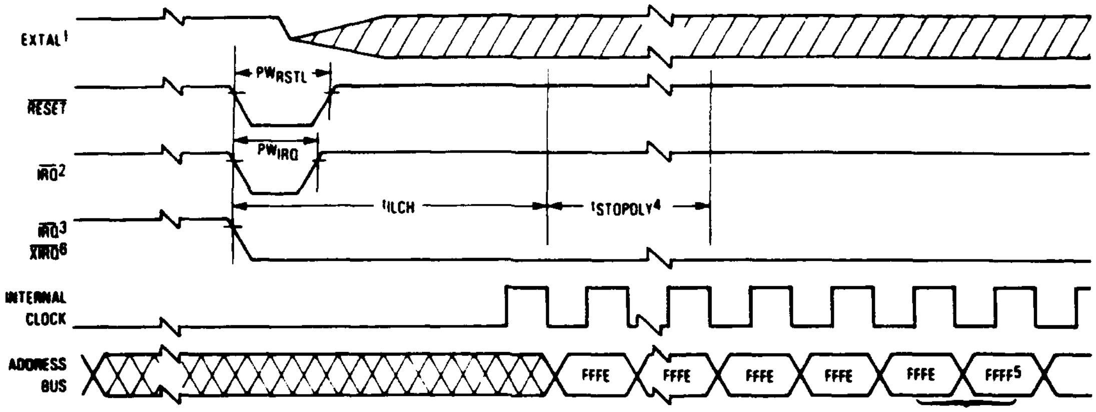


Figure 11-2. Power-On Reset and $\overline{RESET}$ Timing Diagram

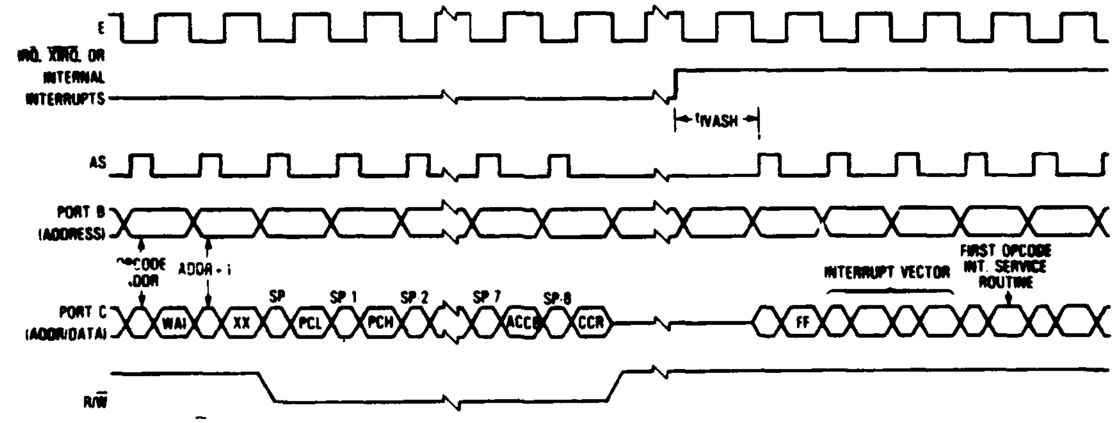


NOTES:

1. Represents the internal gating of the EXTAL pin.
2. Edge-sensitive $\overline{IRQ}$ pin (IRQE bit = 1).
3. Level-sensitive $\overline{IRQ}$ pin (IRQE bit = 0).
4. $t_{STOPDLY}$ = 4064 t_{cyc} if DLY bit = 1 or 4 t_{cyc} if DLY bit = 0.
5. $\overline{RESET}$ vector address shown for timing example.
6. $\overline{XIRQ}$ will cause STOP recovery regardless of X bit state.
7. No vector if STOP terminated by $\overline{XIRQ}$ and no interrupt pending.

1 534

Figure 11-3. Stop Recovery and Power-On Reset Timing Diagram



1 535

Figure 11-4. Wait Recovery From Internal or External Interrupts Timing Diagram

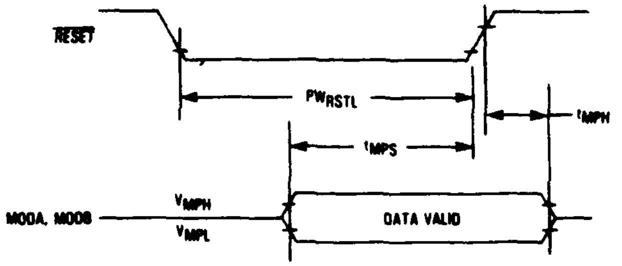


Figure 11-5. Mode Programming

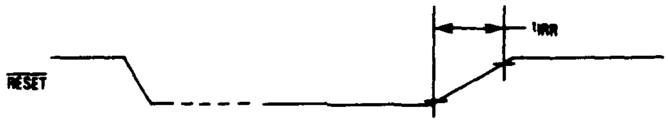


Figure 11-6. Internal $\overline{RESET}$

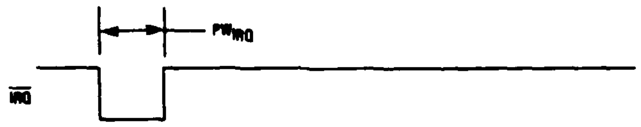


Figure 11-7. Interrupt Timing

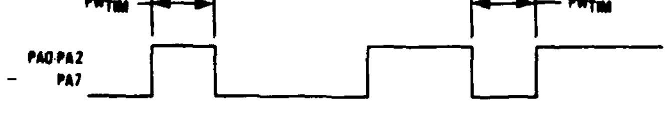


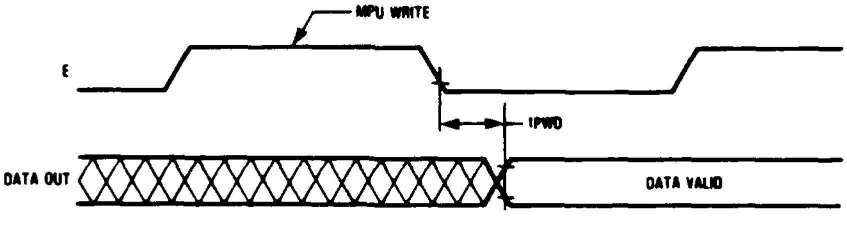
Figure 11-8. Timer Inputs

11.6 PERIPHERAL PORT TIMING (V$_{DD}$ = 5.0 V ± 10%)

Characteristics	Symbol	Min	Max	Unit
Peripheral Data Setup Time (MPU Read, Port A, C, D, and E) (see Figure 11-10)	t_{PDSU}	200	–	ns
Peripheral Data Hold Time (MPU Read, Port A, C, D, and E) (see Figure 11-10)	t_{PDH}	200	–	ns
Delay Time, E Negative Transition to Peripheral Data Valid (MPU Write, Ports A and D) (see Figure 11-9)	t_{PWD}	–	150	ns
Delay Time, Port Data Valid to E Negative Transition (MPU Write, Ports B and C) (see Figures 11-11, 11-13, and 11-15)	t_{PDV}	0	–	ns
Input Data Setup Time (Port C) (see Figures 11-12 and 11-13)	t_{IS}	60	–	ns
Input Data Hold Time (Port C) (see Figures 11-12 and 11-13)	t_{IH}	100	–	ns
Delay Time, E Negative Transition to STRB Asserted (see Figures 11-11, 11-14, and 11-15)	t_{DEB}	140	260	ns
Delay Time, E Negative Transition to STRB Negated (see Figures 11-13 and 11-15)	t_{DEBN}	140	260	ns
Setup Time, STRA Asserted to E Negative Transition (see Note 1 and Figures 11-13, 11-14, and 11-15)	t_{AES}	0	–	ns
Delay Time, STRA Asserted to Port C Data Out Valid (see Figure 11-15)	t_{PCD}	–	100	ns
Hold Time, STRA Negated to Port C Data (see Figure 11-15)	t_{PCH}	10	–	ns
Three-State Hold Time (see Figure 11-15)	t_{PCZ}	0	150	ns

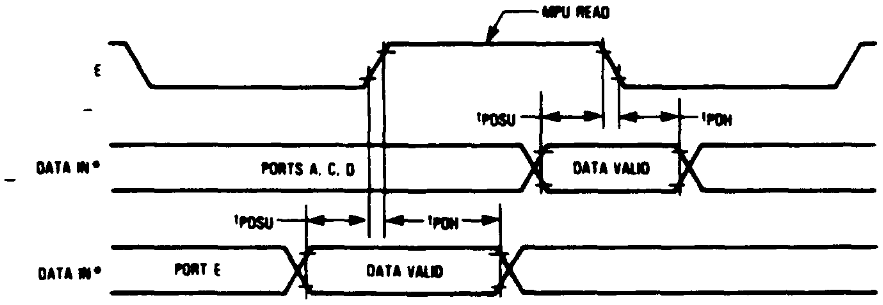
NOTES:

1. If this setup time is met, STRB will acknowledge in the next cycle. If it is not met, the response may be delayed one more cycle.
2. Port C and D timing is valid for active drive (CWOM and DWOM bits not set in PIOC and SPCR registers respectively).
3. All timing is shown with respect to 20% V$_{DD}$ and 70% V$_{DD}$ unless otherwise noted.



1 540

Figure 11-9. Port Write or Timer Output Compare



*Non-latched operation

1 541

Figure 11-10. Port Read

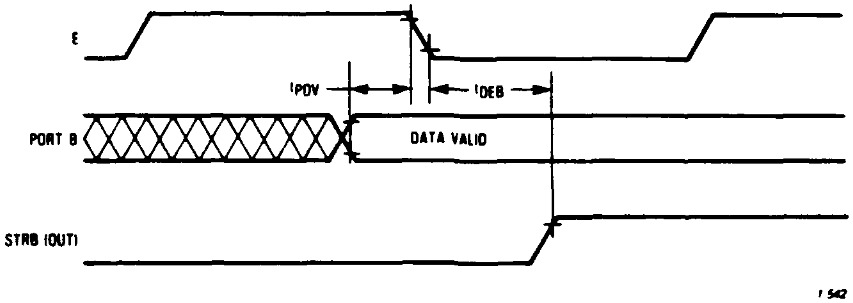


Figure 11-11. Simple Output Strobe

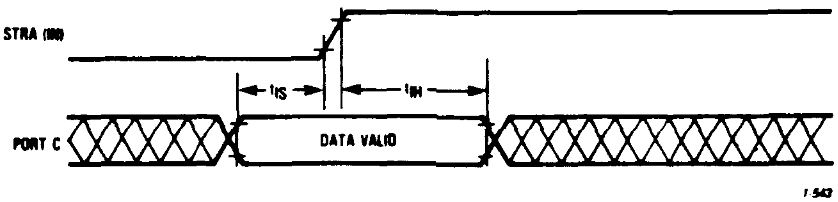


Figure 11-12. Simple Input Strobe

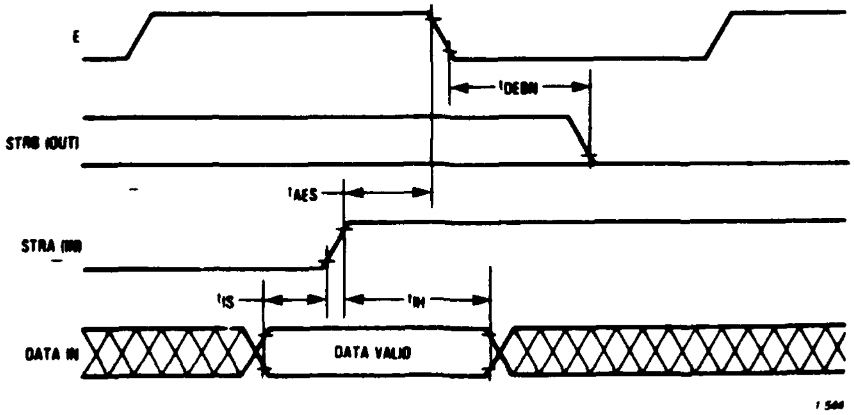


Figure 11-13. Port C Input Handshake

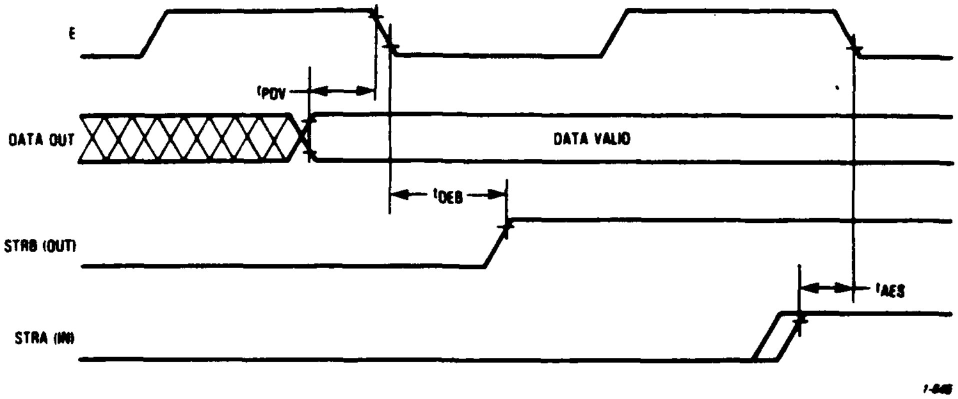


Figure 11-14. Port C Output Handshake

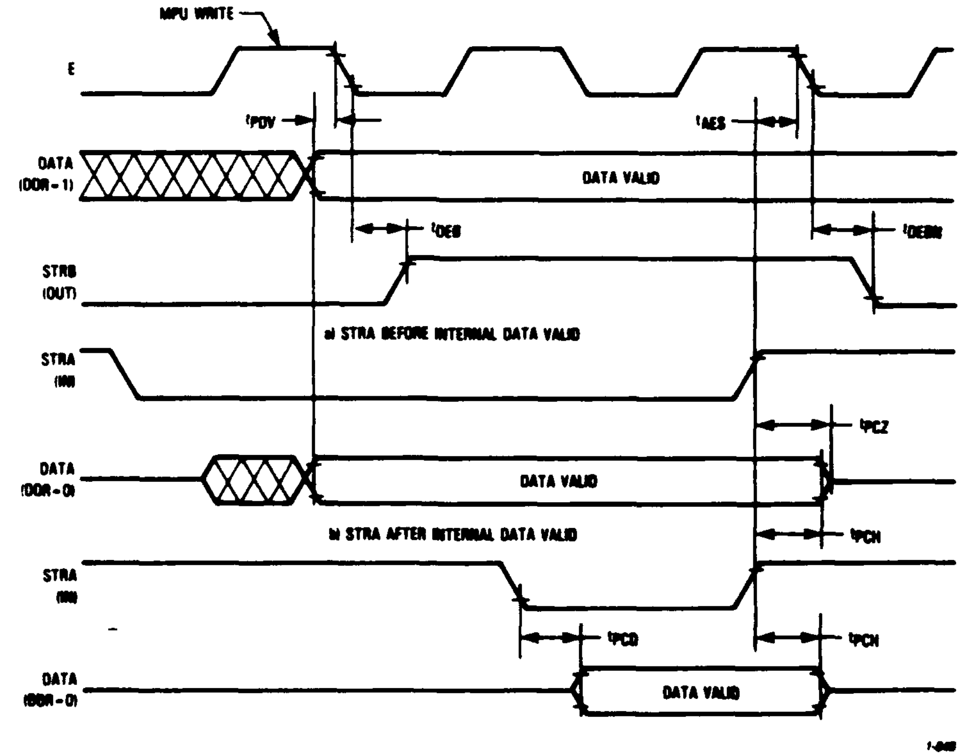


Figure 11-15. Port C Output Handshake with 3-State Enabled (STRA Enables Output Buffer)

11.7 EXPANSION BUS TIMING (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = 25°C ± 5°, unless otherwise noted) (see Figure 11-16)

Num	Characteristic	Symbol	1.05 MHz		2.1 MHz		Unit
			Min	Max	Min	Max	
1	Cycle Time	t_{cyc}	952	–	476	–	ns
2	Pulse Width, E Low	PW_{EL}	400	–	200	–	ns
3	Pulse Width, E High	PW_{EH}	420	–	210	–	ns
4	E Rise and Fall Time	t_r, t_f	–	25	–	20	ns
9	Address Hold Time (see Note 1)	t_{AH}	50	–	30	–	ns
12	Non-Muxed Address Valid Time to E (see Note 2)	t_{AV}	200	–	75	–	ns
17	Read Data Setup Time	t_{DSR}	80	–	30	–	ns
18	Read Data Hold Time	t_{DHR}	10	80	10	80	ns
19	Write Data Delay Time	t_{DDW}	–	225	–	125	ns
21	Write Data Hold Time	t_{DHW}	50	–	30	–	ns
22	Muxed Address Valid Time to E Rise (see Note 2)	t_{AVM}	200	–	75	–	ns
24	Muxed Address Valid Time to AS Fall (see Note 2)	t_{ASL}	60	–	20	–	ns
25	Muxed Address Hold Time (see Note 1)	t_{AHL}	50	–	30	–	ns
26	Delay Time, E to AS Rise (see Notes 1 and 2)	t_{ASD}	90	–	40	–	ns
27	Pulse Width, AS High (see Note 2)	PW_{ASH}	200	–	90	–	ns
28	Delay Time, AS to E Rise (see Notes 1 and 2)	t_{ASED}	90	–	40	–	ns
29	MPU Address Access Time (Computed see Note 2) (12 or 22) Whichever is Smaller, Address Valid + 4 Clock Rise Time + 3 Pulse Width, E High – 17 Read Data Setup Time	t_{ACCA}	615	–	275	–	ns
35	MPU Access Time (Computed see Note 4) + 3 Pulse Width, E High – 17 Read Data Setup Time	t_{ACCE}	–	390	–	180	ns
36	Multiplexed Address Delay (Previous Cycle MPU Read)	t_{MAD}	120	–	80	–	ns
37	Internal Read Data Valid Before E	t_{IRV}	10	–	10	–	ns
38	Multiplexed Address Off Before E	t_{ADP}	0	–	0	–	ns
39	Write Data Setup Time (Computer see Note 5) + 3 Pulse Width, E High – 19 Write Data Delay Time	t_{DSW}	195	–	110	–	ns
40	MODA/$\overline{\text{LIR}}$ Hold Time (During Opcode Fetch)	t_{LRH}	50	–	30	–	ns

NOTES:

1. Affected by input clock duty cycle (XTAL, EXTAL)
2. At specified cycle time.
3. Address access time is computed by: (12 opr 22) + 4 + 3 – 17.
4. No device should drive port C except when E is high or contention may result.
5. E (enable) access time is computed by: 3 – 17.

Timing diagram (Figure 11-16) is located on a foldout page at the end of this document.

11.8 SERIAL PERIPHERAL INTERFACE (SPI) TIMING (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = −40°C to 85°C) *(See Figure 11-17)*

Num	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency				
	Master	$f_{op(m)}$	dc	1.05	MHz
	Slave	$f_{op(s)}$	dc	2.1	
1	Enable Lead Time				ns
	Master	$t_{lead(m)}$	*	–	
	Slave (CPMA = 0)	$t_{lead(S0)}$	240	–	
	Slave (CPMA = 1)	$t_{lead(S1)}$	100	–	
2	Enable Lag Time				ns
	Master	$t_{lag(m)}$	*	–	
	Slave (CPMA = 0)	$t_{lag(S0)}$	0.0	–	
	Slave (CPMA = 1)	$t_{lag(S1)}$	125	–	
3	Clock (SCK) High Time				ns
	Master	$t_{w(SCKH)m}$	TBD	–	
	Slave	$t_{w(SCKH)s}$	TBD	–	
4	Clock (SCK) Low Time				ns
	Master	$t_{w(SCKL)m}$	TBD	–	
	Slave	$t_{w(SCKL)s}$	TBD	–	
5	Data Setup Time (Inputs)				ns
	Master	$t_{su(m)}$	100	–	
	Slave	$t_{su(s)}$	100	–	
6	Data Hold Time (Inputs)				ns
	Master	$t_{h(m)}$	100	–	
	Slave	$t_{h(s)}$	100	–	
7	Access Time	t_a	–	TBD	ns
	Slave				
8	Disable Time (Hold Time to High-Impedance State)	t_{dis}	–	TBD	ns
	Slave				
9	Data Valid				ns
	Master (Before Capture Edge)	$t_{v(B)m}$	TBD	–	
	Slave (After Enable Edge)**	$t_{v(B)s}$	–	200	
10	Data Valid	$t_{v(A)}$	TBD	–	ns
	Master (After Capture Edge)				
11	Rise Times (20% V_{DD} to 70% V_{DD}, CL = 200 pF)				
	SPI Outputs SCK, MOSI, MISO	$t_{r(SCK)m}$	–	100	ns
	SPI Inputs SCK, MOSI, MISO, $\overline{SS}$	$t_{r(SCK)s}$	–	2.0	µs
12	Fall Times (20% V_{DD} to 70% V_{DD}, CL = 200 pF)				
	SPI Outputs SCK, MOSI, MISO	$t_{f(SCK)m}$	–	100	ns
	SPI Inputs SCK, MOSI, MISO, $\overline{SS}$	$t_{f(SCK)s}$	–	2.0	µs
13	Output Data Hold (After Enable Edge)				ns
	Master	$t_{ho(m)}$	0	–	
	Slave	$t_{ho(s)}$	0	–	

*Signal Production Depends on Software

**Assumes 200 pF Load on All SPI Pins

Timing diagram (Figure 11-17) is located on foldout pages at the end of this document.

11.9 A/D CONVERTER CHARACTERISTICS (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = −40°C to 85°C, $f_O \leq 1.0$ MHz $\leq E \leq 2.1$ MHz, unless otherwise noted) (see Note 1)

Characteristic	Parameter	Min	Max	Unit
Resolution	Number of bits resolved by the A/D	8	–	Bit
Non-Linearity	Maximum deviation from the best straight line through the A/D transfer characteristics ($V_{RH} = V_{DD} \pm 100$ mV and $V_{RL} = 0$ V)	–	½	LSB
Quantization Error	Uncertainty due to converter resolution	–	½	LSB
Absolute Accuracy	Difference between the actual input voltage and the full-scale weighted equivalent of the binary output code for all errors	–	± 1	LSB
Conversion Range	Analog input voltage range	V_{RL}	V_{RH}	V
V_{RH}	Maximum analog reference voltage	V_{RL}	$V_{DD} + 0.1$	V
V_{RL}	*Minimum analog reference voltage*	−0.1	V_{RH}	V
Conversion Time	Total time to perform a single analog to digital conversion a. External clock (XTAL, EXTAL) b. *Internal RC oscillator*	 – –	 32 32	 t_{cyc} μs
Monotonicity	Conversion result never decreases with an increase in input voltage and has no missing codes	Guaranteed		
Zero Input Reading	Conversion result when $V_{in} = V_{RL}$	00	–	Hex
Full Scale Reading	Conversion result when $V_{in} = V_{RH}$	–	FF	Hex
Sample Acquisition Time (see Note 1)	Analog input acquisition sampling time a. External clock (XTAL, EXTAL) b. Internal RC oscillator	 – –	 12 12	 t_{cyc} μs
Sample/Hold Capacitance	*Input capacitance on PE0-PE7 pins*	–	8	pF
Input Leakage	Input leakage on A/D pins (PE0-PE7) (see Note 2) (V_{RL}, V_{RH})	– –	400 1.0	nA μA

NOTES:

1. Source impedances greater than 10K ohm *will adversely affect internal RC charging time during input sampling.*
2. The external system error caused by input leakage current is approximately equal to the product of R source and input current.

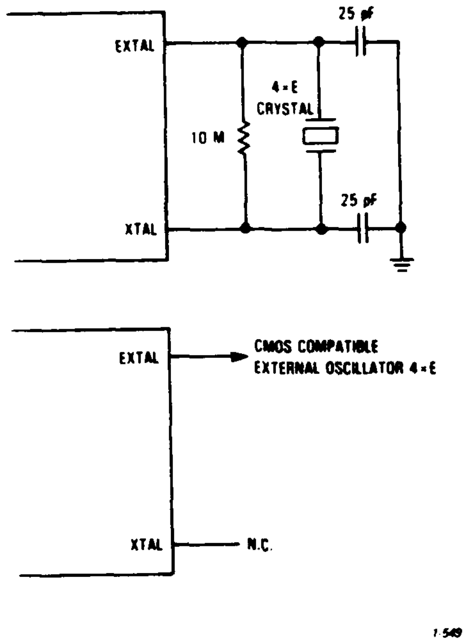


Figure 11-18. Oscillator Circuits

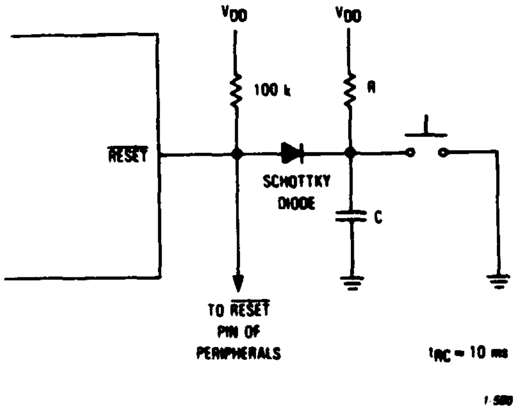


Figure 11-19. Typical Reset Circuit

SECTION 12
MECHANICAL DATA AND ORDERING INFORMATION

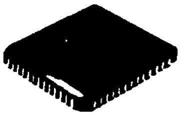
This section contains the pin assignment and package dimension diagrams for the MC68HC11A8 as well as information to be used as a guide when ordering the MCU.

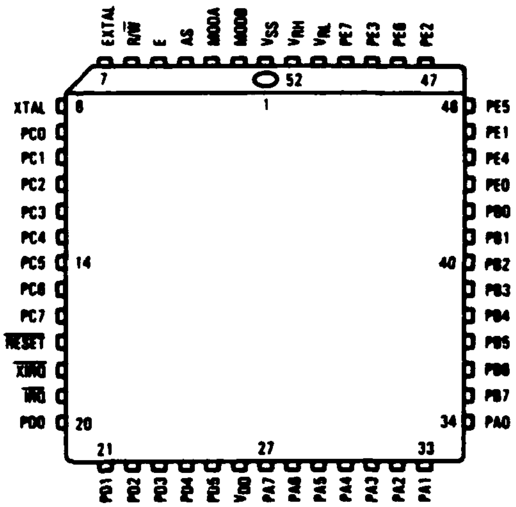
12.1 PIN ASSIGNMENTS

The MC68HC11A8 is available in both a 48-pin plastic dual-in-line package and a 52-lead quad pack. The following paragraphs provide pin assignments for both package versions.

12.1.1 52-Lead Quad Package

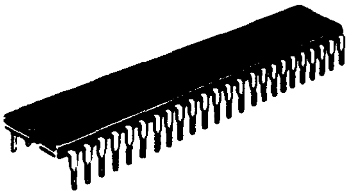
FN SUFFIX
QUAD PACKAGE

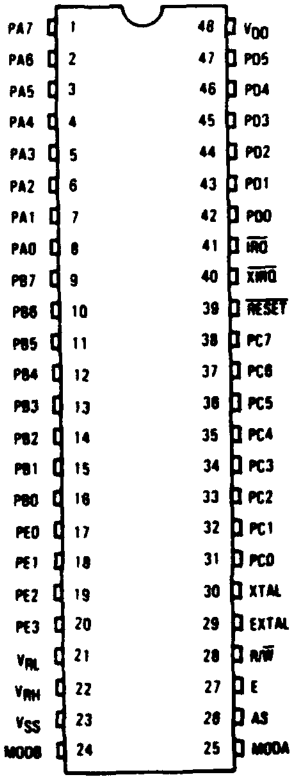




12.1.2 48-Pin Dual-in-Line Package

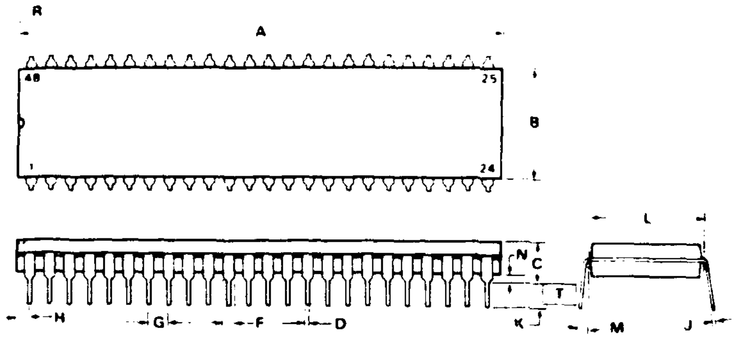
P SUFFIX
PLASTIC PACKAGE





12.2 PACKAGE DIMENSIONS

P SUFFIX
PLASTIC
CASE 767-02

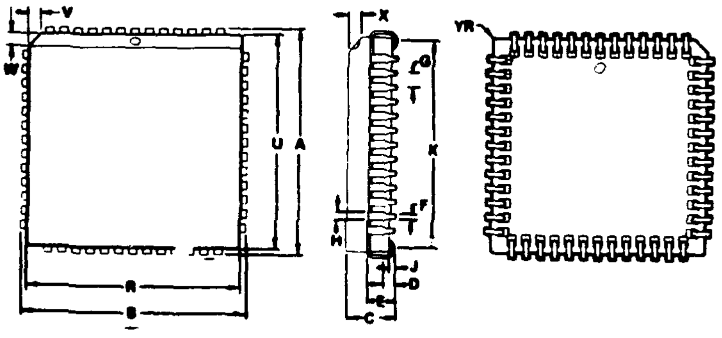


NOTES

1. R IS END OF PACKAGE DATUM PLANE
   T IS BOTH A DATUM AND SEATING PLANE
2. POSITIONAL TOLERANCE FOR LEADS 1 AND 48
   | ⊕ | 0 51 (0 020) | T | B Ⓜ | R |
   POSITIONAL TOLERANCE FOR LEAD PATTERN
   | ⊕ | 0 25 (0 010) | T | B Ⓜ |
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH
4. DIMENSION L IS TO CENTER OF LEADS WHEN FORMED PARALLEL
5. DIMENSIONING AND TOLERANCING PER ANSI Y14 5 1982
6. CONTROLLING DIMENSION INCH

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	61 34	62 10	2 415	2 445
B	13 72	14 22	0 540	0 560
C	3 94	5 08	0 155	0 200
D	0 36	0 55	0 014	0 022
F	1 02	1 52	0 040	0 060
G	2 54 BSC		0 100 BSC	
H	1 79 BSC		0 070 BSC	
J	0 20	0 38	0 008	0 015
K	2 92	3 42	0 115	0 135
L	15 24 BSC		0 600 BSC	
M	0	15	0	15
N	0 51	1 01	0 020	0 040

FN SUFFIX
QUAD PACK
CASE 778-01



NOTES:

1. DIMENSIONS R AND U DO NOT INCLUDE MOLD FLASH
2. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982
3. CONTROLLING DIMENSION: INCH

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	19.94	20.19	0.785	0.795
B	19.94	20.19	0.785	0.795
C	4.19	4.57	0.165	0.180
D	0.64	1.01	0.025	0.040
E	2.16	2.79	0.085	0.110
F	0.33	0.53	0.013	0.021
G	1 27 BSC		0 050 BSC	
H	0.66	0.81	0.026	0.032
J	0.38	0.63	0.015	0.025
K	17.52	18.54	0.690	0.730
R	19.05	19.20	0.750	0.756
U	19.05	19.20	0.750	0.756
V	1 07	1 21	0 042	0.048
W	1 07	1 21	0.042	0.048
X	1 07	1 42	0.042	0.056
Y	0.00	0 50	0.000	0.020

12.3 ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

EPROM(s), MCM2716 or MCM2532
MDOS, disk file

To initiate a ROM pattern for the MCU, it is necessary to first contact your local field service office, local sales person, or your local Motorola representative.

12.3.1 EPROMs

The MCM2716 or MCM2532 type EPROMs, programmed with the customer program (positive logic sense for address and data), may be submitted for pattern generation. The EPROMs must be clearly marked to indicate which EPROM corresponds to which address space. Figure 12-1 illustrates the markings for the four MCM2716 EPROMs required to emulate the MC68HC11A8 MCU. All unused bytes should be programmed to zeros.

After the EPROM(s) are marked, they should be placed in conductive IC carriers and securely packed. Do not use styrofoam.

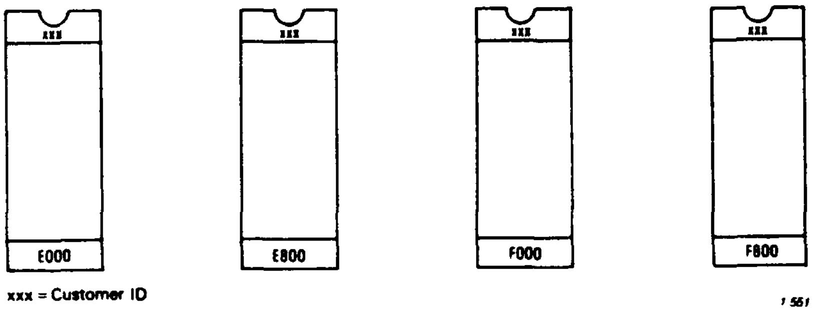


Figure 12-1. EPROM Marking Example

12.3.2 MDOS Disk File

An MDOS disk, programmed with the customer program (positive logic sense for address and data), may be submitted for pattern generation. **WHEN USING THE MDOS DISK, INCLUDE THE ENTIRE MEMORY IMAGE OF BOTH DATA AND PROGRAM SPACE. ALL UNUSED BYTES, INCLUDING THE USER'S SPACE, MUST BE SET TO ZERO.**

12.3.3 Verification Media

All original pattern media (EPROMs or floppy disk) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for creation of the customer mask. If desired, Motorola will program blank MCM2716 or MCM2532 EPROMs (supplied by the customer) from the data file used to create the custom mask to aid in the verification process.

12.3.4 ROM Verification Units

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency, they are usually unmarked and tested only at room temperature (25°C) and five volts. These RVUs are included in the mask charge and are not production parts. These RVUs are not backed nor guaranteed by Motorola Quality Assurance.

12.3.5 Flexible Disks

The disk media submitted must be single-sided, single density, 8-inch, MDOS compatible floppies. The customer must clearly label the disk with the ROM pattern file name and company name. The floppies are not returned by Motorola, as they are used for archival storage. The minimum MDOS system files as well as the absolute binary object file (filename.LO type of file) from the MC68HC11A8 cross assembler must be on the disk. An object file made from the memory dump using the ROLLOUT command is also admissable. Consider submitting a source listing as well as: filename, .LX (EXORciser loadable format). This file will, of course, be kept confidential and is used 1) to speed up the process in house if any problems arise, and 2) to speed up our customer to factory interface if a user finds any software errors and needs assistance quickly from the factory representative.

MDOS is Motorola's Disk Operating System available on development systems such as EXORciser, EXORset, etc.

12.3.6 Sample Order Form

A sample order form is provided on the next page.

Appendix B - Max-Forth manual

MAX-FORTH

REFERENCE MANUAL ADDENDUM

NEW MICROS, INC.
1601 CHALK HILL RD.
DALLAS, TEXAS 75212
T-214-339-2204

THANK YOU

Thank you for purchasing Max-FORTH. We think we have created an excellent product and we hope that it will be of benefit to you. We are looking forward to a continuing business relationship.

If you have problems, however, we want to hear about them. Please refrain from cursing at us, at least until you've had a chance to talk with us (after that, use your own judgment!). Meaning that, we want to support your development effort by providing correct and sufficient information about our product. This doesn't mean we want to work as free consultants, but it does mean you can call us if you need to be pointed in the right direction. If you find something that doesn't work, we want to hear from you--the sooner the better.

It is our desire to add a great deal more information to the user's manual. We will send the first update to you prepaid since you have received a preliminary manual, thereafter, revisions are free for the cost of shipping. We realize that the manual needs more 68HC11 specifics and examples. This is, again, why you are encouraged to call with your questions, so we will know what information is really important in the manual updates.

There is a registration form enclosed with the manual. Please use it. If you don't, manual updates and possible application notes are likely to go to a receiving department dock, rather than directly to you.

Well then, thank you, good luck and good hunting (and may the forth be with you).

COMPANY INTRODUCTION TO NEW MICROS, INC.
Product Lines and Services

We at New Micros, Inc. see ourselves as a leader in high level language single chip computers and related board level products. With the introduction of the F68HC11, we became the only supplier in the world of a single chip computer that can be programmed in high level language, without any support chips. Additionally, we manufacture our own line of standard board level computer products and also design and manufacture semi-custom and fully custom computer systems. We refer here to standard board level products as boards that are already designed and tested and carried as stock items. The semi-custom boards generally follow the same board outline and form factor as the standard line boards and are usually peripherals, rather than processor boards, that mate with the processor boards of the existing lines. Semi-custom boards are designed specifically for a customers needs and may or may not be added to the standard line later. Fully custom boards are those that are designed specifically for a customers application. Form and function follow the customer's needs. Custom boards are not added to the standard product line.

The products which are listed on our price sheet fall into two categories: those which are standard items available only from New Micros, Inc. and those that are generally available commodity items, such as RAM's and Wall Transformer Power Supplies, etc..

The standard processor board items fall into three pricing stratum. The "full up" development systems are priced to the market average for single board computers. This positioning reflects an expectation that the customer will only desired a few systems configured that way. The development systems have all board options installed and are shipped with manuals, Case, Power Supply and FORTH Development Rom. Many of the comparably priced single board computers have less features or options and are, for the most part, only boards - not high level language development systems with power supplies, etc., etc.. The OEM versions have full board options, however, they are supplied without other frills. They have no metal case or wall transformer power supply (although onboard rectification, regulation is installed) or development rom (if applicable). They are intended for the customer who wants a full featured SBC (with RS-232, Watchdog Timer, etc.) without the added expenses. These systems will probably be embedded in a larger machine or system manufactured by the customer, which is why we call the configuration, "OEM". Finally, "The Generic Target Computer"TM or "GTC" versions of our processor boards have the minimum configuration of board options. The GTC's are 5 Volt only systems, without accessories such as connectors, memory selection jumpers, RS-232 or RS-422/485 conversion, etc..

The purpose of the GTC is to give the most features, at the lowest possible price, of any stock single board computer. The GTC's exceed all the known competitors pricing on single board computers. All of the mentioned versions are made on the same PC board. This means the GTC board has the same number of holes as the OEM version, and any or all of the additional features of the OEM boards can be added to the GTC by the customer or by special arrangement with NMI.

Our peripheral boards, which we believe to be the best values around, are available in only one configuration. Still special arrangements are sometimes made on a per case basis to make a reduced cost version if some included features can be removed.

As mentioned earlier, there are also a number of generally available commodity items on the NMI price lists. In essence, these items are listed only as a service to the customer who wants to avoid calling all around to find parts that are assured to work in our systems. Please understand, we may not always be competitive on these items, particularly the memory products. Our feelings will certainly not be hurt if you find a better price else where. We don't move enough of these items to make it possible to follow the latest price swings in the market, but if you need them, we are generally able to supply.

As implied by the fact we have made mention of the custom side of our business, we are equiped and experienced at doing customer promoted design work. We have several CAD packages for PCB and mechanical designs. Our rates are competitive, but our forte' lies in our familiarity with FORTH and dedicated applications. Working in this area is not only a source of income to us. It is also the way we stay in touch with the market, and the designer's needs. Even in our business, we recognize that no single board computer is ever exactly "right" for a final application. We have served many customers applications by making a semi-custom peripheral board to add features and solve interconnect problems. Contact us if we can be of service. (214) 339-2204 for Technical Assistance, 1-(800)-255-4664 for NMI Sales outside Texas, Telex No. 9102500125 NEW MICROS DAL UQ.

ORDER FORM

Company Name ______________________________ Date ____________________

Engineer's Name ______________________ Buyer's Name ____________________

Shipping Address ______________________ Billing Address ____________________

______________________________ ______________________________

Engineer's Phone No ____________________ Buyer's Phone No ____________________

Method of Payment:

○ Net 30 days requires your Purchase Order No: ______________________________

○ Check or Money Order No: __________________________ Amount ______________

○ Mastercard or ○ VISA Exp ______________ Account No ____________________

○ COD (Add $ 1.95 COD fee)

Quantity	Part No	Description	unit	total

shipping & handling ______

If applicable, add COD fee ($1.95) ______

TX residents add 5 1/8 % tax ______

Total ______

1601 CHALK HILL RD SUITE A DALLAS, TEXAS 75212 214 339-2204

PRICE LIST 3/87

Item No	Description	Price in US$
NMIX-0011-CPSDR	R65F11-Based Single-Board Microcomputer: w/Case,PS, Dev ROM	250.00
NMIX-0011-OEM	R65F11-Based Single-Board Microcomputer: OEM Configured	190.00
NMIX-0012-CPSDR	R65F12-Based Single-Board Microcomputer: w/Case,PS, Dev ROM	290.00
NMIX-0012-OEM	R65F12-Based Single-Board Microcomputer: OEM Configured	230.00
NMIX-0013	R65F11-Based Single Board Microcomputer with Built-In FDC	340.00
NMIX-0021-CPS	F68HC11-Based Single-Board Microcomputer	270.00
NMIX-0022-CPS	F68HC11-Based Single-Board Microcomputer with 68HC24	290.00
NMIX-0021-OEM	F68HC11-Based Single-Board Microcomputer	200.00
NMIX-0022-OEM	F68HC11-Based Single-Board Microcomputer with 68HC24	220.00
NMIX-0023	F68HC11-Based Single-Board Microcomputer: spcl price/config	199.00
NMIT-0011-3	R65F11-Based Generic Target Computer:with 3 Memory Sockets	71.00
NMIT-0012-3	R65F12-Based Generic Target Computer:with 3 Memory Sockets	91.00
NMIT-0021	F68HC11-Based Generic Target Computer	90.00
NMIT-0022	F68HC11-Based Generic Target Computer with 68HC24	110.00
NMIX-4012	12-Bit A/D Plus Sign Board	210.00
NMIX-5000	ACIA Board	75.00
R65F11/12:	Enhanced 6502-Based Single-Chip with RSC-FORTH internal:	CALL
F68HC11FN V3.3	68HC11 Single-Chip Microcomputer Max-FORTH V3.3 in int	37.25
XC68HC11A0FN	68HC11 Single-Chip Microcomputer	27.25
XC68HC24	68HC24 Port Replacement Part for use with 68HC11's	20.00
Max-FORTH	New Micros Proprietary FORTH Software	
V1.1	Max-FORTH in ROM	CALL
V2.3	Max-FORTH in ROM with Autostart	CALL
Options & Accessories:		
23CBL232-4	RS-232 Cable for NMIX-0023	15.00
CASE	for OEM Versions of P56F11/12-Based Systems	25.00
NEMA	Industrial Grade Enclosure for R65F11/12-Based Systems	25.00
PS	Power Supply for OEM Versions of R65F11/12-Based Systems	6.00
23PS	Power Supply for NMIX-0023	6.00
DR	Development ROM for R65F11/12-Based Systems-Incl. UM-RSC	62.50
#2 PROM	2K Address Decoder PROM for R65F11/12-Based Systems	8.00
#8 PROM	8K Address Decoder PROM for R65F11/12-Based Systems	8.00
#11 PROM	Address Decoder PROM for NMIX-0013 System	8.00
#17 PROM	Address Decoder PROM for NMIX-0013 System	8.00
#18 PROM	Address Decoder PROM for NMIX-0013 System	8.00
6264	8K RAM with #8 PROM	12.00
2816A	2K EEPROM	20.00
48Z02	2K Battery RAM	20.00
DS1213C/8	8K Battery RAM	30.00
DS1213C/32	32K Battery RAM	50.00
PCB-0013	Bare Board for NMIX-0013 system	38.00
UM-MAX	Max-FORTH User's Manual	30.00

PAYMENT METHOD: Personal check (US orders), Cashier's Check or Money Order (US Currency), or VISA/MC.

SHIPPING CHARGES:			
	U.S.	$3.00	UPS Ground
		$5.00	2nd Day Air
	Canada	$10.00	US Air Mail/UPS
	Europe	$25.00	US Air Mail
	Asia	$30.00	US Air Mail

NEW MICROS INC.

PRODUCT MATRIX
for
NMI PROCESSOR BOARDS 3/87

BOARD FEATURE	X-0011-CPSDR	X-0012-CPSDR	X-0011-OEM	X-0012-OEM	X-0013-	X-0021-CPS	X-0022-CPS	X-0021-OEM	X-0022-OEM	X-0023-	T-0011-3	T-0012-3	T-0021-3	T-0022-3
R65F11 PROCESSOR	X		X		X						X			
R65F12 PROCESSOR		X		X								X		
F68HC11 PROCESSOR						X	X	X	X	X			X	X
MC68HC24 PRU							X		X					X
ADDRESS SPACE	16K	16K	16K	16K	16K	64K	64K	64K	64K	64K	16K	16K	64K.	64K
POWER mA typ @ 5V	220	220	220	220	300	20	20	20	20	20	220	220	20	20
STD XTAL MHz	2	4	2	4	2	8	8	8	8	8	2	4	8	8
STD BAUD X 100	12	24	12	24	12	96	96	96	96	96	12	24	96	96
STD RAM INST KB	2	2	2	2	2	8	8	8	8	8	0	0	0	0
DEVELOPMENT ROM	X	X			X	X	X	X	X	X			X	X
STD MANUALS	2	2	0	0	1	2	2	0	0	2	0	0	0	0
ADDRESING PROM #	2	2	2	2	11	N/A	N/A	N/A	N/A	N/A	2	2	N/A	N/A
PARALLEL PORTS	2	5	2	5	2	3	5	3	5	3	2	5	3	5
ASYNC SERIAL	1	1	1	1	1	1	1	1	1	1	1	1	1	1
SYNC SERIAL						1	1	1	1	1			1	1
A/D 8-CH 8-BIT						X	X	X	X	X			X	X
EEPROM 512 BYTES						X	X	X	X	X			X	X
COUNTER/TIMERS	2	2	2	2	2	9*	9*	9*	9*	9*	2	2	9*	9*
WATCH DOG TIMER	X	X			X	X	X	X	X	X			X	X
RS-232 CONVERSION	X	X			X	X	X	X	X	X				
RS-232 DB25 CONN.	X	X			X	X	X	X	X					
RESET BUTTON	X	X			X	X	X	X	X					
RS-422/485 CONV.	X	X				X	X	X	X					
RECTIFER, REGU.	X	X				X	X	X	X	X				
DC/DC CONVERTOR	X	X				X	X	X	X	X				
SCREW TERMINALS	X	X				X	X	X	X	X				
INSTALLED JMPRS	X	X				X	X	X	X	X				
VSC BUSS CONN.	X	X				X	X	X	X					
CASE, POWER SUP	X	X				X	X							
FLOPPY CONTROLLER					X									
LIST PRICE 1ea $	250	290	190	230	340	270	290	200	220	199	76	91	90	110

* 3 INPUT CAPTURES, 5 OUTPUT COMPARES AND 1 PULSE ACCUMULATOR

The F68HC11 Single Chip Computer

F68HC11FN V3.3

Specifications

The F68HC11 is a complete computer on a chip with operating system and high level language development tools included in the on-board Max-FORTH 8K-byte ROM, which allow it to be programmed with no additional support chips. In minimum configuration, the F68HC11, with a simple crystal circuit (a crystal, a resistor and two capacitors), needs only 5 V power, ground and TTL serial-in/serial-out connections to operate interactively.

Processor: MOTOROLA 68HC11, Enhanced 6800/6801

Language: MAX-FORTH, 231 words: A superset of the FORTH-83 Standard including:
- * The 83 Standard Word Set.
- * The 83 Standard Double Number Extension Word Set
- * The 83 Standard System Extensions Word Set
- * All the 83 Standard Controlled Reference Words
- * Single-Chip System Extensions

Counter/Timers: Enhanced 16-bit timer system: Four stage programmable prescaler, three 16-bit input capture registers, five 16-bit output-compare registers, real time interrupt circuit.
8-bit pulse accumulator (event counting mode). COP watchdog timer. Clock monitor.

Parallel ports: Five - one port of 6 individually configurable lines.
(some lines multifunctioned with serial use).
- one port of 8 input lines (function multiplexed with A/D converter).
- one port of 8 output lines.
- one port of 8 configurable lines.
- one port with 1 configurable line, 3 input lines and 4 output lines.

Serial Channel: Enhanced NRZ Serial Communications Interface (SCI) (UART). Serial Peripheral Interface (SPI) (USART). Channel is 8 or 9 bits with selectable baud rates: 75, 150, 300, 600, 1200, 2400, 4800, or 9600 baud. (default: 9600 baud, 8 data, no parity, 1 stop)

A/D Converter: Eight 8-bit channels, 32 clock cycle conversion time.

Interrupts: 15 maskable interrupts.
2 non-maskable interrupts: SWI and XIRQ.
4 non-maskable reset type interrupts.

Memory Expansion: 64 Kbyte direct addressing.

Supply: 50 mw typical at 8 Mhz, non expanded bus mode 5 VDC 10 ma typical (15 ma max)
100 mw typical at 8 Mhz, expanded bus mode 5 VDC 20 ma typical (27 ma max)
1.7ua typical (10ua max) in STOP mode.

This high speed, low power, CMOS computer-on-a-chip is packaged in a 52 pin PLCC, measuring just 0.75 inches on a side. The chip hosts five, user available, 8-bit parallel ports in the single chip mode. There are three when expanded to have a 64K address and data bus. The asynchronous Serial Communications Interface (SCI) supports the system terminal. The second serial channel, the synchronous Serial Peripheral Interface (SPI), is uncommitted and available to the user. There are also included: a the timer subsystem, a computer operating properly (watchdog) circuit, a fast 8-bit 8-channel A/D converter, and the 1/2K bytes of internal EEPROM.

With an 8 Mhz crystal, operating in single chip mode, the F68HC11 draws less than 10 mA (typ). It supports a lower current wait for interrupt mode, and a stop mode that draws 1.7 uA (typ).

The built-in high level language and operating system, based on the FORTH 83 Standard, gives easy machine access to the user interactively. The vocabulary contains 231 words. Applications can be very compact. Although internal RAM is limited, non trivial applications can be programmed by defining a word in RAM and then moved to EEPROM, using the built-in EEPROM support functions. This frees the RAM to accept other programming.

All the words of FORTH 83 Standard Required Word Set are included, plus all of the standard's Extension Word Sets, all of the [illegible]

Reference Words Set, plus a number of words not associated with the standard, which are part of the operating system. They ease operation in a single chip computer environment. They provide for Headerless Target Compiling, Autostarting of user applications and High Level Interrupts. Hooks are provided for user defined multi-tasking.

The advantage of using the F68HC11 lies the speed of project development and checkout. Even if the F68HC11 is the only chip working in a design, interactive debugging of the hardware and software can be performed.

Max-FORTH V3.3 Words Listing

F88F	TASK	FBE3	(	FE22	@	FDFC	C@
FDEF	!	FDE2	C!	FA0A	2@	F9F4	2!
E7C4	:	E7DA	;	FDCB	+	FDC0	-
FA9B	1-!	FA8C	1+!	FA7D	+!	F887	*
F87D	/	FD43	><	FD4E	SWAP	F9DC	2OVER
F9E8	2SWAP	FD8B	DUP	F9D4	2DUP	FD3B	OVER
FD27	ROT	F906	2ROT	FA6F	PICK	FA29	ROLL
FA49	-ROLL	FD7F	DROP	F9C4	2DROP	FD76	>R
FD6F	R>	FC05	=	FC36	NOT	FC24	0=
F966	D0=	FC1B	0>	FC12	0<	FBF9	U<
FBE4	<	F976	DU<	F96E	D<	F95E	D=
FBDC	>	FDAF	AND	FDA1	OR	FD93	XOR
EF4F	IF	EF43	THEN	EF2D	ELSE	EF23	BEGIN
EF15	UNTIL	EF03	REPEAT	EEFB	WHILE	EEED	AGAIN
EF15	END	EE55	DO	EE82	LOOP	EE61	+LOOP
FAE2	K	FADA	J	FAD2	I	FAD2	R@
EEA3	LEAVE	FEB6	EXIT	F8F8	KEY	F8F0	EMIT
F8E8	?TERMINAL	FBD4	S->D	FBC5	ABS	F93E	DABS
FBA9	MIN	F92A	DMIN	FB99	MAX	F918	DMAX
F75F	SPACES	FA19	DEPTH	F6DA	CR	F6BA	TYPE
F6AE	COUNT	F686	-TRAILING	FC7C	1+	FC71	2+
FC66	1-	FC5B	2-	FC4C	2/	FC41	2*
F9A4	D+	F97E	D-	F903	D2/	F839	/MOD
F831	MOD	F825	*/MOD	F81B	*/	FB35	UM*
FB01	UM/MOD	FAF6	NEGATE	F952	DNEGATE	E802	CONSTANT
E7F2	VARIABLE	E810	2CONSTANT	E7E8	2VARIABLE	FAEA	EXECUTE
FD66	SP@	F7B5	CMOVE>	F7DF	CMOVE	FEB6	;S
E846	CODE-SUB	E83A	CODE	E82A	END-CODE	E81C	USER
F596	.	F5A8	.R	F5B4	D.	F58E	U.
F59F	U.R	F5BE	D.R	F615	#S	F623	#
F604	SIGN	F5F4	#>	F5EA	<#	F586	?
F6E8	EXPECT	E70E	QUERY	F646	BL	F554	STATE
F551	CURRENT	F54E	CONTEXT	F54B	SCR	F548	BLK
F506	DP	F52A	OFFSET	F53F	FLD	F53C	DPL
F533	>IN	F4FA	BASE	F4E5	S0	F4EE	TIB
F539	#TIB	F536	SPAN	F56B	C/L	F108	FIRST
F100	LIMIT	F563	PAD	EF83	HERE	EF7B	ALLOT
EF6F	,	EF63	C,	F777	SPACE	FD81	?DUP
EDF5	TRAVERSE	EF5B	LATEST	EC65	COMPILE	EC5D	[
EC52	]	F40B	HEX	F4C2	DECIMAL	EC3A	;CODE
EC32	<BUILDS	EC22	DOES>	EBF2	."	EC15	.(
F78F	FILL	F787	ERASE	F77F	BLANK	F5DC	HOLD
F224	WORD	F45B	CONVERT	F405	NUMBER	EBC1	FIND
EA53	ID.	E896	CREATE	EAD3	[COMPILE]	EAAF	LITERAL
E75E	INTERPRET	EA00	IMMEDIATE	E864	VOCABULARY	E9F6	FORTH
E9EE	EDITOR	E956	ASSEMBLER	E90C	DEFINITIONS	E9D4	RECURSE
EE3F	>MARK	EF83	<MARK	EE31	>RESOLVE	EE27	<RESOLVE
E854	:CASE	EBA9	'	EBA9	[']	FC5B	LFA
FC71	>BODY	EE1F	CFA	EE10	NFA	EDE8	PFAPTR
E738	LOAD	E728	THRU	EFB7	-->	F186	UPDATE
F16C	EMPTY-BUFFERS	F0B2	SAVE-BUFFERS	F0B2	FLUSH	F09A	(LINE)
F090	.LINE	F060	>L	F122	BUFFER	F0D2	BLOCK
F394	B/BUF	EDB2	H/C	ED00	HWORD	ECEA	AUTOSTART
E936	UNDO	E943	FORGET	F1EC	DUMP	F353	.S
ECB2	WORDS	EFFD	TRIAD	EFCE	INDEX	F02B	LIST
E6EF	QUIT	EBFE	ABORT"	E625	ABORT	FEBC	COLD
FD17	BRANCH	FD05	?BRANCH	FAA7	ATO4	ED44	EEWORD
F398	EEMOVE	F300	EEC!	F88F	FORTH-83		

The F68HC11 is ideal for applications where program development time is critical. Its low power CMOS nature is perfect for solar and battery power applications.

The built-in subsystems and watch dog timer, makes it the natural choice for real time embedded applications. Dedicated applications that require small physical size and interactive field testing and servicing also can benefit from it's features.

The F68HC11 is foundrial for New Micros, Inc. by Motorola, Inc.. Software manual UM-Max and NMIX-0023 assembled and tested board level development system available from New Micros, Inc.. The F68HC11 with internal Max-FORTH and manuals are available exclusively from New Micros, Inc., 1601 Chalk Hill Rd., Dallas, TX 75212. Technical Assistance (214) 339-2204. Sales (800) 255-4664

The "100 Squared"TM
F68HC11 Version

NMIX-0023

Specifications

The "100 Squared"TM is a complete single board computer with operating system and high level language development tools included in the on-board Max-FORTH 8K-byte ROM. In the retail configuration it is provided with 8K bytes RAM and the Max-FORTH ROM. For program development an optional serial cable (Order # 23CBL232-4) can be used connect to an RS-232 terminal (or better yet, host computer with mass storage for editing and downloading of source code). Power must be supplied from a 7 to 18 VAC or 7 to 24 VDC source (Order # 23PS), or from an external regulated 5 VDC source. UM-Max and UM-100HC11 Manuals are included.

Size: 100 mm by 100 mm active component area. 100 mm by 60 mm prototyping area.

Processor: MOTOROLA 68HC11

CPU: Enhanced 6800/6801

Language: MAX-FORTH, 231 words: A superset of the FORTH-83 Standard including:
- * The 83 Standard Word Set.
- * The 83 Standard Double Number Extension Word Set
- * The 83 Standard System Extensions Word Set
- * All the 83 Standard Controlled Reference Words
- * Single-Chip System Extensions

Counter/Timers: Enhanced 16-bit timer system: Four stage programmable prescaler, three 16-bit input-capture registers, five 16-bit output-compare registers, real time interrupt circuit.
8-bit pulse accumulator (event counting mode).
COP watchdog timer.
Clock monitor.

Parallel ports: Three ports:
- one port of 6 individually configurable lines (some lines multifunctioned with serial use).
- one port of 8 input lines (function multiplexed with A/D converter).
- one port with 1 configurable line, 3 input lines and 4 output lines.

Serial Channels: Enhanced NRZ Serial Communications Interface (SCI) (UART). Serial Peripheral Interface (SPI) (USART). RS-232 channel is 8 or 9 bits with selectable baud rates: 75, 150, 300, 600, 1200, 2400, 4800, or 9600 baud (default: 9600 baud, 8 data, no parity, 1 stop) brought to 4 .1" pins, cable optional.

A/D Converter: Eight 8-bit channels, 32 clock cycle conversion time.

Interrupts: 15 maskable interrupts.
2 non-maskable interrupts: SWI and XIRQ.
4 non-maskable reset type interrupts.

Memory Expansion: 64 Kbyte direct addressing.

Sockets: Three 28-Pin JEDEC standard sockets for 28-pin ROM/RAM/EPROM/EEPROM devices.
Accepts: 2764, 2864, 4464, 6264, 27128, 27256, 62256, etc.

Expansion: 34-pin 0.1" dual-in-line pads on board: All data lines (8), all address lines(16) pwr, gnd, R/W, E, AS, OE, INT, RST, MEMDIS (on board memory disable), 1 N.C.. Verticle Stacking Connector optional.

Address Decoder: 74HC138 with jumper selectable options for 8 or 32 KByte devices.

Supply: Power 100 mw typical

Voltage Optional Wall Transformer (7 to 18 VAC/DC) or External Regulated 5 VDC.
Onboard rectification (VM08)
Onboard regulation (7805)
Onboard DC/DC for negative RS-232 supply (7660)

Current 20 ma typical

The "100 Squared"TM
F68HC11 Version

NMIX-0022

Specifications

The "100 Squared"TM is a complete single board computer with operating system and high level language development tools included in the on-board Max-FORTH 8K-byte ROM. In the retail configuration it is provided in a metal case, with power supply, 512 bytes EEPROM, 8K bytes RAM, and the Max-FORTH ROM, ready in all respects to connect to an RS-232 terminal for immediate program development (or better yet, host computer with mass storage and download software). OEM versions of the system are provided at reduced cost without case or power supply.

Size: 100 mm by 100 mm active component area. 100 mm by 60 mm prototyping area.

Processor: MOTOROLA 68HC11

CPU: Enhanced 6800/6801

Language: MAX-FORTH, 231 words: A superset of the FORTH-83 Standard including:
- The 83 Standard Word Set.
- The 83 Standard Double Number Extension Word Set
- The 83 Standard System Extensions Word Set
- All the 83 Standard Controlled Reference Words
- Single-Chip System Extensions

Counter/Timers: Enhanced 16-bit timer system: Four stage programmable prescaler, three 16-bit input-capture registers, five 16-bit output-compare registers, real time interrupt circuit.
8-bit pulse accumulator (event counting mode).
COP watchdog timer.
Clock monitor.

Parallel ports: Five ports:
- one port of 6 individually configurable lines.
  (some lines multifunctioned with serial use).
- one port of 8 input lines (function multiplexed with A/D converter).
- one port of 8 output lines.
- one port of 8 configurable lines.
- one port with 1 configurable line, 3 input lines and 4 output lines.

Serial Channel: Enhanced NRZ Serial Communications Interface (SCI) (UART). Serial Peripheral Interface (SPI) (USART). RS-232 channel is 8 or 9 bits with selectable baud rates: 75, 150, 300, 600, 1200, 2400, 4800, or 9600 baud.
(default: 9600 baud, 8 data, no parity, 1 stop)

A/D Converter: Eight 8-bit channels, 32 clock cycle conversion time.

Interrupts: 15 maskable interrupts.
2 non-maskable interrupts: SWI and XIRQ.
4 non-maskable reset type interrupts.

Memory Expansion: 64 Kbyte direct addressing.

Sockets: Three 28-Pin JEDEC standard sockets for 28-pin ROM/RAM/EPROM/EEPROM devices.
Accepts: 2764, 2864, 4464, 6264, 27128, 27256, 62256, etc.

Expansion: 34-pin 0.1" dual-in-line connector: All data lines (8), all address lines(16) pwr, gnd, R/W, E, AS, OE, INT, RST, MEMDIS (on board memory disable), 1 N.C.

Address Decoder: 74HC138 with jumper selectable options for 8 or 32 KByte devices.

Supply: Power 100 mw typical
Voltage Supplied Wall Transformer (7 to 18 VAC/DC) or External Regulated 5 VDC.
Onboard rectification (VM08)
Onboard regulation (7805)
Onboard DC/DC for negative RS-232 supply (7660)
Current 20 ma typical

Case: Aluminum ribbed extrusion with steel back panel, all anodized black. Front Panel heavy aluminum plate. Interior cut to hold Euro format 160 x 100 mm cards. Rack mountable in 19" Euro system.

The "100 Squared"TM
F68HC11 Version

NMIX-0021

Specifications

The "100 Squared"TM is a complete single board computer with operating system and high level language development tools included in the on-board Max-FORTH 8K-byte ROM. In the retail configuration it is provided in a metal case, with power supply, 512 bytes EEPROM, 8K bytes RAM, and the Max-FORTH ROM, ready in all respects to connect to an RS-232 terminal for immediate program development (or better yet, host computer with mass storage and download software). OEM versions of the system are provided at reduced cost without case or power supply.

Size: 100 mm by 100 mm active component area. 100 mm by 60 mm prototyping area.

Processor: MOTOROLA 68HC11

CPU: Enhanced 6800/6801

Language: MAX-FORTH, 231 words: A superset of the FORTH-83 Standard including:
- * The 83 Standard Word Set.
- * The 83 Standard Double Number Extension Word Set
- * The 83 Standard System Extensions Word Set
- * All the 83 Standard Controlled Reference Words
- * Single-Chip System Extensions

Counter/Timers: Enhanced 16-bit timer system: Four stage programmable prescaler, three 16-bit input-capture registers, five 16-bit output-compare registers, real time interrupt circuit.
8-bit pulse accumulator (event counting mode).
COP watchdog timer.
Clock monitor.

Parallel ports: Three ports:
- one port of 6 individually configurable lines (some lines multifunctioned with serial use).
- one port of 8 input lines (function multiplexed with A/D converter).
- one port with 1 configurable line, 3 input lines and 4 output lines.

Serial Channel: Enhanced NRZ Serial Communications Interface (SCI) (UART). Serial Peripheral Interface (SPI) (USART). RS-232 channel is 8 or 9 bits with selectable baud rates: 75, 150, 300, 600, 1200, 2400, 4800, or 9600 baud.
(default: 9600 baud, 8 data, no parity, 1 stop)

A/D Converter: Eight 8-bit channels, 32 clock cycle conversion time.

Interrupts: 15 maskable interrupts.
2 non-maskable interrupts: SWI and XIRQ.
4 non-maskable reset type interrupts.

Memory Expansion: 64 Kbyte direct addressing.

Sockets: Three 28-Pin JEDEC standard sockets for 28-pin ROM/RAM/EPROM/EEPROM devices.
Accepts: 2764, 2864, 4464, 6264, 27128, 27256, 62256, etc.

Expansion: 34-pin 0.1" dual-in-line connector: All data lines (8), all address lines(16) pwr, gnd, R/W, E, AS, OE, INT, RST, MEMDIS (on board memory disable), 1 N.C.

Address Decoder: 74HC138 with jumper selectable options for 8 or 32 KByte devices.

Supply: Power 100 mw typical
Voltage Supplied Wall Transformer (7 to 18 VAC/DC) or External Regulated 5 VDC.
Onboard rectification (VM08)
Onboard regulation (7805)
Onboard DC/DC for negative RS-232 supply (7660)
Current 20 ma typical

Case: Aluminum ribbed extrusion with steel back panel, all anodized black. Front Panel heavy aluminum plate. Interior cut to hold Euro format 160 x 100 mm cards. Rack mountable in 19" Euro system.

The "Max-FORTH"™, FORTH-83 Standard Language

F68HC11 Internal ROM and XC68HC11 External ROM Versions

Specifications

"Max-FORTH"™ for the 68HC11 is a ROM based operating system and language. When combined with a 68HC11, whether in internal or external ROM, "Max-FORTH"™ creates a complete development enviroment. Max-FORTH programs can be written, tested and run under its control.

Although usual development configurations would include external RAM and mass storage (either on the 68HC11 system or on the host terminal), short programs actually can be developed on a 68HC11 system using "Max-FORTH"™ with no extra support (beyond XTAL circuit, pull ups, power and serial I/O connections).

"Max-FORTH"™ produces compact code that is suitable for ROMing or EEPROMing. Both "Headed" and "Target Compiled" code applications can be created. While the compiled programs may not execute as fast as well written assembly language programs, they do compare favorably with the results from other compiled languages. They are usually more compact, more quickly written, and more easily tested in "Max-FORTH"™'s interactive programming environment.

"Max-FORTH"™ closely follows the FORTH-83 Standard in order to be compatible with other FORTH's, also to be easily supported, learned and used. In addition to the FORTH-83 Required Word Set, "Max-FORTH"™ also contains all the FORTH-83 Extension Word Sets, all the Controlled Reference Words and some of the Uncontrolled Reference Words. It also has many single chip specific extensions and operating system words not found in the standard.

"Max-FORTH"™ is available in several versions:

> Version 1.0 is supplied in an external ROM for use on systems such as Motorola's EVB system or the New Micros, Inc. NMIX-0021 single board computer. This version does not have autostart capability. It is otherwise "full featured". Order # Max68HC11V1.0
>
> Version 2.0 is also supplied in external ROM for use on the same type systems. This version will autostart a user program at any 1K boundry, or in the EEPROM. This version is suitable for volume production use, under license agreement. The license agreement requirements to acquire the Version 2.0 ROM's are, therefore, stricter than for Version 1.0. Version 2.0 is essentially identical to Version 3.0 which is supplied in the NMI 68HC11 MCU. Order # Max68HC11V2.0
>
> Version 3.0 is supplied in the internal ROM of the NMI F68HC11 MCU. These chips are manufactured by Motorola for sale by New Micros, Inc. These chips will autostart a user program at any 1K boundry, or in the EEPROM and are suitable for volume production use, under license agreement. Order # F68HC11
>
> Version 4.0, not yet released, will allow the user to target compile up to 5K bytes of program on a 3K byte Max-FORTH kernel, which may be submitted to Motorola for production as an internal single chip ROM code in the MC68HC11A8, under license agreement for the kernel with New Micros, Inc.. Order # not yet assigned

User Manual, Order # UM-Max, is suitab. for use with Versions 1.0, 2.0 and 3.0

WORDS LIST FOR "Max-FORTH"™ V1.0 6/86

The following words list was captured from a "Max-FORTH"™ V1.0 system. To conserve ROM space, only the character count and the non-underlined characters actually exist in the internal ROM, i.e., SPACES and SPACEZ are not differentiated. Unless **WIDTH** is set to 3 (number of characters — maximum is 31) the names entered will be normal length.

```
WORDS
 F893 TASK           EBE0 (              FE26 @              FE00 C@
 FDF3 !              FDE6 C!             FA0E 2@             F9F8 2!
 E7C4 :              E7DA ;              FDCF +              FDC4 -
 FA9F 1-!            FA90 1+!            FA81 +!             F888 *
 F881 /              FD47 ><             FD52 SWAP           F9E0 2OVER
 F9EC 2SWAP          FD8F DUP            F9D8 2DUP           FD3F OVER
 FD28 ROT            F9CA 2ROT           FA73 PICK           FA2D ROLL
 FA4D -ROLL          FD83 DROP           F9C8 2DROP          FD7A >R
 FD73 R>             FC09 =              FC3A NOT            FC28 0=
 F96A D0=            FC1F 0>             FC16 0<             FBFD U<
 FBE8 <              F97A DU<            F972 D<             F962 D=
 FBE0 >              FDB3 AND            FDA5 OR             FD97 XOR
 EF4E IF             EF42 THEN           EF2C ELSE           EF22 BEGIN
 EF14 UNTIL          EF02 REPEAT         EEFA WHILE          EEEC AGAIN
 EF14 END            EE54 DO             EE81 LOOP           EE60 +LOOP
 FAE6 K              FADE J              FAD6 I              FAD6 R@
 EEA2 LEAVE          FEBB EXIT           F8FC KEY            F8F4 EMIT
 F8EC ?TERMINAL      FBD8 S->D           FBC9 ABS            F942 DABS
 FBAD MIN            F92E DMIN           FB9D MAX            F91C DMAX
 F763 SPACES         FA1D DEPTH          F6DE CR             F6BE TYPE
 F6B2 COUNT          F68A -TRAILING      FC80 1+             FC75 2+
 FC6A 1-             FC5F 2-             FC50 2/             FC45 2*
.F9A8 D+             F982 D-             F907 D2/            F83D /MOD
 F835 MOD            F829 */MOD          F81F */             FB39 UM*
 FB05 UM/MOD         FAFA NEGATE         F956 DNEGATE        E802 CONSTANT
 E7F2 VARIABLE       E810 2CONSTANT      E7E8 2VARIABLE      FAEE EXECUTE
 FD6A SP@            F7D9 CMOVE>         F7E3 CMOVE          FEBB ;S
 E846 CODE-SUB       E83A CODE           E82A END-CODE       E81C USER
 F597 .              F5A9 .R             F5B5 D.             F58F U.
 F59F U.R            F5BF D.R            F616 #S             F624 #
 F605 SIGN           F5F5 #>             F5E8 <#             F587 ?
 F6EC EXPECT         E70E QUERY          F647 BL             F555 STATE
 F552 CURRENT        F54F CONTEXT        F54C SCR            F549 BLK
 F507 DP             F52B OFFSET         F540 FLD            F53D DPL
 F534 >IN            F4FB BASE           F4E6 S0             F4EF TIB
 F53A #TIB           F537 SPAN           F56C C/L            F1C9 FIRST
 F1C1 LIMIT          F564 PAD            EF82 HERE           EF7A ALLOT
 EF6E ,              EF62 C,             F778 SPACE          FB85 ?DUP
 EDF4 TRANSVERSE     EF5A LATEST         EC64 COMPILE        EC5C [
 EC51 ]              F4CC HEX            F4C3 DECIMAL        EC39 ;CODE
 EC31 <BUILDS        EC21 DOES>          EBED ."             EC12 .(
 F793 FILL           F788 ERASE          F783 BLANK          F5DD HOLD
 F225 WORD           F45C CONVERT        F406 NUMBER         EBBE FIND
 EADE ID.            E896 CREATE         EACE [COMPILE]      EAAA LITERAL
 E75E INTERPRET      E9FE IMMEDIATE      E864 VOCABULARY     E9F4 FORTH
 E9EC EDITOR         E9E4 ASSEMBLER      E9DA DEFINITIONS    E9D2 RECURSE
 EE3E >MARK          EF82 <MARK          EE30 >RESOLVE       EE26 <RESOLVE
 E854 ;CASE          EBA8 '              EBA4 [']            FC5F LFA
 FC75 >BODY          EE1E CFA            EE0F NFA            EDE7 PFAPTR
 E738 LOAD           E728 THRU           EFB6 -->            F187 UPDATE
 F16D EMPTY-BUFFERS  F0B3 SAVE-BUFFERS   F0B3 FLUSH          F098 (LINE)
 F091 .LINE          F08C >L             F123 BUFFER         F0D3 BLOCK
 F393 B/BUF          EDB1 N/C            ECFF NWORD          ECE9 AUTOSTART
 E934 UNDO           E941 FORGET         F1ED DUMP           F392 .S
 ECB1 WORDS          EFFC TRIAD          EFCD INDEX          F02A LIST
 E6EF QUIT           EBF9 ABORT"         E6E5 ABORT          FEC1 COLD
 FD1B BRANCH         FD09 ?BRANCH        FAAB ATO4           ED43 EEWORD
 F397 EEMOVE         F3BF EEC!           F893 FORTH-83    OK
```

ERRATA

The following errata has been noted in the Max-FORTH REFERENCE MANUAL Preliminary June, 1986.

Page 10, last paragraph. The Data Stack is limited to 16 and the Return Stack to 15 words.

Page 25, 26 and 27. Replace all occurrences of ." with .(and " with) .

Page 130. A word has been added to the language. It is **ATO4** . See supporting sheets for further information on this word.

Page 158. The "head" for **CSP** has been removed. It can not be found in the dictionary.

Page 172. The "head" for **DLITERAL** has been removed. It can not be found in the dictionary.

Page 184. Three words have been added to the language. They are **EEC!** , **EEMOVE** and **EEWORD** . See supporting sheets for further information on these words.

Page 281. The "head" for **DLITERAL** has been removed. It can not be found in the dictionary. Remove reference in compiler section.

Page 291. Three words have been added to the language. They are **EEC!** , **EEMOVE** and **EEWORD** . See supporting sheets for further information on these words. Add reference in memory section.

Page 292. A word has been added to the language. It is **ATO4** . See supporting sheets for further information on this word. Add reference in operating system section.

Page 292. The "head" for **CSP** has been removed. It can not be found in the dictionary. Remove reference in security section.

Page 302. The order of UP , W and IP have been changed to reduce the pain of accidentally trying to clear some memory location with the erroneous entry of xxxx 0 ! . Since W is the least critical of the three variables, accidental clearing should not cause system crash. W is now at $0000, IP at $0002, and UP at $0004. Information associated with each label, otherwise correct, moves with its label.

Page 302. Change KHZ to C/10MS which is the time used for the programming of each EEPROM location, and its default setting to 20000.

Page 303. Change default setting of UFIRST to $D7FC.

Page 308. The EEPROM addresses should be $Bxxx as oppossed to $9xxx.

Page 314. Reset routine heavily modified. Function maintained, except there is no warm download or autostart.

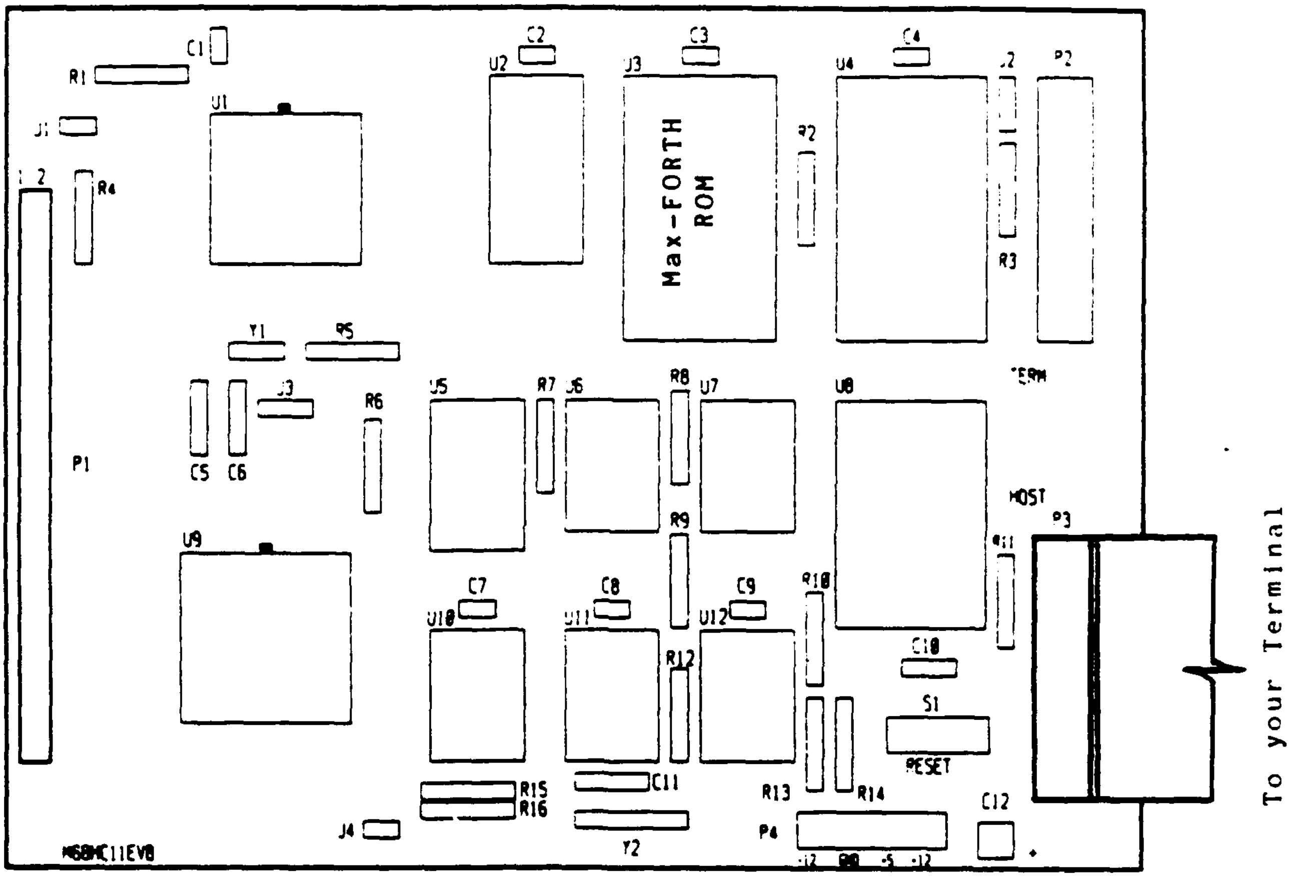
Page 315. Download value for W and IP removed from CNTOP table.

Page 316. Change KHZ to C/10MS which is the time used for the programming of each EEPROM location, and its default setting to 20000.

Page 303. Change default setting of UFIRST to $D7FC.

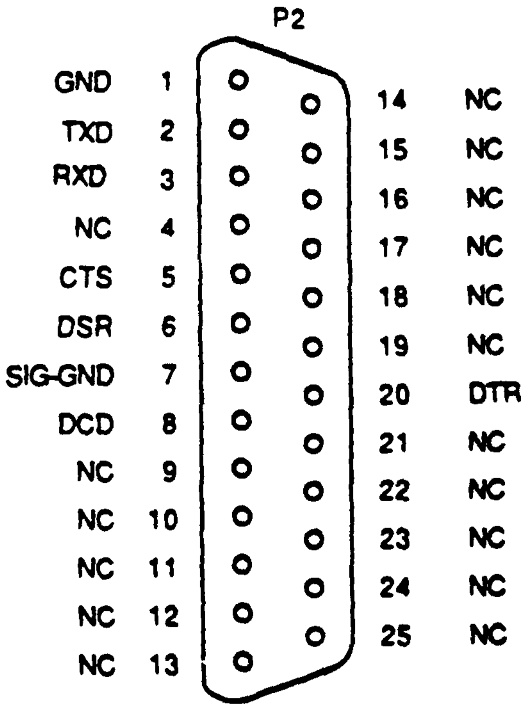
Hardware Configuration

RS-232 Terminal Interconnection



EVB Parts Placement Diagram

To communicate with the Max-FORTH ROM, use the RS-232 terminal near the power connection, labeled as HOST and replace the Buffalo ROM_with the Max-FORTH ROM. Connection to this port may require a null modem which swaps pins 2 and 3. Pins 4 & 5 and pins 6 & 20 may also need to be swapped depending on the particular terminal used.



RS-232 Pin Assignment Diagram for
the Host Port of the EVB

WORDS LIST FOR Max-FORTH V1.0 6/86

The following words list was captured from a Max-FORTH V1.0 system. To conserve ROM space, only the character count and the non-underlined characters actually exist in the internal ROM, i.e., SPACES and SPAXYZ are not differentiated. Unless you set **WIDTH** to three (number of characters—maximum is 31) the names you enter will be normal length.

WORDS

F893	TASK	EBE0	(	FE26	@	FE00	C@
FDF3	!	FDE6	C!	FA0E	2@	F9F8	2!
E7C4	:	E7DA	;	FDCF	+	FDC4	-
FA9F	1-!	FA90	1+!	FA81	+!	F88B	*
F881	/	FD47	><	FD52	SWAP	F9E0	2OVER
F9EC	2SWAP	FD8F	DUP	F9D8	2DUP	FD3F	OVER
FD2B	ROT	F9CA	2ROT	FA73	PICK	FA2D	ROLL
FA4D	-ROLL	FD83	DROP	F9C8	2DROP	FD7A	>R
FD73	R>	FC09	=	FC3A	NOT	FC28	0=
F96A	D0=	FC1F	0>	FC16	0<	FBFD	U<
FBE8	<	F97A	DU<	F972	D<	F962	D=
FBE0	>	FDB3	AND	FDA5	OR	FD97	XOR
EF4E	IF	EF42	THEN	EF2C	ELSE	EF22	BEGIN
EF14	UNTIL	EF02	REPEAT	EEFA	WHILE	EEEC	AGAIN
EF14	END	EE54	DO	EE81	LOOP	EE60	+LOOP
FAE6	K	FADE	J	FAD6	I	FAD6	R@
EEA2	LEAVE	FEBB	EXIT	F8FC	KEY	F8F4	EMIT
F8EC	?TERMINAL	FBD8	S->D	FBC9	ABS	F942	DABS
FBAD	MIN	F92E	DMIN	FB9D	MAX	F91C	DMAX
F763	SPACES	FA1D	DEPTH	F6DE	CR	F6BE	TYPE
F6B2	COUNT	F68A	-TRAILING	FC80	1+	FC75	2+
FC6A	1-	FC5F	2-	FC50	2/	FC45	2*
F9A8	D+	F982	D-	F907	D2/	F83D	/MOD
F835	MOD	F829	*/MOD	F81F	*/	FB39	UM*
FB05	UM/MOD	FAFA	NEGATE	F956	DNEGATE	E802	CONSTANT
E7F2	VARIABLE	E810	2CONSTANT	E7E8	2VARIABLE	FAEE	EXECUTE
FD6A	SP@	F7B9	CMOVE>	F7E3	CMOVE	FEBB	;S
E846	CODE-SUB	E83A	CODE	E82A	END-CODE	E81C	USER
F597	.	F5A9	.R	F5B5	D.	F58F	U.
F59F	U.R	F5BF	D.R	F616	#S	F624	#
F605	SIGN	F5F5	#>	F5EB	<#	F587	?
F6EC	EXPECT	E70E	QUERY	F647	BL	F555	STATE
F552	CURRENT	F54F	CONTEXT	F54C	SCR	F549	BLK
F507	DP	F52B	OFFSET	F540	FLD	F53D	DPL
F534	>IN	F4FB	BASE	F4E6	S0	F4EF	TIB
F53A	#TIB	F537	SPAN	F56C	C/L	F1C9	FIRST
F1C1	LIMIT	F564	PAD	EF82	HERE	EF7A	ALLOT
EF6E	,	EF62	C,	F77B	SPACE	FD85	?DUP
EDF4	TRAVERSE	EF5A	LATEST	EC64	COMPILE	EC5C	[
EC51	]	F4CC	HEX	F4C3	DECIMAL	EC39	;CODE
EC31	<BUILDS	EC21	DOES>	EBED	."	EC12	.(
F793	FILL	F78B	ERASE	F783	BLANK	F5DD	HOLD
F225	WORD	F45C	CONVERT	F406	NUMBER	EBBE	FIND
EADE	ID.	E896	CREATE	EACE	[COMPILE]	EAAA	LITERAL
E75E	INTERPRET	E9FE	IMMEDIATE	E864	VOCABULARY	E9F4	FORTH

E9EC	EDITOR	E9E4	ASSEMBLER	E9DA	DEFINITIONS	E9D2	RECURSE
EE3E	>MARK	EF82	<MARK	EE30	>RESOLVE	EE26	<RESOLVE
E854	:CASE	EBA4	'	EBA4	[']	FC5F	LFA
FC75	>BODY	EE1E	CFA	EE0F	NFA	EDE7	PFAPTR
E738	LOAD	E728	THRU	EFB6	-->	F187	UPDATE
F16D	EMPTY-BUFFERS	F0B3	SAVE-BUFFERS	F0B3	FLUSH	F09B	(LINE)
F091	.LINE	F06C	>L	F123	BUFFER	F0D3	BLOCK
F393	B/BUF	EDB1	H/C	ECFF	HWORD	ECE9	AUTOSTART
E934	UNDO	E941	FORGET	F1ED	DUMP	F352	.S
ECB1	WORDS	EFFC	TRIAD	EFCD	INDEX	F02A	LIST
E6EF	QUIT	EBF9	ABORT"	E6E5	ABORT	FEC1	COLD
FD1B	BRANCH	FD09	?BRANCH	FAAB	AT04	ED43	EEWORD
F397	EEMOVE	F3BF	EEC!	F893	FORTH-83		

NAME : ATO4

PRONUNCIATION : "assembly-to-fourth"

VERSION : NMI NMIDR

STACK NOTATION : (--- n)

# ARGUMENTS IN : 0

# ARGUMENTS OUT: 1

RETURN STACK :

DICTIONARY :

PAD :

INPUT :

OUTPUT :

GROUP : Operating System

ATTRIBUTE :

STANDARD :

SHORT DEFINITION: Returns address of subroutine call to high level word as indicated in D register.

LONG DEFINITION: Execution returns the address of a machine code subroutine which can be used as the object of a JSR instruction to call a high level word from an assembly language routine, such as an interrupt process. The processor D register must contain the address of the CFA of the word to be executed. The processor Y register must point to free RAM since the high level word will use it as the Data Stack pointer. Program control will be returned to the instruction following the JSR after completion of the specified high level word.

NAME : **EEMOVE**

PRONUNCIATION : "e-e-move"

VERSION : NMI NMIDR

STACK NOTATION : (addr1 addr2 u ---)

# ARGUMENTS IN : 3

# ARGUMENTS OUT: 0

RETURN STACK :

DICTIONARY :

PAD :

INPUT :

OUTPUT :

GROUP : Memory

ATTRIBUTE :

STANDARD :

SHORT DEFINITION: Moves towards high memory the u bytes at addresses addr1 and addr2. addr2 should be in EEPROM.

LONG DEFINITION: Move the u bytes at addresses addr1 to addr2. The byte at addr1 is moved first, proceeding toward high memory. If u is zero, nothing is moved. addr2 should be in EEPROM. EEC! is used to accomplish this function. See EEC!

NAME : **EEWORD**

PRONUNCIATION : "e-e-word"

VERSION : NMI NMIDR
STACK NOTATION : (---)

# ARGUMENTS IN : 0

# ARGUMENTS OUT: 0

RETURN STACK :

DICTIONARY : If not headerless, the complete definition, or if headerless, the code field and parameter field, of latest definition move from code to EEPROM dictionary.

PAD :

INPUT :

OUTPUT :

GROUP : Memory

ATTRIBUTE :

STANDARD :

SHORT DEFINITION: Moves code of last defined word from the codes memory to the EEPROM memory.

LONG DEFINITION: Latest defined word's complete definition if not headerless, or the code section if headerless is lifted from codes dictionary, relinked and placed in the EEPROM dictionary. Dictionary pointers are re-adjusted accordingly.

NAME : **EEC!**

PRONUNCIATION : "e-e-c-store"

VERSION : NMI NMIDR

STACK NOTATION : (16b addr ---)

# ARGUMENTS IN : 2

# ARGUMENTS OUT: 0

RETURN STACK :

DICTIONARY :

PAD :

INPUT :

OUTPUT :

GROUP : Memory

ATTRIBUTE :

STANDARD :

SHORT DEFINITION: Stores the least significant byte of 16b into addr in EEPROM.

LONG DEFINITION: The addr in EEPROM is erased, then least significant 8 bits of 16b are programmed into the byte at addr. The time period of the programming is set by C/10MS and Timer Output Capture for both the erasure and programming cycles.

TOUR OF THE 68HC11 Max-FORTH MEMORY MAP

A first step in preparing to use a dedicated computer system is to learn the system's memory map. This is an easy task with the use of the Max-FORTH **DUMP** command. Remember, **DUMP** displays one or more lines each containing 16 memory locations (8 words) which are most easily read when in hex mode, so begin by entering **HEX** .

Starting in low memory, examine the contents of the first three words (six memory locations—one byte or two hex digits per location which means two locations per word) by entering **0 6 DUMP** . (This, of course, displays a full dump-line of 16 memory locations—8 words, $0000-$000F. Leading zeros at each location are not displayed.) Examine the first three words ($0000-$0006), ignoring the others for the moment. From the manual's Appendix C (IMPORTANT: see errata for corrections), it is seen thet these locations contain the system variables of Max-FORTH, namely:

W (Word pointer, $0000-$0001);
IP (Instruction Pointer, $0002-$0003); and
UP (User Pointer, $0004-$0005).

Any two bytes of memory reserved to hold only the address of a particular item (word, block, stack, etc.) is refered to as a pointer. Thus W and IP are pointers whose contents change every time forth executes a new definition. Programmers rarely have need to use W or IP . Advanced programmers might use them in code level definitions when making multitaskers, etc.. UP points to the USER area. By modifying UP to point to another area, the values that user variables return will be a different set. This again is useful in multitaskers. Be sure that the new USER area is initialized with its set of values before changing UP to point to it, otherwise, the systems will crash and it will be difficult to recover.

> It is worth mentioning that if you experience a system crash from which it is difficult to recover with a simple reset, a <CTRL>-G key combination in the Serial Communications Interface input register will force a cold reset. (<CTRL>-G is the ASCII "BELL" character, $07 hex.) By entering <CTRL>-G and hitting reset again, you should recover from any badly modified variable, blown dictionary links, crashed processor, or similar problem. A bad autostart, however, can only be rectified by removing or disabling the offending memory device.

Notice that UP points to the current USER memory area starting at $0006. To examine this area enter **6 74 DUMP** . The first four words ($0006, $0008, $000A and $000C) are zero'ed by the cold download. They represent **DNLINK** , **UPLINK** , **PRIORITY** and **RPSAVE** which are user variables reserved for multitasking. These variables are not used by the Version 1.0 Max-FORTH implementation.

The following two words ($000E and $0010) are the default stack values, **R0** and **S0**, assigned by **ABORT** . They can be modified to point into RAM if larger stack areas are desired. They will not be put into effect until **ABORT** is executed. Normally, the stacks should stay in internal RAM. **TIB**, **PAD**, and the dictionary areas can be moved to external RAM which will allow room for 28 Return Stack words and 39 Data Stack words.

The next two word locations ($0012 and $0014), KEY-CB-PTR and EMIT-CB-PTR, point to control blocks that determine how the built-in I/O routines work. Alternate control blocks can be constructed to handle alternate I/O devices without modifying the I/O routines. Each control block is six bytes long and contains the following:

1) a two-byte address of the I/O device status register to be read,

2) a one-byte mask with which the status value read will be AND'ed to screen out extraneous bits,
3) a one-byte mask with which the AND'ed result will be XOR'ed to obtain a zero result when the device is ready for transfer, and
4) a two-byte address of the I/O device data register.

These pointers (KEY-CB-PTR and EMIT-CB-PTR) are initialized to point to the beginning of the default Serial Communications Interface control blocks. The pointer can be modified to point to control blocks that handle LCD displays and keypad controllers or alternate serial chips, etc. to redirect system I/O.

The next three word locations ($0016, $0018 and $001A) are the actual machine level vectors to the I/O subroutines that provide the functions mentioned above. If a control block solution is not sufficient, you may write your own machine code subroutine. Its address is installed in UKEY , UEMIT or U?TERMINAL as appropriate.

Following is the pointer (at $001C) to the Terminal Input Buffer, **TIB**, and normally contains an address pointing to the RAM just beyond the bottom of the Return Stack. It is limited by UC/L ($001E) to 16 characters. **TIB** can be moved to external RAM and made large by modifying these USER variables.

The USER variable which indicates that a COLD reset has been performed is called CLD/WRM ($0020). As long as it contains a $A55A pattern and there is no <CTRL>-R in the SCI, the system will not do a cold download of system variables at every reset.

PAD in most FORTH systems is a fixed distance above the dictionary pointer, but in Max-FORTH that would be impossible with the limited internal RAM, so **PAD** is restricted to be at a "fixed" location, which can be "moved" by changing UPAD ($0022). Since the user may work "above" **PAD** with temporary variables and the system works "below" **PAD** when doing output number conversions, ensure that **PAD** has free RAM on both sides of its new setting.

BASE, the current number base, is kept in the next location ($0024), and performs as per most other FORTH's.

The next location ($0026) is used in conjunction with the Timer Output Capture 5 in EEPROM programming words to time the number of cycles to program the EEPROM. It is called C/10MS for "Cycles Per 10 Milliseconds delay". You should decrease this value from the default decimal value of 20000 if your system has a clock slower than 2 Mhz.

The line termination character can be changed from a carriage return to any other character by modification of the next location ($0028). Similarly the back space character can be changed from a back space to a delete by modifying the following location ($002A).

The Dictionary Pointer, **DP** , and the heads dictionary pointer follow ($002C and $002E). The location after them ($0030) is the EEPROM dictionary usage pointer that is added to when a word is moved into EEPROM with **EEWORD** . The flag that controls headerless code generation follows ($0032).

Additional standard FORTH USER definitions follow ($0034 through $0070). Following are four USER variable locations ($0072, $0074, $0076 and $0078) which are reserved for mass storage requirements. Only the first location ($0072) has any reference in the internal code. It is used to set the base address of the RAMDSK mass storage simulator. To use RAMDSK, which is pointed to by the default setting of UR/W ($0056), set the first location to the base address of free RAM (ie: **C400 72 !**) and clear the buffers (which are at default locations $D7FC through $E000) (ie: **EMPTY-BUFFERS**). The RAM screens can then be listed, loaded and saved, etc..

This ends the USER area while the default location of the dictionary begins at $007A. When a cold reset occurs, **TASK** is moved to the beginning of the dictionary. New entries in the dictionary will build up from TASK ($0083 and up).

The Data Stack is above the dictionary. It grows down as more entries are added ($00C7 and down). There aren't many locations between the dictionary and the stack, so some care should be exercised. Needless to say, running the stack into the dictionary or vice versa is a very bad idea.

The dictionary can be moved after a cold reset simply by forgeting task and storing the address of the new dictionary RAM in DP (ie: **FORGET TASK HEX C004 DP !**).

The area immediately above the data stack ($00C8 down from $00CF) is used by the system numeric output routines. The usage builds down from just below **PAD** till the number of output characters for a given print or "dot word" is reached. (Note: only 8 digit numbers are provided for in the default setting.) (The two lowest locations are used as "secret" storage places for the settings of **BLK** ($00C8) and **>IN** ($00CA) after an error message is generated so you can look up the location of the error on mass storage after the crash. Better get them before you print any numbers after a crash...)

The beginning of **TIB** and **PAD** are both at the same location ($00D0), the idea is to prevent **PAD** from being used in internal RAM if **TIB** is in use. Sixteen bytes are reserved for **TIB** ($00D0 through $00DF).

The Return Stack occupies the upper end of internal RAM ($00E0 through $00FF).

The next location of interest in the 68HC11 map is the first location that an autostart pattern can be located ($0400). Versions 2.0 and greater do autostart pattern searches in EEPROM and then on every 1K boundary starting there and on.

Since reset puts the register block of the 68HC11 at $1000 in the "middle of things", the Max-FORTH operating system moves them to $9000. (Originally it was intended to put them at $B000, but the EVB has a special write latch from $A000 to $BFFF that precluded that.) The write that moves the register block will still affect external memory when the move is performed, so always watch out for location $103D which will have $09 stored into it with every reset. The registers are moved into $9000-$903F. See the appropriate Motorola documentation for their description. To look at them, enter **9000 40 DUMP** .

The EEPROM is the next natural occurrence in the memory map ($B600 through $B7FF). The first word location ($B600) may hold the autostart pattern. This location is the first high level autostart location checked after power up. If the first location is the autostart pattern, the next ($B602) is the address of the CFA (code field address) of the word to be autostarted. If interrupts are not used, the next 247 bytes of EEPROM are unassigned. If interrupts are to be used, the particular interrupt enable must have a machine code program at its EEPROM target location. Starting at location $B7BF and thereafter for each interrupt vector in ROM, three bytes are reserved. They will normally contain a jump instruction to another portion of memory which performs the appropriate interrupt function. It can return to the interrupted program by doing an RTI at its completion.

At the very end of EEPROM, a word location ($B7FE) is checked by the reset routine for an autostart pattern which means that there is a machine code instruction in the previous three bytes ($B7FB) to be executed at power-up before the write-once-only registers are modified. If that autostart pattern is found, the system jumps to subroutine to the code address. It will normally contain a jump instruction to another portion of memory which sets the OPTIONS and INIT registers as desired. It can reenter the start up procedures by doing an RTS at its completion. To look at the EEPROM, enter **B600 200 DUMP** .

(On the Motorola EVB, the area from $C000 to $DFFF is RAM. This can be used for mass storage buffers and screens, or program space or for **TIB** and **PAD** etc.)

The Max-FORTH ROM runs from $E000 through $FFFF and contains the heads of Max-FORTH, followed by non-runtime words, followed by the runtime kernel. The **DUMP** command can be used to explore any portion of the ROM. It will be noted that the heads are made up mostly of ASCII characters and pointers. The non runtime codes are mostly pointers. The runtime codes are a mix of pointers, machine code and miscellaneous tables etc..

DOWN LOADING SOURCE CODE

Development with the Max-FORTH system will normally be done using a host computer. (One alternative would be to add sufficient RAM and use the internal RAMDSK. Another alternative would be hooking up mass storage to the system and writing a user mass storage handler that would replace RAMDSK in the USER variable UR/W. Either method requires you to live with the limited FORTH line editors available or to write your own.)

Source text can be edited on a host PC with a familiar editor. The code can be down loaded and tested a portion at a time using a communications package. This can be continued until the program is complete. PC's are usually graced with printer, etc., so all in all, using a PC as a host can save development time.

Any editor that produces an ASCII text file (ie: non-document mode) should be suitable. Communications packages should be able to handle 9600 baud transfers (or you have to run the XTAL slower) and should have file send and capture capabilities.

The file send program should have a wait-for-echo'ed-character setting and a wait-at-end-of-line-for-return-character. The character to be waited for at the end of line should be a LINE FEED, ASCII $0A (<CTRL>-J).

In Microstuff's Crosstalk these settings are the CWait Echo and LWait CHaracter ^J . With those settings, a smooth down load of a code file by the SEnd command should be possible.

Be sure to watch the down load as it occurs to spot any error messages returned from the Max-FORTH system.

USING EXTERNAL MEMORY

The internal RAM and EEPROM of the 68HC11 is limited. Many applications will require more RAM for program development. Using a Terminal Input Buffer longer than 16 characters is also desireable.

The following brief listing moves the internal pointers for **TIB** , **PAD** and the dictionary for using external memory. The input line is also extended to 80 (decimal) characters. In this example, the memory is located at $C000 as in the case of the Motorola EVB. Other addresses for RAM can be accomodated by changing the listed $Cxxx constants accordingly.

```
HEX
C004 1C !
( TIB MOVE )
50 1E !
( C/L CHANGE )
C060 22 !
( PAD MOVE )
FORGET TASK
C080 DP !
( DICTIONARY MOVE )
```

Note that the first four locations at $C000 are left unassigned so that an autostart pattern can be placed there ($C000 is a 1K boundary) after the program is complete. If a Battery RAM were used in that socket, the program would be saved through power down. If autostarting is not done, the address of the main routine to be run can be **EXECUTE**'ed from the terminal, or it can be run by its name if the dictionary pointers are intact. To be able to use the Battery RAM words' heads, the USER area must be battery backed, otherwise, the dictionary pointers must be relinked. If the program is ROM'ed later, it should allow for the locations $C004 through $C080 to become "unwritable".

The USER area can also be moved to give more room in the data stack, although, it is not normally necessary to do so.

AFTER THE CRASH

You may occasionally experience a system crash, from which it is difficult to recover. A simple reset may give you a Max-FORTH Vx.x prompt and a carriage return may even give you an "OK", however, Max-FORTH doesn't seem to understand anything typed in that is over 1 character long. (A "no Max-FORTH prompt" condition probably means an autostart problem or hardware failure.)

This is usually the result of a blown dictionary link. If the linked list of the dictionary is opened (ie: one of the link fields is erased, written over, badly manipulated, etc.) Max-FORTH will accept your input and then go on a wild goose chase looking for a match with your word all over memory until it finds a $0000 in a link field which indicates "end of dictionary" (which, coincidentally, may never be found). It will either lock up or echo the entry with a question mark. This can be simulated by doing: **0 LATEST PFAPTR LFA !** . You might want to wait a minute before you try this, however, until you know how to recover ...

Another problem can happen when the dictionary pointer gets out of RAM if, for instance, you did a **HEX E000 DP !** (which puts the dictionary pointer in ROM). The problem is that the outer interpreter takes your input line in **TIB** and parses a word at a time out of it. It moves the parsed word to **HERE** which is an exceptionally bad idea if **DP** is in ROM. ROM is funny that way, no matter what you write to it, it never takes it. Of course when the dictionary search compares the dictionary list with the word at **HERE**, result has nothing to do with the characters (and number of them) of the word you typed in. It prints junk and a question mark.

A similar thing can happen if a word is moved to EEPROM with **EEWORD** and then forgotten. Guess where **DP** ends up?

Putting a bad value in UP can be interesting. Changing UEMIT or UKEY to some erroneous values can have some extremely quiet results, too.

If any of the above conditions occur, the only way to recover is to power down and re-power or to force a cold reset.

> A <CTRL>-G key combination in the Serial Communications Interface input register will force a cold reset. (<CTRL>-G is the ASCII "BELL" character, $07 hex.) Entering the <CTRL>-G and hitting reset again should recover from any badly modified variable, blown dictionary links, crashed processor, etc. problems.

A bad autostart, however, can only be rectified by disabling the offending memory device. If the autostart pattern is in the part's EEPROM, it may be erased using a Motorola EVM for recovery.

1. AUTOSTARTING

All Max-FORTH versions have autostarting capabilities. Even Version 1.x, advertised to the contrary, autostarts. It has been modified, however, to find itself before looking for any user programs. (Version 1.x was created so the product could be safely distributed, without a signed license agreement, in ROM's that were copyable. By reducing the autostart capabilities, the part is not desireable for mass production.)

The autostart capability is the means by which the language or the designated user program is started at power-up or reset. It makes up the better part of the Max-FORTH's operating system. In Version 2.x and greater releases, there are three distinct actions in the autostart sequence.

In brief summary, the first autostart attempt allows a machine code routine to alter the INIT, OPTION and TMSK2 registers immediately after reset. The second autostart attempt checks the beginning of EEPROM and then all 1K memory map boundaries for a temporary autostart pattern. The third autostart attempt checks the beginning of EEPROM and all of memory map 1K boundaries for a primary autostart pattern. Each will be discussed in detail.

The first autostart occurs within a few cycles after reset. It looks at the EEPROM at $B7FE for a primary autostart pattern, $A55A. If that pattern is found, a JSR to location $B7FB is executed. At that location, three empty bytes allow the insertion of a jump instruction which passes control to a designated machine code routine. The routine can alter the INIT (REG 3-Ø & RAM 3-Ø bits), OPTIONS (IRQE, DLY, CR1 & CRØ bits) and TMSK2 (PR1 & PRØ bits) registers which must be established in the first 64 clock cycles following reset. Since they can only be written once and only in the 64 cycle period, any settings made in that routine will not be affected by later system initialization action. Normally, the machine code routine will return to the system initialization and autostart functions by performing an RTS. When the rest of the system initialization is not required, control could be maintained by user machine code programs.

The second autostart attempt checks the beginning of EEPROM, $B6ØØ, and then searches all 1K memory map boundaries, starting at location $Ø4ØØ and continuing through $FCØØ, for a temporary autostart pattern, $A44A. If the temporary autostart pattern is found at a given location $XXØØ, the next word location following the autostart pattern, $XXØ2, is taken to be the CFA of the word to be run. This word could be anywhere in the memory map, although this program will typically be in the memory device positioned at the boundary. The autostarted routine can either be a high level definition or a code definition. The unique advantage of this autostart attempt and the temporary autostart pattern is code produced may not be preemptable and could cause high-level interrupt and multitasking failures.

The SP register is used for the FORTH Return Stack Pointer. No values can be taken from, or left on, the machine stack by the end of the definition. While a value can be temporarily pushed on the stack, it must come off before the return to NEXT.

The A, B, D, X and CC registers can be used without consequence.

Since the address of NEXT varies from version to version and revision to revision, **CODE-SUB** is again used in the next example. Using a **CODE** definition would require providing the address of NEXT for all versions and revisions. If a new revision was released with a different NEXT, this example may not work. Using **CODE-SUB** in defining a word here, assures that this example will be able to return control to FORTH easily in all versions and revisions.

The example below reads the first four A/D channels and places them on the stack.

```
CODE-SUB READ-A/D-CH0-3
         CE C, 9030 , ( LDX $9030 )
         ( SET ADCTL FOR MULT READINGS, STRT CONV )
         86 C, 10 C, ( LDAA # 10 )
         A7 C, 00 , ( STAA 0,X , $9030 )
         ( WAIT UNTIL CCF SET )
( SPIN ) 1F C, 00 C, 80 C, FC C, ( BRCLR 0,80,SPIN )
         4F C, ( CLRA )
         ( TAKE DATA, OPEN STACK, STORE DATA )
         E6 C, 01 C, ( LDAB 1,X )
         18 C, 09 C, ( DEY )
         18 C, 09 C, ( DEY )
         18 C, ED C, 00 C, ( STD 0,Y )
         E6 C, 02 C, ( LDAB 2,X )
         18 C, 09 C, ( DEY )
         18 C, 09 C, ( DEY )
         18 C, ED C, 00 C, ( STD 0,Y )
         E6 C, 03 C, ( LDAB 3,X )
         18 C, 09 C, ( DEY )
         18 C, 09 C, ( DEY )
         18 C, ED C, 00 C, ( STD 0,Y )
         E6 C, 04 C, ( LDAB 4,X )
         18 C, 09 C, ( DEY )
         18 C, 09 C, ( DEY )
         18 C, ED C, 00 C, ( STD 0,Y )
         39 ( RTS )
END-CODE
```

One other requirement of the Assembler Extension Word Set is the **;CODE** word. This wc (in conjuction with a newly defined defining word) causes later defined words to use t machine code interpreter following the **;CODE** . As an example, the following program se ment simulates the set up for a two engine monitoring program that uses identical nar for both engine's parameters. A variable **CHANNEL** is used to control a new type variar created by the defining word **CHANNEL-VARIABLE** .

```
Ø CONSTANT PORT
1 CONSTANT STBD
VARIABLE CHANNEL ( VALUE SHOULD BE Ø OR 1 )

: MAKE-CHANNEL-VARIABLE <BUILDS Ø , Ø ,
;CODE
FC C, ' CHANNEL @ , ( LDD CHANNEL )
F3 C, ' CHANNEL @ , ( ADDD CHANNEL )
8F C, ( XGDX )
EC C, ØØ C, ( LDD Ø,X )
18 C, Ø9 C, ( DEY )
18 C, Ø9 C, ( DEY )
18 C, ED C, ( STD Ø,Y )
7E , FE4E , ( JMP VERSION 1.Ø NEXT )
END-CODE

MAKE-CHANNEL-VARIABLE ENGINE-SUPPLY-TEMP
MAKE-CHANNEL-VARIABLE ENGINE-RETURN-TEMP
MAKE-CHANNEL-VARIABLE ENGINE-SUPPY-FLOW
MAKE-CHANNEL-VARIABLE ENGINE-RETURN-FLOW
MAKE-CHANNEL-VARIABLE ENGINE-RPM

PORT CHANNEL ! ( NOW WORK ON PORT ENGINE )
 ( etc. ... )
 STBD CHANNEL ! ( NOW WORK ON STBD ENGINE )
 ( etc. ... )
```

1. CODE DEFINITIONS

Although the 8K versions of Max-FORTH do not have an included assembler, code definitions can still be added. Three key words, **CODE** , **END-CODE** and **CODE-SUB** have been added to facilitate this.

CODE is a defining word that creates a head for the name string that follows it. The Code Field of the created word points to its own Parameter Field. This causes control to transfer to the code in the Parameter Field when the word is executed. **CODE** does not set the compilation mode but does leave security values on the stack for **END-CODE** which ends the code definition.

Usually, in systems with assemblers, what goes between the **CODE** <name> and **END-CODE** , is assembly mnemonics and their parameters. Although there is an **ASSEMBLER** vocabulary, as required by the Assembler Extension Word Set, there was not enough room for a full assembler, so the vocabulary is empty. In order to enter machine code, the code must be hand assembled and placed in the dictionary using **,** and **C,** . At the end of the code definition's execution, control must be transfered back to the Max-FORTH system. This will usually be done by a JMP NEXT instruction. Not knowing the address of NEXT can be a big drawback to using machine code definitions.

For that very reason, the defining word, **CODE-SUB** , was added to Max-FORTH. It allows the entry of a machine coded subroutine to be entered as a FORTH definition. Like CODE , **CODE-SUB** creates a head for its name string, and leaves security for **END-CODE** . **CODE-SUB** , however, puts the address of an interpreter in the new definition's Code Field. The interpreter will JSR to the new definition's Parameter Field when it is called, and will JMP NEXT after the called routine returns. The normal way to finish a **CODE-SUB** definition, is by compiling an RTS and using the standard **END-CODE** definition termination. The **CODE-SUB** definition can either be executed by name from high level FORTH or called from a machine code routine by JSR'ing to its Parameter Field.

The following example illustrates the creation of a code definition that causes the processor to go into the wait mode. It takes advantage of the previously described **CODE-SUB** word.

```
CODE-SUB WAIT
  3E ( WAIT INSTRUCTION )
  39 ( RTS )
END-CODE
```

It should be noted that certain registers have significance to the virtual FORTH machine and require special treatment in code definitions. The Y register is used for the Data Stack Pointer. It should be left alone for the most part. (Note: the Data Stack grows down in memory.) If something is to be removed from the stack, it should be removed and then the Y register should be incremented twice (ie: LDD 0,Y INY INY). If something is to be put on the stack, the Y register should FIRST be decremented twice and then the value should be put on (ie: DEY DEY STD 0,Y). If this order is not observed, the that, the autostarted definition need not be an endless loop. It can terminate normally with a ; for high level words or a JMP NEXT for code definitions. When the system turns control over to a temporary autostart word, the IP is set to return control to the third autostart attempt. In this manner, normal termination of a temporary autostart word will cause initiation of the primary autostart search.

The third autostart attempt checks the beginning of EEPROM, $B600, and then searches all 1K memory map boundaries, starting at location $0400 and continuing through $FC00, for a primary autostart pattern, $A55A. If the primary autostart pattern is found at a given location $XX00, the next word location following the autostart pattern, $XX02, is taken to be the CFA of the word to be run. This word could be anywhere in the memory map, although this program will typically be in the memory device positioned at the boundary. The autostarted routine can either be a high level definition or a code definition. The autostarted definition will normally be an endless loop. If it terminates, with a ; for high level words or a JMP NEXT for code definitions, the system again returns control to the beginning of the third autostart attempt. In this case, the same word will be found and autostarted again and again. If no user autostarts are found, a primary autostart pattern should be found at $E000, the beginning of the Max-FORTH ROM. If by some circumstance it is not there, the search will begin again at the third step. It will so continue until an autostart pattern is found.

The three levels of autostart system give a great deal of flexibility to the system user. With no interference, the Max-FORTH operating system will find and run itself. A user program can intercept the primary autostart search by having an autostart pattern at a memory location lower than that of the Max-FORTH's ROM. A specific application to be started and the system thereby dedicated to its operation, instead of the running the Max-FORTH outer interpreter. A user may simply want to modify the normal start up procedure without taking permanent system initialization responsibilities. In this case, the temporary autostart option allows linking of additional ROM's and command vocabularies into Max-FORTH or even multitasking without interfering with the primary autostart abilities. The temporary autostart option is also useful to modify baud rates, etc., when there is no opportunity to catch the primary autostart by being placed in "lower" memory (ie: primary autostart pattern located at $B600 or $0400). A ROM higher in the memory map can still perform its function and allow the primary search to find its mark.

1. USING COMPILATION ADDRESSES

The headerless code feature of the Max-FORTH language is not available in many other versions of FORTH, so it is necessary to point out the significance of the PFAPTR and how it can affect the use of a word's compilation address in an application program.

The use of the Parameter Field Address is common place in Max-FORTH and other FORTH's as well. Max-FORTH's dictionary structure is slightly different because of its PFAPTR which allows headerless code generation and testing. Words such as ' do not return a PFA, but rather a PFAPTR. The word's PFAPTR is a part of its head. It serves as the connection between a word's head and its code. To convert a PFAPTR to the actual PFA of a word, only a @ is required. Some care must be used in a compiled program to be sure that the address of the PFAPTR which is in the head is available to be fetched. If the program is target compiled and the heads discarded, there will be a mysterious error haunting in the headerless program.

```
-----------------------------------------------------------------------
:length field : name field : link field : parameter-field-pointer field :
-----------------------------------------------------------------------
                         Word's Head Components

                     ------------------------------
                     : code field : parameter field :
                     ------------------------------
                         Word's Code Components
```

For example:

```
: INITI
  ' NEW-KEY-RTN @ ( note @ changes PFAPTR to PFA ) UKEY !
;
```

works fine if the program is not headerless, or if the heads are still in memory. But if the program is headerless, and the program is in ROM and the heads go away with a powerdown, the program will not work at power up.

It is suggested that the following format be used instead.

```
: INITI
  [ ' KEW-KEY-RTN @ ] LITERAL UKEY !
;
```

Note in this situation that the [causes the ' and @ to be executed in the immediate mode. The] returns to the compile mode and the **LITERAL** compiles a literal with the value from the stack left by the [ ] . All the indirection involving the head is done at compilation time, rather than at run time.

1. INTERRUPTS

The 68HC11 has many separate interrupt vectors. In order not to compromise their user availability (since the ROM inside the chip can not be user modifiable), these vectors were assigned with addresses outside the ROM. The addresses chosen for these vectors to point at were the high locations of the EEPROM from $B7xx to $B7xx. This collection of 3 byte locations will typically consist of a jump table. At each location, a user machine coded jump instruction will direct the interrupt on to the location of its own machine code handler. If EEPROM has been disabled, external memory can provide this second redirection.

A machine code handler must first be written to use a particular interrupt. It will normally end with an RTI to return to the interrupted program. The JMP op-code and the address of the code definition's parameter field should be installed in the jump table. The particular interrupt can then be enabled.

```
CODE IRQRTN ( HANDLE EXT PIN IRQ'S )
  ( ... )
  ( ... )
  ( ... )
  3B C, ( RTI )
END-CODE

CODE-SUB CLEAR-CC-MASKS
  86 C, ØØ C, ( LDAA # Ø )
  Ø6 C, ( TAP )
  39 C, ( RTS )
END-CODE

: ENABLE-IRQ'S
  7E B7E9 EEC! ( JMP OP-CODE )
  [ ' IRQRTN @ >< FF AND ] LITERAL B7EA EEC! ( HI BYTE )
  [ ' IRQRTN @    FF AND ] LITERAL B7EB EEC! ( LO BYTE )
  CLEAR-CC-MASKS
;
```

In this example, **ENABLE-IRQ's** puts the JMP op-code in the address ($B7E9) pointed to by the External IRQ Pin Vector ($FFF2). It fetches the address of the interrupt routine that will handle the IRQ Pin's requests, **IRQRTN** and puts it in after the JMP op-code. This is accomplished one byte at a time. Each time, the compilation mode is left. The interrupt routine's name is ticked. This returns its PFAPTR. The PFAPTR is converted to a PFA by the **@** . The appropriate high byte or low byte of the result is masked out. The compilation is then reentered. The value on the stack is compiled as a literal to be programmed into EEPROM at run time. A routine that clears the condition code register mask bits is then called , enabling interrupts. Be aware that the stop mode and XIRQ bits are also effected by that routine. High level interrupts can easily be handled in Max-FORTH. Low level interrupt handlers that call high level interrupts can be written with only 3 lines of machine code. This is possible because of the inclusion of a subroutine called **ATO4** which can start a high level word from machine code.

To run a high level word, all that is necessary is to first load the D Register with the Code Field Address of the word to be run, load the Y Register with a pointer to free memory area that can be used as data stack and to then jump subroutine to **ATO4** . This routine catches the system after the completion of the so called high level word and does an RTS to return to the machine code that called it. The high level word will probably use the Data and Return Stacks, but since Max-FORTH has been written to be totally preemptable, this should pose no problem. The SP and Y registers should not need adjustment. (In many instances, the Y Register would not need to be modified either. Only the **FIND** routine in Max-FORTH uses the Y register for anything except a very carefully maintained stack pointer. If the foreground program is not using **FIND** or in other words, the outer interpreter, but is instead a dedicated program itself, and the user has not entered machine coded definitions that tamper with Y's purpose as a data stack pointer, no concern need be paid to Y's contents during interrupt.)

There is a built-in constant that returns the address of this special subroutine. To aquire this address simply enter **ATO4** .

```
: HI-LEVEL-IRQRTN ( HANDLE EXT PIN IRQ'S )
  ( ... )
  ( ... )
  ( ... )
;

                                   cfa
CODE LO-LEVEL-IRQRTN
  CC C, ' HI-LEVEL-IRQRTN , ( LDD # CFA-OF-HI-LEVEL-IRQRTN )
  BD C, ATO4 , ( JSR ATO4 )
  3B C, ( RTI )
END-CODE

CODE-SUB CLEAR-CC-MASKS
  86 C, 00 C, ( LDAA # 0 )
  06 C, ( TAP )
  39 C, ( RTS )
END-CODE

: ENABLE-IRQ'S
  7E B7E9 EEC! ( JMP OP-CODE )
  [ ' IRQRTN @ >< FF AND ] LITERAL B7EA EEC! ( HI BYTE )
  [ ' IRQRTN @    FF AND ] LITERAL B7EB EEC! ( LO BYTE )
  CLEAR-CC-MASKS
;
```

1. PROGRAMMING THE INPUT / OUTPUT OF THE F68CH11

All of the F68HC11 onboard input/output features are controlled by a single register block of 64 locations. Understanding the use of those registers means understanding all of the I/O features of the F68HC11.

The location of this register block is controlled (you might have quessed) by a register. Upon reset, this register establishes the register block at $1000. Although the Max-FORTH operating system allows modification, the register set will normally be moved by the operating system to $B000 for Rev 2 parts ($9000 for Rev 0 and Rev 1) to give the most consistently contiguous memory map possible. (At $B000, the registers are in the same 4K block of memory already partitioned by the EEPROM.) The following discussion assumes the register set is at $B000.

The number of register may seem formidable. It will be easier to understand them by first grouping them by their function. There are five major groups: port, timer, serial channel, A/D and miscelaneous.

The 68HC11 includes a 40 I/O pins in five 8-bit ports. All of these pins serve multiple functions depending on the operating mode and the internal control registers. These registers are in memory from $B000 to $B00A.

PORTA	$B000
PIOC	$B002
PORTC	$B003
PORTB	$B004
PORTCL	$B005
DDRC	$B007
PORTD	$B008
DDRD	$B009
PORTE	$B00A

The first register, PORTA at $B000, is used to read and write Port A. When Port A is used for general purpose I/O bits, bits 0, 1 and 2 are configured for input-only and writes to these bits have no meaning or effect. Bits 3, 4, 5 and 6 of Port A are configured for output-only. Reads of these bits return the levels sensed at the inputs to the pin drivers. Port A bit 7 (PA7) can be configured as a general-purpose I/O using the DDRA7 bit in the PACTL register. Port A may also be configured as: three input capture functions (IC1, IC2, IC3), four output compare functions (OC2, OC3, OC4, OC5), with a pulse accumulator input (PA1) or a fifth output compare functions (OC1). Each port A bit that is not used for a capture or compare function may be used as a general purpose input or output line.

B7	B6	B5	B4	B3	B2	B1	B0	
PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	$B000 **PORTA**

The second location at SBØØ1 is reserved and has no function currently.

The Port I/O Control register, PIOC at location SBØØ2, has a collection of control bits dealing largely with handshake control. With the exception of bit 7, which is read only, the PIOC is a read/write register.

Due to the complexity of the bits in the PIOC each bit will be described separately.

B7	B6	B5	B4	B3	B2	B1	BØ	
STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB	SBØØ2 **PIOC**

Bit 7 **STAF** Strobe A Interrupt Status Flag. This bit is set when a selected edge of strobe A occurs. Clearing it depends on the state of HNDS and OIN bits. STAF is cleared by reset.

HNDS	**OIN**	**Clearing Mechanism**
Ø	X	Reading PIOC (with STAF Set) Followed by a Read of PORTCL
1	Ø	Reading PIOC (with STAF Set) Followed by a Read of PORTCL
1	1	Reading PIOC (with STAF Set) Followed by a Write to PORTCL

Bit 6 **STAI** Strobe A Interrupt Enable Mask. When this bit is set and the I bit in the condition code register is clear, STAF (when set) will request an interrupt. STAI is cleared by reset.

Bit 5 **CWOM** Port C Wire-OR Mode. When clear, port C operates normally. When set, port C behaves as open-drain outputs. CWOM is cleared by reset.

Bit 4 **HNDS** Handshake Mode. When clear, strobe A acts as a simple input strobe to latch data into PORTCL, and strobe B acts as a simple output strobe which pulses after a write to port B. When set, a handshake protocol involving port C, STRA and STRB is selected (see the definition of the OIN bit).

Bit 3 **OIN** Output or Input Handshaking. This bit has no meaning when HNDS=Ø. Otherwise, when OIN is clear, input handshake mode is selected. When OIN is set, output handshake mode is selected. OIN is cleared by reset.

Bit 2 **PLS** Pulse/Interlocked Handshake Operation. This bit has no meaning if HNDS=Ø. When PLS is clear, interlocked handshake operation is selected. In this mode, strobe B, once activated, stays active until the selected edge of strobe A is detected. When PLS is set, strobe B is pulsed for two E cycles. This bit is undefined coming out of reset.

Bit 1 **EGA** Active Edge for Strobe A. When clear, falling edge of STRA is selected. When output handshake is selected, port C bits obey the DDRC while STRA is low, but port C is forced to output when STRA is high.

When set, rising edge of STRA is selected. When output handshake is selected, port C bits obey the DDRC while STRA is high, but port C is forced to output when STRA is low. This bit is set by reset.

Bit 2 **INVB** Invert Strobe B. When clear, the active level on strobe B is a logic zero. When set, the active level on strobe B is a logic one. It is set by reset.

Location $B004, PORTC, is used to read and write the Port C pins. All Port C pins are general-purpose input/output pins. The direction of the Port C lines are controlled by a direction register, DDRC. Port C inputs can be latched by the STRA input. Port C may also be used in full handshake modes of parallel I/O where the STRA input and STRB output act as handshake control lines.

B7	B6	B5	B4	B3	B2	B1	B0
PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0

$B003 **PORTC**

Location $B005, PORTB, is used to read and write Port B. All of the port B pins are general-purpose output pins. During reads of this port, the level sensed at the input side of the port B output drivers is read. Port B may also be used in a simple strobed output mode where the STRB pulses each time port B is written.

B7	B6	B5	B4	B3	B2	B1	B0
PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0

$B004 **PORTB**

Location $B005, PORTCL, is used to read Port C, but it returns the value latched at the time of the last significant edge on STRA. All Port C pins are general-purpose input/output pins. The direction of the Port C lines are controlled by a direction register, DDRC. Port C inputs can be latched by the STRA input. Port C may also be used in full handshake modes of parallel I/O where the STRA input and STRB output act as handshake control lines.

B7	B6	B5	B4	B3	B2	B1	B0
PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0

$B005 **PORTCL**

The next location at $B006 is reserved and has no function currently.

A data direction register, DDRC at $B007, determines whether the individual Port C pins are input or outputs. The data direction register can be either read or written to in order to set the Port C I/O directions. Each bit in PORTC data register has a bit position in the DDRC register. When a bit is configured for output, by being set to 1, the value returned by a read is the value at the input to the pin driver. When a line is configured as an input, by clearing the DDRC bit, the pin becomes a high impedance input. If a write is executed to a line that is configured as an input, the value does not affect the I/O pin, but the bit is stored in an internal latch so that if the line is later reconfigured as an output, then this value appears at the I/O pin.

B7	B6	B5	B4	B3	B2	B1	BØ	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit Ø	$BØØ7 **DDRC**

Location $BØØ8, PORTD, is used to read and write Port D. Port D bits Ø-5 may be used for general I/O or with the serial communications interface (SCI) and serial peripheral interface (SPI) subsystems. Bits 6 and 7 are used as handshake control signals for ports B and C.

B7	B6	B5	B4	B3	B2	B1	BØ	
		PD5	PD4	PD3	PD2	PD1	PDØ	$BØØ8 **PORTD**

A data direction register, DDRD at $BØØ9, determines whether the five individual general purpose Port D pins act as input or outputs. The data direction register can be either read or written to in order to set the Port D I/O directions. Each bit in PORTD data register has a bit position in the DDRC register. When a bit is configured for output, by being set to 1, the value returned by a read is the value at the input to the pin driver. When a line is configured as an input, by clearing the DDRD bit, the pin becomes a high impedance input. If a write is executed to a line that is configured as an input, the value does not affect the I/O pin, but the bit is stored in an internal latch so that if the line is later reconfigured as an output, then this value appears at the I/O pin.

B7	B6	B5	B4	B3	B2	B1	BØ	
		Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit Ø	$BØØ9 **DDRD**

Note that bits 6 and 7 of Port D are dedicated to bus control (AS and R/W) while in expanded mode or parallel I/O strobes (STRA and STRB) while in single chip mode. For this reason, bits 6 and 7 of Port D are not available as general purpose I/O lines and the associated bits in the DDRD and PORTD registers are not implemented.

Location $BØØA, PORTE, is used to read Port E. In all operating modes, Port E is used for general-purpose inputs and/or analog-to-digital (A/D) channel inputs. Port E should not be read while an A/D conversion is actually taking place. Writes to the PORTE address have no meaning or effect.

```
   B7    B6    B5    B4    B3    B2    B1    BØ
|-----|-----|-----|-----|-----|-----|-----|-----|
| PE7 | PE6 | PE5 | PE4 | PE3 | PE2 | PE1 | PEØ |   $BØØE PORTE
|-----|-----|-----|-----|-----|-----|-----|-----|
```

The largest of the five groups of registers concern the timer functions:

CFORC	$BØØB	TOC4	$BØ1C
OC1M	$BØØC	TOC5	$BØ1E
OC1D	$BØØD	TCTL1	$BØ2Ø
TCNT	$BØØE	TCTL2	$BØ21
TIC1	$BØ1Ø	TMSK1	$BØ22
TIC2	$BØ12	TFLG1	$BØ23
TIC3	$BØ14	TMSK2	$BØ24
TOC1	$BØ16	TFLG2	$BØ25
TOC2	$BØ18	PACTL	$BØ26
TOC3	$BØ1A	PACNT	$BØ27

The timer has a single 16-bit free-running counter which is clocked by the output of a four-stage prescaler (divide by 1, 4, 8 or 16), which is in turn driven by the MCU E clock. Input functions are called input captures. These input captures record the count from the free-running counter in response to a detected edge on an input pin. Output functions, called output compares, can automatically affect output pins when there is a match between a 16-bit output-compare register and the free-running counter. Additionally, one output compare can affect any or all of the five output pins (Bits 7-3) in Port A at once, as a result of a successful compare. This function allows control of multiple I/O pins automatically with a single output compare.

This timer system has a total of three input capture register and five output compare registers.

The first register in the group deals with the output compare function. The write-only register, CFORC at $BØØB, allows output compares to be forced by writing to CFORC with the associated bit set for each output compare that is to be forced. The action taken as a result of a forced compare is the same as if there was a match between the OCx register and the free-running counter, except for the corresponding interrupt flag status bits which are not set. Reads of this location have no meaning or effect and always return logic zeros ($ØØ).

Five of the bit positions in the CFORC register, bits 7 - 3, correspond to the five output compares. Bits 2, 1 and Ø of the CFORC register are not implemented. Note that the compare force function is not generally recommended for use with the output toggle function because a normal compare occurring immediately before or after the force may result in the undesirable operation.

B7	B6	B5	B4	B3	B2	B1	B0	
FOC1	FOC2	FOC3	FOC4	FOC5	-	-	-	$B00B **CFORC**

Bits 7-3 **FOC1-FOC5** Force Output Compare x Action
0 - has no meaning
1 - causes the action which is programmed for output com pare except that the OCxF is not set.

Bits 2-0 Not implemented. Read as a logic zeros.

The next register, OC1M at $B00C, is used in conjunction with Output Compare 1 to specify the bits of port A which are to be affected as a result of a successful OC1 compare.

B7	B6	B5	B4	B3	B2	B1	B0	
OC1M7	OC1M6	OC1M5	OC1M4	OC1M3	-	-	-	$B00C **OC1M**

The bits of OC1M register correspond bit-for-bit with the output bits of Port A (bits 7 through 3 only). For each bit that is affected by the successful Output 1 Compare, the corresponding bit in OC1M should be set to one. Bits zero through two are not implemented and always read as zeros. This register is cleared by reset.

The register immediately following, OC1D at $B00D, is also used in conjunction with Output Compare 1. It specifies the data which is to be stored to the affected bits of Port A as the result of a successful OC1 compare. Bits zero through two are not implemented and always read as zeros. This register is not affected by reset.

B7	B6	B5	B4	B3	B2	B1	B0	
OC1D7	OCID6	OC1D5	OC1D4	OC1D3	-	-	-	$B00D **OC1D**

The bits of OC1D correspond bit-for-bit with the bits of port A (bits 7 through 3 only). When a successful OC1 compare occurs, for each bit that is set in OC1M, the corresponding data bit in OC1D is stored in the corresponding bit of port A.

If there is a conflicting situation where an OC1 compare and another output compare function occur during the same E cycle with both attempting to alter the same port A bit, the OC1 action overrides.

(Note that the pulse accumulator function shares bit 7 of port A. If the DDRA7 control bit in the PACTL register is set, then Port A bit 7 is configured as an output and OC1 can obtain access by setting OC1M bit 7. Further, if the PAEN control bit in the PACTL register is set, enabling the pulse accumulator, OC1 compares cause the pulse accumulator to take the appropriate action of the pulse counting or gating mode).

The next two locations, TCNT at $B00E and $B00F, can be used to read the counter at any time without affecting its value because it is clocked and read during opposite half cycles of the MPU E clock. A counter read should first address the most significant byte. An MPU read of this address causes the least significant byte to be transferred to a buffer. This buffer is not affected by reset and is accessed when reading the least significant byte of the counter. For double byte read insructions, the two accesses occur on consecutive bus cycles. The counter is cleared to $0000 during reset and is a read-only register in all but the test mode.

```
   B7    B6     B5     B4     B3     B2     B1     B0
|-----|-----|-----|-----|-----|-----|-----|-----|
|Bit15|  -  |  -  |  -  |  -  |  -  |  -  |Bit 8|   $B00E \
|-----|-----|-----|-----|-----|-----|-----|-----|          > TCNT
|Bit 7|  -  |  -  |  -  |  -  |  -  |  -  |Bit 0|   $B00F /
|-----|-----|-----|-----|-----|-----|-----|-----|
```

The counter is cleared to $0000 during reset and is a read-only register in all but the test mode.

The input capture registers are 16-bit read-only registers which are not affected by reset and are used to latch the value of the counter when a defined transition is sensed by the corresponding input capture edge detector. (The level transition which triggers counter transfer is defined by the corresponding input edge bits, EDGxB and EDGxA, in TCTL2.)

B7	B6	B5	B4	B3	B2	B1	BØ		
Bit15	-	-	-	-	-	-	Bit 8	$BØ1Ø \	> TIC1
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ11 /	

B7	B6	B5	B4	B3	B2	B1	BØ		
Bit15	-	-	-	-	-	-	Bit 8	$BØ12 \	> TIC2
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ13 /	

B7	B6	B5	B4	B3	B2	B1	BØ		
Bit15	-	-	-	-	-	-	Bit 8	$BØ14 \	> TIC3
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ15 /	

The result obtained by an input capture corresponds to the value of the counter one cycle after the transition which triggered the edge-detection logic. The selected edge transition sets the ICxF in TFLG1 and can cause an interrupt if the corresponding ICxI bit(s) is (are) set in the TMSK1 register. A read of the Input Capture Register's MSB inhibits captures for one E cycle to allow a double-byte read of the full 16-bit register.

The output compare registers are 16-bit read/write registers which are initialized to $FFFF by reset. They can be used as output waveform controls and as elapsed time indicators. If an output compare is not utilized, the unused registers may be used as storage locations.

All output compare registers have a separate dedicated comparator for comparing against the free-running counter. If a match is found, the corresponding output compare flag (OCxF) bit in TFLG1 is set and a specified action is automatically taken. For output compare functions two through five, the automatic action is controlled by pairs of bits in the TCTL1 control register (OMx and OLx). Each pair of control bits are encoded to specify the output action taken as a result of a successful OCx compare.

An interrupt can also accompany a successful output compare, provided that the corresponding interrupt enable bit (OCxl) is set in TMSK1.

B7	B6	B5	B4	B3	B2	B1	BØ		
Bit15	-	-	-	-	-	-	Bit 8	$BØ16 \	> **TIC1**
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ17 /	

B7	B6	B5	B4	B3	B2	B1	BØ		
Bit15	-	-	-	-	-	-	Bit 8	$BØ18 \	> **TIC2**
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ19 /	

B7	B6	B5	B4	B3	B2	B1	BØ		
Bit15	-	-	-	-	-	-	Bit 8	$BØ1A \	> **TIC3**
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ1B /	

B7	B6	B5	B4	B3	B2	B1	BØ		
Bit15	-	-	-	-	-	-	Bit 8	$BØ1C \	> **TIC4**
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ1D /	

B7	B6	B5	B4	B3	B2	B1	BØ		
Bit15	-	-	-	-	-	-	Bit 8	$BØ1E \	> **TIC5**
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ1F /	

After an MPU write cycle to the most significant byte, output compares are inhibited for one E cycle in order to allow writing of two consecutive bytes before making the next comparison. If both bytes of the register are to be changed, a double-byte write instruction should be used in order to take advantage of the compare inhibit feature. MPU writes can be made to either one of the bytes of the output compare register without affecting the other byte.

Timer Control Register 1, TCTL1, is an 8-bit read/write register. All bits in this register are cleared to zero during reset.

B7	B6	B5	B4	B3	B2	B1	BØ	
OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5	$BØ2Ø **TCTL1**

Bits 7, 5, 3, 1 - **OMx** Output Mode.

Bits 6, 4, 2, Ø - **OLx** Output Level.

These two control bits (OMx and OLx) are encoded to specify the output action to be taken as a result of a successful OCx compare.

OMx	**OLx**	**Action Taken Upon Successful Compare**
Ø	Ø	Timer disconnected from output pin logic
Ø	1	Toggle OCx output line
1	Ø	Clear OCx output line to zero
1	1	Set OCx output line to one

Timer Control Register 2, TCTL2, is an 8-bit read/write register except for bits 6 and 7 which are not implemented.

B7	B6	B5	B4	B3	B2	B1	BØ	
-	-	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A	$BØ21 **TCTL2**

Bits 7, 6 Not implemented. Read as logic zeros.

Bits 5, 3, 1 EDGxB Input Capture x Edge Control. These two bits (EDGxB and EDGxA) are cleared to zero by bits 4, 2, Ø EDGxA reset and are encoded to configure the input sensing logic for input capture x as follows:

EDGxB	EDGxA	Configuration
Ø	Ø	Capture disabled
Ø	1	Capture on rising edges only
1	Ø	Capture on falling edges only
1	1	Capture on any (rising or falling) edge

The timer interrupt mask register, TMSK1, is a read/write register.

B7	B6	B5	B4	B3	B2	B1	BØ	
OC1I	OC2I	OC3I	OC4I	OC5I	IC1I	IC2I	IC3I	$BØ22 **TMSK1**

Bits 7-3 **OCxI** Output Compare x Interrupt. If the OCxI mask bit is set when the OCxF flag bit is set, a hardware interrupt sequence is requested.

Bits 2-Ø **ICxI** Input Capture x Interrupt. If the ICxI mask bit is set when the ICxF flag bit is set, a hardware interrupt sequence is requested.

The Timer System Flag Register 1, TFLG1, indicates the occurence of timer system events and, together with the TMSK1 register, allows the timer sub-system to operate in a polled or interrupt driven system. For each bit in the timer flag register 1 (TFLG1), there is a corresponding bit in the timer mask register 1 (TMSK1) in the same bit position. If, and only if, the mask bit is set each time that the conditions for the corresponding flag are met, a hardware interrupt sequence, as well as the flag bit being set, is requested.

To clear a flag in this register, a pattern of ones should be written to the bits to be cleared. The zeros in that pattern will not affect the state of any bit. Bit manipulation instructions would be inappropriate for flag clearing because they are read-modify-write instructions. Even though the instruction mask implies that the programmer is only interested in some of the bits in the manipulated location, the entire 8-bit location is actually read and rewritten which may clear other bits in the register.

B7	B6	B5	B4	B3	B2	B1	BØ	
OC1F	OC2F	OC3F	OC4F	OC5F	IC1F	IC2F	IC3F	$BØ23 **TFLG1**

Bits 7-3 **OCxF** Output Compare x Flag. This flag bit is set each time the timer counter matches the output compare register x value. A write of zero does not affect the state of this bit. A write of a one causes this bit to be cleared.

Bits 2-Ø **ICxF** Input Capture x Flag. This flag bit is set each time a selected active edge is detected on the ICx input line. A write of a zero does not affect this bit. A write of a one causes this bit to be cleared.

The Timer System Mask Register 2, TMSK2, is used to control whether or not a hardware interrupt sequence is requested, as a result of a status bit being set in timer system flag register 2. In addition, two timer prescaler bits are included in this register. For each of the four most significant bits in timer flag register 2 (TFLG2), there is a corresponding bit in the timer mask register 2 (TMSK2) in the same bit position. All bits in the TMSK2 Register are cleared by reset.

B7	B6	B5	B4	B3	B2	B1	BØ	
TOI	RTII	PAOVI	RAII	-	-	PR1	PRØ	$BØ24 **TMSK2**

Bit 7 **TOI** Timer Overflow Interrupt Enable. This read/write bit is cleared by reset. If this bit is set, a timer overflow interrupt request is initiated each time the TOF bit (in TFLG2) is set, as a result of a timer counter overflow.

Bit 6 **RTI** Interrupt Enable. This read/write bit is used to enable or inhibit RTIF interrupt flags from causing hardware interrupt sequences.

It is cleared to zero by reset. When RTII is clear, the RTIF flag is masked (inhibited). When RTII is set, the RTIF flag is enabled to cause a hardware interrupt.

Bit 5 **PAOVI** Pulse Accumulator Overflow Interrupt Enable. This read/write bit is cleared to zero by reset and is used to enable or inhibit PAOVF interrupt flags from causing hardware interrupt sequences. When it is set, a hardware interrupt results when the PAOVF bit is set.

Bit 4 **PAII** Pulse Accumulator Input Interrupt Enable. This read/write bit is used to enable or inhibit PAIF interrupt flags from causing hardware interrupt sequences. It is cleared to zero by reset. When it is set, a hardware interrupt results if the PAIF bit is set.

Bit 3-2 These bits are not implemented. Reads of these bits will always return a logic zero.

Bit 1-Ø **PR1,PRØ** Timer Prescaler Selects. These two bits may be read at any time but may only be written during initialization. Reset clears these bits. Writes are disabled after the first write or after 64 E cycles out of reset. If the MCU is in special test or special bootstrap mode, then these two bits may be written any time.

These two bits specify the timer prescaler divide factor:

PR1	PRØ	Divide-by-Factor
Ø	Ø	1
Ø	1	4
1	Ø	8
1	1	16

Timer System Flag Register 2, TFLG2, indicates the occurence of timer system events and, together with the TMSK2 register, allows the timer sub-systems to operate in a polled or interrupt driven system. For each bit in timer flag register 2 (TFLG2), there is a corresponding bit in timer mask register 2 (TMSK2) in the same bit position. If the mask bit is set each time that the conditions for the corresponding flag are met, a hardware interrupt sequence, as well as the flag bit being set, is requested.

The timer system status register indicates when interrupt conditions have occurred. To clear a bit or bits, a logic one is written to it, or them.

Bit manipulation instructions are inappropriate for flag clearing because they are read-modify-write instructions. Even though the instruction mask implies that the programmer is only interested in some of the bits in the manipulated location, the entire 8-bit location is actually read and rewritten which may clear other bits in the register.

```
   B7     B6     B5     B4     B3     B2     B1     BØ
|-----|-----|-----|-----|-----|-----|-----|-----|
| TOF | RTIF|PACVF| PAIF|  -  |  -  |  -  |  -  |    $BØ25 TFLG2
|-----|-----|-----|-----|-----|-----|-----|-----|
```

Bit 7 **TOF** Timer Overflow. This bit is cleared by reset. It is set to one each time the 16-bit free-running counter advances from a value of $FFFF to $ØØØØ. In order to acknowledge (clear) the TOF flag the user must perform a write operation to TFLG2 with bit 7 set.

Bit 6 **RTIF** Real Time Interrupt Flag. This bit is cleared by reset. RTIF is set at each rising edge of the selected tap point. To clear the RTIF flag, a software write is performed to the TFLG2 register with bit 6 set to one.

Bit 5 **PAOVF** Pulse Accumulator Overflow Interrupt Flag. This bit is cleared by reset. PAOVF becomes set when the count in the pulse accumulator rolls over from $FF to $ØØ. This bit is cleared by a write to the TFLG2 register with bit 5 set.

Bit 4 **PAIF** Pulse Accumulator Input Edge Interrupt Flag. This bit is cleared by reset. PAIF becomes set when an active edge is detected on the PAI input pin. This bit is cleared by a write to the TFLG2 register with bit 4 set.

Bits 3-Ø These bits are not implemented and they read as logic zeros.

When the count changes from $FFFF to $ØØØØ, the timer overflow flag (TOF) bit is set in TFGL2. An interrupt can be enabled by setting the interrupt enable bit (TOI) in TMSK2.

The real time feature on the 68HC11 is configured and controlled by two bits in the PACTL control register (RTR1 and RTRØ) to select one of four interrupt rates. The RTII bit in TMSK2 enables the interrupt capability. Every time out causes the RTIF bit to be set in TFLG2 and if RTII is set, an interrupt request is generated. After reset, one entire real time interrupt period elapses before the RTIF flag is set for the first time.

The pulse accumulator is an 8-bit counter which can operate in either one of the two modes depending on the state of the PAMOD control bit in PACTL. In the event counting mode, the 8-bit counter is clocked to increasing values by an external pin. In the gated time accumulation mode, an E/64 clock drives the 8-bit counter, but only while the external PAI input pin is in a selected state.

The pulse accumulator uses port A bit 7 as its PAI input, but this pin also shares function as a general purpose I/O pin and as a timer output compare 1 output. Although port A bit 7 would normally be configured as an input when being used for the pulse accumulator, it still drives the pulse accumulator system even when it is configured for use in its alternate capacities

Four bits in this register are used to control an 8-bit pulse accumulator system and two other bits are used to select the rate for the real time interrupt system.

```
   B7     B6     B5     B4     B3     B2     B1     B0
|-----|-----|-----|-----|-----|-----|-----|-----|
|DDRA7| PAEN|PAMOD|PEDGE|  -  |  -  | RTR1| RTR0|   SB026 PATCL
|-----|-----|-----|-----|-----|-----|-----|-----|
```

Bit 7 **DDRA7** Data Direction for Port A Bit 7. This read/write bit is used to enable or disable the output driver for the Port A bit 7 pin. DDRA7 is cleared by reset. When DDRA7 is zero, the Port A bit 7 pin is configured for input only and when DDRA7 is one, Port A bit 7 is configured for output. Note that even when Port A bit 7 is configured for output, this pin still acts as the input to the pulse accumulator system.

Bit 6 **PAEN** Pulse Accumulator System Enable. This read/write bit is used to enable or disable the pulse accumulator system. PAEN is cleared by reset. When it is set, the pulse accumulator is enabled, and when clear, the pulse accumulator system is disabled.

Bit 5 ***PAMOD*** *Pulse Accumulator Mode.* This *read/write* bit controls whether the pulse accumulator is to operate in the external event counting mode or the gated time accumulation mode. When it is zero, the pulse accumulator counts pulses on the PAI input pin (port A bit 7). When it is set, the pulse accumulator operates in the gated time accumulation mode and the PAI input allows the pulse accumulator counter to count. The PAMOD bit is cleared to zero by reset.

PAMOD	PEDGE	Action on Clock
0	0	PAI Falling Edge Increments the Counter
0	1	PAI Rising Edge Increments the Counter
1	0	A '0' on PAI Inhibits Counting
1	1	A '1' on PAI Inhibits Counting

Bit 4 **PEDGE** Pulse Accumulator Edge Control. This read/write bit has different meanings depending on the state of the PAMOD control bit. This bit is cleared by reset.

Bit 3-2 These bits are not implemented and read as logic zeros.

Bit 1-0 **RTR1,RTR0** RTI Interrupt Rate Selects. These two read/write bits select one of four rates for the real time period interrupt circuit. Reset clears these two bits and after reset, a full RTI period elapses before the first RTI interrupt.

RTR1	RTR0	Divide E by	Xtal = 2^{23}	Xtal = 8.0 Mhz	Xtal = 4.0 Mhz
0	0	2^{13}	3.91 ms	4.10 ms	8.19 ms
0	1	2^{14}	7.81 ms	8.19 ms	16.38 ms
1	0	2^{15}	15.62 ms	16.38 ms	32.77 ms
1	1	2^{16}	31.25 ms	32.77 ms	65.54 ms
		E =	2.1 Mhz	2.0 Mhz	1.0 Mhz

SPCR	$B028
SPSR	$B029
SPDR	$B02A
BAUD	$B02B
SCCR1	$B02C
SCCR2	$B02D
SCSR	$B02E
SCDR	$B02F

The serial peripheral interface (SPI) is a synchronous interface built into the 68HC11 MCU which allows several 68HC11 MCUs or a 68HC11 plus peripheral devices, to be interconnected. In an SPI, separate wires (signals) are required for data and clock as the clock is not included in the data stream. An SPI system may be configured as a master or as a slave.

The four basic signals (MISO, MOSI, SCK and SS) used to transmit data by the serial peripheral interface are discussed in the following paragraphs. Each signal is described for both the master and slave modes.

Any SPI outputs has to have its corresponding data direction bit in DDRD set. If this bit is clear, the pin is disconnected from the SPI logic and becomes a general-purpose input.

There are three registers in the serial peripheral interface which provide control, status and data strorage functions.

B7	B6	B5	B4	B3	B2	B1	B0	
SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	$B028 **SPCR**

7 **SPIE** Serial Peripheral Interrupt Enable. When the SPIE bit is set, interrupts are allowed. Interrupts are masked when this bit is clear The SPIE bit is cleared by reset.

Bit 6 **SPE** Serial Peripheral Enable. When serial peripheral enable bit is set, it enables the SPI system by connecting it to the external pins. Because the SPE bit is cleared by reset, the SPI system is not connected to the external pins upon reset.

Bit 5 **DWOM** Port D ??? . If the DWOM bit is set, Port D output pins function as open-drain outputs, and when the DWOM is clear, Port D output pins function normally. The DWOM bit is cleared by reset.

Bit 4 **MSTR** Master. If the MSTR bit is set, the SPI is a master device. If cleared, the SPI is a slave device.

Bit 3 **CPOL** Clock Polarity. When the CPOL bit is cleared and data is not being transferred, a steady state low value is produced at the SCK pin of the master device. Conversely, if this bit is set, the SCK pin will idle high. This bit is also used in conjunction with the clock phase control bit to produce the desired clock-data relationship between master and slave. The CPOL bit is cleared by reset.

Bit 2 **CPHA** Clock Phase. The CPHA bit, in conjunction with the CPOL bit, controls the clock-data relationship between master and slave. In general, the CPHA bit selects which clock edge captures data and allows it to change states. The CPHA bit is set by reset.

Bit 1-Ø **SPR1 SPRØ** Serial Peripheral Rate Selects. These two bits select one of four baud rates to be used as a SCK if a device is a master; however, they have no effect in the slave mode. The SPR1 and SPRØ bits are not affected by reset.

SPR1	SPRØ	Internal Processor Clock Divide By
Ø	Ø	2
Ø	1	4
1	Ø	16
1	1	32

B7	B6	B5	B4	B3	B2	B1	BØ
SPIF	WCOL	-	MODF	-	-	-	-

$BØ29 **SPSR**

Bit 7 **SPIF** Serial Peripheral Data Transfer Flag. The SPIF bit is set upon completion of data transfer between the processor and external device. If SPIF goes high and if SPIF is set, a serial peripheral interrupt is generated. While SPIF is set, all writes to the serial peripheral data register are inhibited. Clearing the SPIF bit is accomplished by read-

ing the SPSR (with SPIF set) followed by an access
SPIF bit is cleared by reset.

Bit 6 **WCOL** Write Collision. The WCOL bit is set when an at write to the serial peripheral data register while da taking place. Clearing the WCOL bit is accomplished by r (with WCOL set) followed by an access to SPDR. The WCOL b. by reset.

Bit 5 Not implemented and reads as zero.

Bit 4 **MODF** Mode Fault Flag. The mode fault flag indicates that th have been a multi-master conflict for system control and allows a exit from system operation to a reset or default system state. The bit is normally clear, and is set only when the master device has its pin pulled low. Setting the MODF bit affects the internal seri peripheral interface system in the following ways:

1. An SPI interrupt is generated if SPIE = 1
2. The SPE bit is cleared. This disables the SPI.
3. The MSTR bit is cleared, thus forcing the device into the slave mode.

Clearing the MODF bit is accomplished by reading the SPSR (with MODF set), followed by awrite to the SPCR. Control bits SPE and MSTR maybe restored to their original set state after the MODF bit has been cleared. The MODF bit is cleared by reset.

Bit 3-Ø These bits are not implemented and read as zeros.

```
    7     6     5     4     3     2     1     Ø
|-----------------------------------------------|
|      SERIAL PERIPHERAL DATA I/O REGISTER      |    $BØ2A SPDR
|-----------------------------------------------|
```

The serial peripheral data I/O register is used to transmit and receive data on the serial bus. Only a write to this register will initiate transmission/reception of another byte and this will only occur in the master device. At the completion of transmitting a byte of data, the SPIF status bit is set in both the master and slave devices.

When the user reads the serial peripheral data I/O register, a buffer is actually being read. The first SPIF must be cleared by the time a second transfer of data from the shift register to the read buffer is initiated or an overrun condition will exist.

A write to the serial peripheral data I/O register is not buffered and places data directly into the shift register for transmission.

A full-duplex asynchronous Serial Communications Interface (SCI) is provided with a standard NRZ format (one start bit, eight or nine data bits and one stop bit) and a variety of baud rates. The SCI transmitter and receiver are functionally independent, but use the same data format and bit rate. "Baud" and "bit rate" are used synonymously in the following description.

Receive data (RxD) or transmit data (TxD) is the serial data which is transfered to the internal data bus from the input pin (RxD) and from the internal bus to the output pin (TxD). The user has option bits in serial communications control register 1 (SCCR1) to determine the "wake-up" method (WAKE bit) and data word length (M bit) of the SCI. Serial communications control register 2 (SCCR2) provides control bits which individually enable/disable the transmitter or receiver (TE and RE, respectively) enable system interrupts (TIE, TCIE, ILIE) and provide the wake-up enable bit (RWU) and the send break code bit (SBK). The baud rate register bits allow the user to select different baud rates which may be used as the rate control for the transmitter and receiver.

Data transmission is initiated by a write to serial communications data register (SCDR). Provided that the transmitter is enabled, data stored in the SCDR is transferred to the transmit serial shift register. This transfer of data sets the TDRE bit of the SCI status register (SCSR) and may generate an interrupt if the transmit interrupt is enabled. The transfer of data to the transmit shift register is synchronized with the bit rate clock. All data is transmitted bit zero first. Upon completion of data transmission, the TC (transmission complete) bit of the SCSR is set (provided no pending data, preamble or break is to be sent) and an interrupt may be generated if the transmit complete interrupt is enabled. If the transmitter is disabled and the data, preamble or break (in the transmit shift register) has been sent, the TC bit will also be set. This will also generate an interrupt if the TCIE bit is set.

When the SCDR is read, it contains the last data byte received, provided that the receiver is enabled. The RDRF bit of the SCSR is set to indicate that a data byte has been transfered from the input serial shift register to the SCDR, which can cause an interrupt if the receiver interrupt is enabled. The data transfer from the input serial shift register to the SCDR is synchronized by the receiver bit rate clock. The OR (overrun), NF (noise), or FE (framing error) bits of the SCSR may be set if data reception errors occured.

An idle line interrupt is generated if the idle line interrupt is enabled and the IDLE bit (which detects idle line transmission) of SCSR is set. This allows a receiver that is not in the wake-up mode to detect the end of a message, the preamble of a new message or to resynchronize with the transmitter.

The baud rate register, BAUD, provides the means for selecting different baud rates which may be used as the rate control for the transmitter and receiver. The SCP0-SCP1 bits function as a prescaler for the SCR0-SCR2 bits. Together, these five bits provide multiple baud rate combinations for a given crystal frequency.

B7	B6	B5	B4	B3	B2	B1	BØ
-	-	SCP1	SCPØ	-	SCR2	SCR1	SCRØ

SBØ2B **BAUD**

Bit 7-6 These two bits are not implemented.

Bit 5 **SCP1,SCPØ** Table below shows the prescale values attained from the E clock. Reset clears SCP1-SCPØ bits (divide-by-one).

SCP1	SCPØ	Internal Processor Clock Divide By
Ø	Ø	1
Ø	1	3
1	Ø	4
1	1	13

Bit 3 Not implemented.

Bit 2-Ø **SCR2 SCR1 SCRØ** These three bits select the baud rate of both the transmitter and the receiver. Table below shows the prescaler value that divides the output of the first stage. Reset does not affect the SCR2-SCRØ.

SCR2	SCR1	SCRØ	Prescaler Output Divide By
Ø	Ø	Ø	1
Ø	Ø	1	2
Ø	1	Ø	4
Ø	1	1	8
1	Ø	Ø	16
1	Ø	1	32
1	1	Ø	64
1	1	1	128

Note that there is a fixed rate divide-by-16 between the receive clock (Rx) and the transmit clock (Tx). The actual divider chain is controlled by the combined SCPØ-SCP1 and SCRØ-SCR2 bits in the baud rate register as illustrated.

SCP Bit	CLK* Dvd By	Crystal Frequency (MHz)				
0 1		8.3886	8.0	4.9152	4.0	3.6864
0 0	1	131.072 KBaud	125.000 KBaud	76.80 KBaud	62.50 KBaud	57.60 KBaud
0 1	3	43.690 KBaud	41.666 KBaud	25.60 KBaud	20.833 KBaud	19.20 KBaud
1 0	4	32.768 KBaud	31.250 KBaud	19.20 KBaud	15.625 KBaud	14.40 KBaud
1 1	13	10.082 KBaud	9600 Baud	5.907 KBaud	4800 Baud	4430 Baud

* The clock in the "Clock Divided By" column is in the internal processor clock.

Note: The divided frequencies shown above represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown on Table for some representative prescaler outputs.

SCP Bit	CLK* Divided	Crystal Frequency (MHz)				
2 1 0	By	131.072 KBaud	32.768 KBaud	76.80 KBaud	19.20 KBaud	9600 Baud
0 0 0	1	131.072 KBaud	32.768 KBaud	76.80 KBaud	19.20 KBaud	9600 Baud
0 0 1	2	65.536 KBaud	16.384 KBaud	38.40 KBaud	9600 Baud	4800 Baud
0 1 0	4	32.768 KBaud	8.192 KBaud	19.20 KBaud	4800 Baud	2400 Baud
0 1 1	8	16.384 KBaud	4.096 KBaud	9600 Baud	2400 Baud	1200 Baud
1 0 0	16	8.191 Kbaud	2.048 KBaud	4800 Baud	1200 Baud	600 Baud
1 0 1	32	4.096 KBaud	1.024 KBaud	2400 Baud	600 Baud	300 Baud
1 1 0	64	2.048 KBaud	512 Baud	1200 Baud	300 Baud	150 Baud
1 1 1	128	1.024 KBaud	256 Baud	900 Baud	150 Baud	75 Baud

Note: This illustrates how the SCI select bits can be used to provide lower transmitter baud rates by further dividing the prescaler output frequency. The five examples are only representative samples. In all cases, the baud rates shown are transmit baud rates (transmit clock) and the receiver clock is 16 times higher in frequency than the actual baud rate.

The Serial Communications Control Register 1, SCCR1, provides the control bits which:

1. determine the word length and
2. selects the method used for the wake-up feature

B7	B6	B5	B4	B3	B2	B1	B0	
R8	T8	-	M	WAKE	-	-	-	$B02C **SCCR1**

Bit 7 **R8** If the M bit is set, this bit provides a storage location for the ninth bit in the receive data word. Reset does not affect this bit.

Bit 6 **T8** If the M bit is set, this bit provides a storage location for the ninth bit in the transmit data word. Reset does not affect this bit.

Bit 5 This bit is not implemented and reads as zero.

Bit 4 **M** This bit selects the word length. Reset clears this bit.

0 = 1 start bit, 8 data bits, 1 stop bit
1 = 1 start bit, 9 data bits, 1 stop bit

Bit 3 **WAKE** This bit allows the user to select the method for receiver "wake-up".

When clear, an idle line condition (10 consecutive ones if M=0 or 11 consecutive ones if M=1) will wake-up the receiver.

When set, detection of a one in last data bit (eight data bit if M=0, ninth data bit if M=1) will wake-up the receiver.

Bit 2-0 These bits are not implemented and read as zeros.

The serial communications control register 2, SCCR2, provides control bits which individually enable/disable the SCI functions.

B7	B6	B5	B4	B3	B2	B1	B0	
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	$B02D **SCCR2**

Bit 7 **TIE** Transmit Interrupt Enable. When the TIE bit is set, the SCI interrupt occurs when TDRE is set. When TIE is clear, the TDRE interrupt is disabled. Cleared by reset.

Bit 6 **TCIE** Transmission Complete Interrupt. When the TCIE bit is set, the SCI interrupt occurs when TC is set. When TCIE is clear, the TC interrupt is disabled. Cleared by reset.

Bit 5 **RIE** Receive Interrupt Enable. When the RIE bit is set, the SCI interrupt occurs when OR or RDRF are set. When RIE is clear, the OR and RDRF interrupts are disabled. Cleared by reset.

Bit 4 **ILIE** Idle Line Interrupt Enable. When the ILIE bit is set, the SCI interrupt occurs when IDLE is set. When ILIE is clear, the IDLE interrupt is disabled. Cleared by reset.

Bit 3 **TE** Transmit Enable. When the TE bit is set, the transmit shift register output is applied to the TxD line. Depending on the state of control bit M (SCCR), a preamble of 10 (M=0) or 11 (M=1) consecutive ones is transmitted when software sets the TE bit from a cleared state. After loading the last byte in the serial communications data register and receiving the interrupt from TDRE, the user can clear TE. Transmission of the last byte will then be completed before the transmitter gives up control of the TxD pin. Cleared by reset.

Bit 2 **RE** Receive Enable. When the receive enable bit is set, the receiver is enabled. When RE is clear, the receiver is disabled and all of the status bits associated with the receiver (RDRF, IDLE, OR, NF and FE) are inhibited. Cleared by reset.

Bit 1 **RWU** Receiver Wake-up. When the RWU bit is set, it enables the "wake-up" function. If the WAKE bit is cleared, RWU is cleared after receiving 10 (M=0) or 11 (M=1) consecutive ones. If the WAKE bit is set, RWU is cleared after receiving a data word whose MSB is set. Cleared by reset.

Bit 0 **SBK** Send Break. If the SBK bit is toggled set and cleared, the transmitter sends 10 (M=0) or 11 (M=1) zeros and then reverts to idle or sending data. If SBK remains set, the transmitter will continually send whole blocks (sets of 10 or 11) zeros until cleared. At the completion of the break code, the transmitter sends at least one high bit to guarantee recognition of a valid start bit. Reset clears the SBK bit.

The Serial Communications Status Register, SCSR, provides inputs to the interrupt logic circuits for generation of the SCI system interrupt.

B7	B6	B5	B4	B3	B2	B1	B0	
TDRE	TC	RDRF	IDLE	OR	NF	FE	-	$B02E SCSR

Bit 7 **TDRE** Transmit Data Register Empty. The TDRE bit is set to indicate that the content of of the serial communications data register have been transferred to the transmit serial shift register. This bit is cleared by reading the SCSR (with TDRE=1) followed by a write to the SCDR. Reset sets the TDRE bit.

Bit 6 **TC** Transmit Complete. The TC bit is set at the end of a data frame, preamble, or break condition if:

1. TE= 1, TDRE = 1 and no pending data, preamble or break is to be transmitted; or

2. TE = Ø and the data, preamble, or break (in the transmit shift register) has been transmitted.

The TC bit is a status flag which indicates that one of the above conditions have occured. The TC bit is cleared by reading the SCSR (with TC set) followed by a write to the SCDR. Reset sets the TC bit.

Bit 5 **RDRF** Receiver Data Register Full. The RDRF bit is set when the receiver serial shift register is transferred to the SCDR. The RDRF bit is cleared when the SCSR is read (with RDRF set) followed by a read of the SCDR. Reset clears the RDRF bit.

Bit 4 **IDLE** Idle Line Detect. The IDLE bit, when set, indicates a receiver idle line is selected. The IDLE bit is cleared by reading the SCSR with IDLE set followed by a read of the SCDR. The IDLE bit is inhibited when the RWU bit is set. Reset clears the IDLE bit.

Bit 3 **OR** Overrun. The OR bit is set when the next byte is ready to be transferred from the receive shift register to the SCDR which is already full (RDRF bit is set). The only valid data is located in SCDR when OR is set. The OR bit is cleared when the SCSR is read (with OR set), followed by a read of the SCDR. Reset clears the OR bit.

Bit 2 **NF** Noise Flag. The NF bit is set if there is noise on any of the received bits, including the start and stop bits. The NF bit is not set until the RDRF flag is set. The NF bit is cleared when the SCSR is read (with NF set), followed by a read of the SCDR. Reset clears the NF bit.

Bit 1 **FE** Framing Error. The FE bit is set when no stop bit was detected in the data string received. The FE bit is set at the same time as the RDRF is set. If the byte received causes both framing and overrun errors, the processor will only recognize the overrun error. The framing error flag inhibits further transfer of data into SCDR until it is cleared. The FE bit is cleared when the SCSR is read (with FE=1) followed by a read of the SCDR. Reset clears the FE bit.

Bit Ø Not implemented. Reads as zero.

The Serial Communications Data Register, SCDR, performs two functions, i.e., it acts as the receive data register when it is read and as the transmit data register when it is written.

```
   B7     B6     B5     B4     B3     B2     B1    BØ
|------------------------------------------------------|
|        SERIAL COMMUNICATIONS DATA REGISTER           |    $BØ2F SCDR
|------------------------------------------------------|
```

ADCTL	$BØ3Ø
ADR1	$BØ31
ADR2	$BØ32

```
ADR3    $B033
ADR4    $B034
```

The 68HC11 includes an 8-channel multiplexed-input successive approximation analog-to-digital (A/D) converter with sample and hold to minimize conversion errors caused by rapidly changing input signals. Two dedicated pins (V_{RL}, V_{RH}) are provided for the reference supply voltage inputs. These pins may be connected to a lower noise power supply to assure full accuracy of the A/D conversion. The 8-bit A/D converter has a linearity error of 1/2 LSB and accepts analog inputs which range from V_{RL} to V_{RH}. Smaller analog input ranges can also be obtained by adjusting V_{RH} and V_{RL} to the desired upper and lower limits. Each conversion is accomplished in 32 MCU E clock cycles, provided the E rate is equal to or greater than 1 MHz. For systems which operate at clock rates less than 1.0 MHz, an internal R-C oscillator must be used to clock the A/D system by setting the CSEL bit in the OPTION register.

All bits in the ADCTL register may be read or written, except bit 7 which is a read-only status indicator and bit 6 which always reads as zero. Bit 7 is cleared at reset but the other bits are not affected by reset.

```
   B7    B6    B5    B4    B3    B2    B1    B0
|-----|-----|-----|-----|-----|-----|-----|-----|
| CCF |  -  | SCAN| MULT|  CD |  CC |  CB |  CA |     $B030 ADCTL
|-----|-----|-----|-----|-----|-----|-----|-----|
```

Bit 7 **CCF** Conversion Complete Flag. This read-only status indicator is set when all four A/D result registers contain valid conversion results. Each time the ADCTL register is written, this bit is automatically cleared to zero and a conversion sequence is started. In the continuous modes, conversions continue in a round-robin fashion and the result registers continue to be updated with current data even though the CCF bit remains set.

Note: The user must write to register ADCTL to initiate conversion.

Bit 6 Not implemented. Reads as zero.

Bit 5 **SCAN** Continuous Scan Control. When this control bit is cleared, the four requested conversions are performed once to fill the four result registers. When this control bit is set, conversions continue in a round-robin fashion with the result registers being updated as data becomes available.

Bit 4 **MULT** Multi-Channel/Single Channel Control. When this bit is cleared, the A/D system is configured to perform four consecutive conversions on the single channel specified by the four channel select bits CD through CA (bits 3-0 of ADCTL). When this bit is set, the A/D system is configured to perform a conversion on each of four channels where each result register corresponds to one channel.

Bit 3-0 **CD CC CB CA** Channel Select D, Channel Select C, Channel Select B, Channel Select A. Bits 3 through Ø are used to select one of 16 A/D channels. When a multiple channel mode is selected (MULT = 1), the two least-significant select bits (CB and CA) have no meaning and the CD and CC bits specify which group of four channels is to be converted. The signals selected by the four channel select control bits are shown in the table below.

CD	CC	CB	CA	Channel Signal	Result in ADRx if MULT=1
Ø	Ø	Ø	Ø	ANØ	ADR1
Ø	Ø	Ø	1	AN1	ADR2
Ø	Ø	1	Ø	AN2	ADR3
Ø	Ø	1	1	AN3	ADR4
Ø	1	Ø	Ø	AN4*	ADR1
Ø	1	Ø	1	AN5*	ADR2
Ø	1	1	Ø	AN6*	ADR3
Ø	1	1	1	AN7*	ADR4
1	Ø	Ø	Ø	RESERVED	ADR1
1	Ø	Ø	1	RESERVED	ADR2
1	Ø	1	Ø	RESERVED	ADR3
1	Ø	1	1	RESERVED	ADR4
1	1	Ø	Ø	V_{RH} Pin	ADR1
1	1	Ø	1	V_{RL} Pin	ADR2
1	1	1	Ø	$(V_{RH})/2$	ADR3
1	1	1	1	RESERVED	ADR4

There are two variations of single-channel operation. In the first variation (SCAN=Ø), the single-selected channel is converted four consecutive times with the first result being stored in the ADR4 register. After the fourth conversion is complete, all conversion activity is halted until a new conversion command is written to the ADCTL control register. In the second variation (SCAN=1), conversions continue to be performed on the selected channel with the fifth conversion being stored to the ADR1 register (overwriting the first conversion result), the sixth conversion overwrites ADR2, and so on continuously.

There are two variations in multiple-channel operation. In the first variation (SCAN=Ø), the selected group of four channels are converted, one each, with the first result being stored in the ADR1 result register and the fourth result being stored in the ADR4 register. After the fourth conversion is complete, all conversion activity is halted until a new conversion command is written to the ADCTL control register. In the second variation (SCAN=1), conversions continue to be performed on the selected group of channels with the fifth conversion being stored in the ADR1 register (replacing the earlier conversion result for the first channel in the group), the sixth conversion overwrites ADR2, and so on continuously.

The A/D result registers are read-only registers used to hold an 8-bit conversion result. Writes to these registers have no effect. Data in the A/D result registers is not valid unless the CCF flag bit in ADTCL is set, indicating conversion complete.

B7	B6	B5	B4	B3	B2	B1	BØ	
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ31 **ADR1**

B7	B6	B5	B4	B3	B2	B1	BØ	
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ32 **ADR2**

B7	B6	B5	B4	B3	B2	B1	BØ	
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ33 **ADR3**

B7	B6	B5	B4	B3	B2	B1	BØ	
Bit 7	-	-	-	-	-	-	Bit Ø	$BØ34 **ADR4**

The next four locations at $BØ35 through $BØ38 are reserved and have no function currently.

OPTION	$BØ39
COPRST	$BØ3A
PPROG	$BØ3B
HPRIO	$BØ3C
INIT	$BØ3D
TEST1	$BØ3E
CONFIG	$BØ3F

The Configuration Options Register (OPTION) is a special purpose 8-bit register that is used (optionally) during initialization to configure internal system configuration options. With the exception of bits 7, 6 and 3 (ADPU, CSEL and CME) which may be read or written at any time, this register may be written to only once after a reset and thereafter is a read-only register. If no write is performed to this location within 64 E-clock cycles after reset, then bits 5, 4, 1 and Ø (IRQ, DLY, CR1 and CRØ) will become read-only to minimize the possibility of any accidental changes to the system configuration. While in special test modes, the protection mechanism on this register is overridden and all bits in the OPTION register may be written.

B7	B6	B5	B4	B3	B2	B1	BØ
ADPU	CSEL	IRQE	DLY	CME	-	CR1	CRØ

$BØ39 **OPTION**

Bit 7 **ADPU** A/D Powerup. This bit is cleared by reset and controls operation of the on-chip analog-to-digital converter. When ADPU is clear, the A/D system is powered down and conversion requests will not return meaningful information. To use the A/D system, this bit should be set.

Bit 6 **CSEL** A/D Clock Source Select. This bit is cleared by reset and determines the clocking source for the on-chip A/D. When this bit is zero, the MCU E clock drives the A/D system. When CSEL is one, an on-chip R-C oscillator is enabled and clocks the A/D system at about 1.5 MHz rate. When running with an E clock less than 1 MHz, CSEL must be high to program or erase the EEPROM.

Bit 5 **IRQE** IRQ Edge/Level Sensitive. This read/write bit is cleared at reset. When it is clear, the IRQ pin is configured for level sensitive wired-OR operation (low level) and when it is set, the IRQ is configured for edge-only sensitivity (falling edges) on the IRQ pin.

Bit 4 **DLY** STOP Exit Turn-On Delay. This bit may only be written under special circumstances as described above. This bit is set to one during reset and controls whether or not a relatively long turn-on delay will be imposed before processing can resume after a STOP period. If an external clock source is supplied, this delay can be inhibited so that processing can resume within a few cycles of a wake up from STOP mode. When DLY is a one, delay is imposed to allow oscillator stabilization and when DLY is a zero, this delay is bypassed.

Bit 3 **CME** Clock Monitor Enable. This control bit may be read or written at any time. It controls whether or not the monitor circuit will trigger a reset sequence when a slow or absent system clock is detected. When it is clear, the clock monitor circuit is disabled. When it is set, the clock monitor circuit is enabled. Systems operating at or below 2ØØ KHz should not use the clock monitor function. Reset clears the CME bit.

Bit 2 Not implemented and reads as a logic zero.

Bit 1-Ø **CR1-CRØ** COP Timer Rate Selects.

C R 1	C R 0	$E/2^{15}$ Dvd'ed by	Xtal = 2^{23} Timeout -0/ +15.6 ms	Xtal = 8.0 Mhz Timeout -0/ +15.6 ms	Xtal = 4.0 Mhz Timeout -0/ +15.6 ms
0	0	1	15.625 ms	16.384 ms	32.768 ms
0	1	4	62.5 ms	65.536 ms	131.07 ms
1	0	8	250 ms	262.14 ms	524.29 ms
1	1	16	1 s	1.049 s	2.1 s
		E =	2.1 Mhz	2.0 Mhz	1.0 Mhz

The clock monitor function is enabled by the CME control bit in the OPTION register. When CME is zero, the function is disabled and when CME is one, the clock monitor function detects the absence of an E clock for more than a certain period of time. The timeout period is dependent on processing parameters and will be between 5 and 100 microseconds. This means that an E clock rate of 200 KHz or more will never cause a clock monitor failure and an E clock rate of 10 KHz or less will definitely cause a clock monitor failure. This implies that systems operating near or below an E clock rate of 200 KHz should not use the clock monitor function.

Upon detection of a slow or absent clock, the clock monitor circuit (if enabled by CME=1) will cause a system reset to be generated. This reset is issued to the external system via the bidirectional RESET pin.

Special considerations are needed when using a STOP function and clock monitor in the same system. Since the STOP function causes the clocks to be halted, the clock monitor function will generate a reset sequence if it is enabled at the time the STOP mode is entered.

The 68HC11 includes 512 bytes of EEPROM located in the area $B600 through $B7FF which has the same read cycle time as the internal ROM. PPROG Register (EEPROM Programming Control) register is used to control programming and erasure of the 512-byte internal EEPROM. Reset clears this register to $00 so EEPROM is configured so EEPROM is configured for normal reads.

B7	B6	B5	B4	B3	B2	B1	B0	
ODD	EVEN	-	BYTE	ROW	ERASE	EELAT	EEPGM	$B03B PPROG

Bit 7 **ODD** Odd Rows. Used to Program Odd Rows (TEST)

Bit 6 **EVEN** Even Rows. Used to Program Even Rows (TEST)

Bit 5 - Not Implemented

Bit 4 **BYTE** Byte Erase. Used for Erasing Bytes - Overrides Bit 3
Ø = Row or Bulk Erase
1 = Erase Only One Byte

Bit 3 **ROW** Row Erase. Used for Row Erasing
Ø = Bulk Erase
1 = Row Erase

Bit 2 **ERASE** Erase. Enables the Erase
Ø = Normal Read or Program
1 = Erase Mode

Bit 1 **EELAT** EEPROM Latch Control.
Ø=EEPROM Address and Data configured for Read
1=EEPROM Address and Data configured for Programming

Bit Ø **EEPGM** EEPROM Program Command.
Ø = Switched Off
1 = Turned On

The operating modes for the 512-byte EEPROM are as follows:

Normal Read - In this mode, the ERASE bit in the PPROG register must be clear (not in programming mode). While these two bits are cleared, the ROW and EEPGM bits in the OPTION register have no meaning or effect, and the 512-byte EEPROM may be read as if it were a normal ROM.

Programming - During EEPROM programming, the ROW bit is not used. If the E clock frequency is less than 1 MHz the CSEL bit in the OPTION register must be set.

Normal sequence of events in programming the EEPROM:

1) Write xxxx xØ1Ø to the PPROG register. This specifies program normal mode (ERASE bit=Ø), address/data buses configured to latch address and data information (EELAT bit=1), and erase voltage turned off (EEPGM bit=Ø).

2) Write data to be programmed to the desired EEPROM address. This write causes the address and data to be latched in a parallel internal latch.

3) Write EEPGM bit to one (xxxx xØ11). This couples the EEPROM programming supply voltage to the EEPROM array, to program the specified data into the specified data address in EEPROM.

4) Delay for 1Ø milliseconds.

5) Write xxxx xØ1Ø to the PPROG register to turn off the programming voltage.

6) Repeat steps 2 through 5 until all desired locations have been programmed.

7) Write EELAT bit back to zero to allow the programmed data to be verified.

Erase - If the E clock frequency is less than 1 MHz, the CSEL bit in the OPTION register must be set when erasing the EEPROM.

Three erase modes of EEPROM:

1) full 512-byte simultaneous "bulk" erase
2) "row" erase where only one row (16 bytes) is erased at a time and
3) "byte" erase where a single specified byte is erased.

NOTE

The erased state of all EEPROM cell is logic one. On early parts, byte and row erase are not implemented.

Normal procedures for erasure of the entire EEPROM:

1) Write xxxx Ø11Ø to the PPROG register. This specifies the "all" erase mode (ROW bit=Ø), erase mode (ERASE bit=1), EEPROM configured for address/data latching (EELAT bit=1), and erase voltage turned off (EEPGM bit=Ø).

 1a) A write must be done to any EEPROM address after Step 1.

 1b) Optionally, if the CONFIG register is also to be erased, a write to the address of the CONFIG register must be performed after "bulk" erase was specified by the write, in step 1 above, and before programming voltage is turned on in step 2 below.

In the case of erasure, the data involved in this write operation is unimportant and the write is needed only for the addressing information it provides.

2) -Write xxxx Ø111 to the PPROG register to turn on the the erase voltage to the EEPROM array.

3) Wait for 1Ø milliseconds to allow the erasure to complete.

4) Write xxxx Ø11Ø to the PPROG register to turn off the erase voltage.

5) Write xxxx ØØØØ to the PPROG register to return the EEPROM to the normal read configuration.

Normal procedure for erasure of a row of EEPROM:

1) Write xxxx 111Ø to the PPROG register. This specifies the "row" erase mode (ROW bit=1), erase mode (ERASE bit=1), address/data buses configured to latch row address information (EELAT bit =1), and erase voltage turned of (EEPGM=Ø).

2) Write to an address in the EEPROM row to be erased (each row is 16 bytes). This latches the row addressing information for the row to be erased.

3) Write xxxx 1111 to the PPROG register to turn on the erase voltage to the EEPROM array.

4) Wait for 1Ø milliseconds to allow the erasure to complete.

5) Write xxxx 111Ø to the PPROG register to turn off the erase voltage.

6) Write xxxx ØØØØ to the PPROG register to return the EEPROM to the normal read configuration.

Normal procedure for erasure of a single byte of EEPROM:

1) Write xxx1 x11Ø to the PPROG register. This specifies the byte erase mode (BYTE=1; ROW=x), erase mode (ERASE bit=1), address/data buses configured to latch address information (EELAT bit=1), and erase voltage turned off (EEPGM bit=Ø).

2) Write to the address in the EEPROM to be erased (data is ignored). This latches the address of the byte to be erased.

3) Write xxx1 x111 to the PPROG register. This turns on the erase voltage to the EEPROM array. EEPGM was not changed to one in the same write operation as the write that configured ROW, ERASE, and EELAT because of the possibility of enabling the erase voltage before the erase mode specification was stable.

4) Wait for 1Ø milliseconds to allow the erasure to complete.

5) Write xxx1 x11Ø to the PPROG register to turn off the erasure voltage.

6) Write xxx ØØØØ to the PPROG register to return the EEPROM to the normal read configuration.

The COPRST Register is used to reset the watch dog timer. The sequence required to accomplish this is:

1) write $55 to the CORPST register at $BØ3A, followed by
2) write $AA to the same address.

Both writes must occur in correct order prior to timeout but, any number of instructions may be executed between the writes. The elapsed time between adjacent software reset sequences must never be greater than the COP timeout period.

B7	B6	B5	B4	B3	B2	B1	BØ	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit Ø	$BØ3A **COPRST**

The HPRIO register is used to select one of the I bit related interrupt sources to be elevated to the highest I bit masked position in the priority resolution circuit. In addition, four miscellaneous system control bits are included in this register.

B7	B6	B5	B4	B3	B2	B1	BØ	
RBOOT	SMOD	MDA	IRV	PSEL3	PSEL2	PSEL1	PSELØ	$BØ3C **HPRIO**

Bit 7 **RBOOT** Read Bootsrap ROM. The read bootstrap ROM bit only has meaning when the SMOD bit is a logic one (special bootstrap mode or special test mode). At all times, this bit reverts to its logic zero disabled state and may not be written.

When set, upon reset in bootstrap mode only, the small bootstrap loader program is enabled. When clear, by reset in the other three modes, this ROM is disabled and accesses to this area are treated as external accesses.

Bit 6 **SMOD** Special Mode. The special mode write-only bit reflects the status of the MODB input pin at the rising edge of reset. It is set if the MODB pin is at or above 1.8 times V_{DD}, volts during reset. Otherwise, it is cleared or under software control from the special modes.

Bit 5 **MDA** Mode Select A. The mode select A bit reflects the status of the MODA input pin at the rising edge of reset. While the SMOD bit is a logic one (special test or special bootstrap mode in effect), the MDA bit may be written, thus, changing the operating mode of the MCU. When the SMOD bit is a logic zero, the MODA bit is a read-only bit and the operating mode cannot be changed without going through a reset sequence.

The table below summarizes the relationship between the SMOD and MDA bits and the MODB and MODA input pins at the rising edge of reset.

Inputs		Mode Description	Latched at Reset	
MODA	MODB		SMOD	MDA
Ø	1	Single Chip (Mode Ø)	Ø	Ø
1	1	Expanded Multiplexed (Mode 1)	Ø	1
Ø	*	Special Bootstrap	1	Ø
1	*	Special Test	1	1

1 = logic high
Ø = logic low * = 1.8
* = 1.8 times V_{DD}(or higher)

Bit 4 **IRV** Internal Read Visibility. The internal read visibility bit is used in the special modes (SMOD=1) to affect visibility of internal reads on the expansion data bus. IRV is writable only if SMOD=1 and returns to zero if SMOD=Ø. If the bit is zero, visibility of internal reads are blocked. If the bit is one, internal reads are visible on the external bus.

Bit 3-Ø **PSEL3, PSEL2, PSEL1, PSELØ** Priority Selects 3-Ø. These four select bits are used to specify one I bit related interrupt source which becomes the highest priority I bit related source.

PSEL3	PSEL2	PSEL1	PSEL0	Interrupt Source Promoted
0	0	0	0	Timer Overflow
0	0	0	1	Pulse Accumulator Overflow
0	0	1	0	Pulse Accumulator Input Edge
0	0	1	1	SPI Serial Transfer Complete
0	1	0	0	SCI Serial System
0	1	0	1	Reserved (Default to IRQ)
0	1	1	0	IRQ (External Pin or Parallel I/O)
0	1	1	1	Real Time Interrupt
1	0	0	0	Timer Input Capture 1
1	0	0	1	Timer Input Capture 2
1	0	1	0	Timer Input Capture 3
1	0	1	1	Timer Output Compare 1
1	1	0	0	Timer Output Compare 2
1	1	0	1	Timer Output Compare 3
1	1	1	0	Timer Output Compare 4
1	1	1	1	Timer Output Compare 5

Note: During reset, PSEL3, PSEL2, PSEL1 and PSEL0 are initialized to 0:1:0:1 which corresponds to "Reserved (default to IRQ)" being the highest priority I bit related interrupt source.

Interrupts in the 68HC11 obey a fixed hardware priority circuit to resolve simultaneous requests; however, one I bit related interrupt source may be elevated to the highest I bit priority position in the resolution circuit. The first six interrupt sources are not masked by the I bit in the condition code register and have the fixed priority interrupt relationship of: reset, clock monitor fail, COP fail, illegal opcode and XIRQ. (SWI is actually an instruction and has highest priority other than reset in the sense that once the SWI opcode is fetched no other interrupt can be honored until the SWI vector has been fetched). Each of these sources is an input to the priority resolution circuit. The highest I bit masked priority input to the resolution circuit is assigned under software control (of the HPRIO register) to be connected to any one of the remaining I bit related interrupt sources. In order to avoid timing races, the HPRIO register may only be written while the I bit related interrupts are inhibited (I bit in condition code register is a logic one). An interrupt that is assigned to this high priority position is still subject to masking by any associated control bits or the I bit in the condition code register. The interrupt vector address is not affected by assigning a source to this higher priority position.

The INIT Register is special purpose 8-bit register is used (optionally) during initialization to change the default locations of RAM and internal registers in the MCU memory map. It may be written to only once within the initial 64 E cycles after a reset and thereafter becomes a read-only register.

B7	B6	B5	B4	B3	B2	B1	B0		
RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0	$B03D	**INIT**

The default starting address for internal RAM is $0000 and the default starting address of the 64 byte internal register space is $1000 (the INIT register is initialized to $01 by reset). The upper four bits of the INIT register specify the starting address for the internal 256 byte RAM and the lower four bits of INIT specify the starting address for the 64 byte internal register space. The four bits reflect the upper nibble of the 16-bit address. The Max-FORTH operating system moves the registers to $B000 at reset for all versions with Revision 2 ($9000 for Rev 0 - 1).

Since the TEST1 Register at $B03E is only available in the test mode, it is not included in this section.

The 68HC11 allows an end user to configure the MCU system to his specific requirements through the use of hardwired options such as the mode select pins, semi-permanent EEPROM control bit specification. The CONFIG control register is implemented in EEPROM cells and controls the presence of ROM and EEPROM in the memory map, as well as the COPON COP watchdog system enable. An optional security feature is available to allow user protection of data in 68HC11 EEPROM and RAM.

B7	B6	B5	B4	B3	B2	B1	B0		
-	-	-	-	NOSEC	NOCOP	ROMON	EEON	$B03F	**CONFIG**

Bit 7-4 Not implemented. Read as logic zero.

Bit 3 **NOSEC** Security Mode Option. When the security mask option is specified, this bit can be used to enable a software antitheft mechanism. When cleared, this bit forces the MDA mode control bit to zero so that only single-chip modes of operation can be selected. If the bit is cleared during MCU reset in the special bootstrap mode, EEPROM and RAM are erased before the boot loading process continues.

Bit 2 **NOCOP** COP System OFF. When this bit is clear, the COP watchdog forced-reset function is enabled. When this bit is set, the COP watchdog circuit is disabled.

Bit 1 **ROMON** Enable On-Chip ROM Select. When this bit is clear, the 8K internal ROM is disabled and that memory space becomes an externally accessed space.

Bit 0 **EEON** Enable On-Chip EEPROM Select. When this bit is clear, the 512-

byte internal EEROM is disabled and that memory space becomes an externally accessed space.

Since the CONFIG register is implemented with EEPROM cells, special provisions must be made to erase and program this register. The normal EEPROM control bits in the PPROG register are used for this purpose.

1. MOVING THE USER AREA

The USER area normally resides in memory from $ØØØ6 to $ØØ79. It contains 58 word locations that hold the variables describing how Max-FORTH uses the rest of memory and how it interfaces to the outside world. When a cold reset occurs, the operating system initializes the first 41 words with values needed to start with a cleanly linked system.

There are several reasons why it could be advantageous to move the USER area. One might to be to give the data stack more room in low memory. (Ofcourse the dictionary would also have to be completely moved out in this case, too.) Another might be to put the USER area into battery backed RAM, so the operator could modify one location (the UP) to recover the complete dictionary and configuration. As an alternative, the user could create several USER areas, and thereby have several dictionaries or configurations in memory at once, switching back and forth by manipulating a single variable.

The following example creates a second user area at SCØØØ. The easiest way to create a new USER area is to copy the reset initialized RAM area to the new location. The new copy can then be edited to remove all references to its origin. This is done in this example by adding an offset to the existing values in the USER area to accomodate its new location. After the new area is initialized, placing its address in the user area pointer (UP) completes the transfer. Notice in this example the new user has different stack initializtion values as well. Execution of **ABORT** installs these new stack locations, and breaks the last ties with the original configuration.

```
COLD

HEX
ØØØØ CØØØ 1ØØ       ( Move copy of system RAM to new area )
CMOVE
CØØØ CØØE +!        ( Alter RØ for return stack init      )
CØØØ CØ1Ø +!        ( Alter SØ for data stack init        )
CØØØ CØ1C +!        ( Alter the location of the TIB       )
CØØØ CØ22 +!        ( Alter the location of the PAD       )
CØØØ CØ2C +!        ( Alter the dictionary pointer        )
CØØØ CØ34 +!        ( Alter the vocabulary link           )
CØØØ CØ38 +!        ( Alter the FORTH pointer             )
CØØØ CØ4Ø +!        ( Alter the EDITOR vocabulary linkage )
CØØØ CØ46 +!        ( Alter the ASSEMBLER "    "    "     )
CØØØ CØ4A +!        ( Alter the FENCE limit               )
CØØØ CØ6A +!        ( Alter CURRENT                       )
CØØØ CØ6C +!        ( Alter CONTEXT                       )

CØØ6 Ø4   !         ( Alter the UP to point to new area   )
ABORT               ( Install RØ and SØ values            )
```

MULTITASKING

Although Max-FORTH does not directly support multitasking, multitasking is possible. The user may add on a multitasking program with relative ease since Max-FORTH was written to be totally preemptable. When Max-FORTH was created, user variable were reserved to specify each task's priority, record the task's state and allow for a double linked list of tasks to be created.

Using these variables and interrupts, the user can create round robin or priority multitaskers, or combinations thereof. Possibly the simplest structured multitasker is the round robin time sliced scheme. The following example demonstrates such a system. For simplicity's sake, only one task is open to communicate with the terminal. (The other tasks' USER variables, KEY-BC-PTR and EMIT-BC-PTR, could have been modified to allow them to talk to their own terminal.)

(In order to find the value of NEXT2 in your revision of Max-FORTH, tick the address of **COLD** and **DUMP** the following 100 hex bytes. The address of NEXT2 is held in the two bytes immediately following the first 7E op-code in the that listing, about 90 hex bytes down.)

```
COLD

HEX
FORGET TASK

( MAKE NEW     )
( USER AREA    )
( FOR TASK #1 )
0000 C000 100
CMOVE
C000 C00E +!
C00E C010 +!
C100 C01C !
50 C01E !
C160 C022 !
C180 C02C !
C000 C034 +!
C000 C040 +!
C000 C046 +!
C000 C06A +!
C000 C06C +!
D006 C006 !
C806 C008 !

( MAKE NEW     )
( USER AREA    )
( FOR TASK #2 )
0000 C800 100
```

```
CMOVE
C800 C80E +!
C800 C810 +!
C900 C81C !
50 C81E !
C960 C822 !
C980 C82C !
C800 C834 +!
C800 C840 +!
C800 C846 +!
C800 C86A +!
C800 C86C +!
C006 C806 !
D006 C808 !

( MAKE NEW     )
( USER AREA    )
( FOR TASK #3 )
0000 D000 100
CMOVE
D000 D00E +!
D000 D010 +!
D100 D01C !
50 D01E !
D160 D022 !
D180 D02C !
D000 D034 +!
D000 D040 +!
D000 D046 +!
D000 D06A +!
D000 D06C +!
C806 D006 !
C006 D008 !

C006 04  !
ABORT
( Assumes three tasks and 8K of RAM at $C000 )
( The first task at $C000 slowly counts up on PB )
( The second task at $C800 acts as a thermostat: reads AN0 output PA6 )
( The third task at $D000 runs the normal Max-FORTH outer interpreter )
(   the task will also be master task, starting & handling of others  )

( Begin with simple task first )
: TASK ;
: WAIT 1000 0 DO LOOP ;
: RUN-TASK-1
  0
  BEGIN
    1+
    DUP
    B004 C!
    DUP
  C00A ! ( THIS LINE IS ONLY TO LEAVE A TRACE IN MEM )
  WAIT
 AGAIN
;

' RUN-TASK-1 CFA C002 ! ( SAVE CFA TO RUN IN AUTOSTART POS FOR LATER )


( Switch to define the second task )
C806 04  !
ABORT
: TASK ;

VARIABLE LO-SET-POINT
VARIABLE HI-SET-POINT
VARIABLE HEATER-STATE
VARIABLE TEMP

: READ-A/D
 0 B030 C! ( Start conversion for PE0 )
 BEGIN B030 C@ 80 AND UNTIL ( Wait for conversion complete )
 B031 C@ TEMP !
;

: CHK-LO?
 TEMP C@ LO-SET-POINT C@ <
;

: HEATER-ON
 1 HEATER-STATE !
 FF B000 !
;

: CHK-HI?
 TEMP C@ HI-SET-POINT C@ >
;

: HEATER-OFF
 0 HEATER-STATE !
 BF B000 !
;

: RUN-TASK-2
 70 LO-SET-POINT !
 74 HI-SET-POINT !
 BEGIN
  READ-A/D
  HEATER-STATE @ C80A ! ( THIS LINE IS ONLY TO LEAVE A TRACE IN MEM )
  CHK-LO?
  IF
   HEATER-ON
  THEN
```

```
    CHK-HI?
    IF
      HEATER-OFF
    THEN
  AGAIN
;

' RUN-TASK-2 CFA C802 ! ( SAVE CFA TO RUN IN AUTOSTART POS FOR LATER )

( Switch to define the third task )
D006 04  !
ABORT
: TASK ;

CODE IRQRTN
  86 C, 40 C,    ( LDAA # $40       CLEAR THE INTERRUPT SOURCE )
  B7 C, B025 ,   ( STAA $B025       BY KNOCKING DOWN BIT TFLG2 )
  DE C, 02 C,    ( LDX IP           SAVE IP ON CONTEXT'S STACK )
  3C C,          ( PSHX                                        )
  DE C, 00 C,    ( LDX W            SAVE W  ON CONTEXT'S STACK )
  3C C,          ( PSHX                                        )
  DE C, 04 C,    ( LDX UP           GET THIS TASK PTR AT $0004 )
  AF C, 06 C,    ( STS RPSAVE,X     SAVE OLD CONTEXT )
  EE C, 02 C,    ( LDX UPLINK,X     FIND NEXT TASK )
  DF C, 04 C,    ( STX UP           AND SAVE FOR NXT IRQ )
  AE C, 06 C,    ( LDS RPSAVE,X     SWITCH CONTEXT )
  38 C,          ( PULX                                        )
  DF C, 00 C,    ( STX W            RECOVER W  CONTEXT'S STACK )
  38 C,          ( PULX                                        )
  DF C, 02 C,    ( STX IP           RECOVER IP CONTEXT'S STACK )
  3B C,          ( RTI )
END-CODE

CODE-SUB CLEAR-CC-MASKS
  86 C, 00 C, ( LDAA # 0 )
  06 C, ( TAP )
  39 C, ( RTS )
END-CODE

: ENABLE-MULTITASKING
  ( INSTALL THE REAL-TIME-INTERRUPT JUMP TABLE ENTRY )
  7E B7E6 EEC! ( JMP OP-CODE )
  [ ' IRQRTN @ >< FF AND ] LITERAL B7E7 EEC! ( HI BYTE )
  [ ' IRQRTN @    FF AND ] LITERAL B7E8 EEC! ( LO BYTE )

  ( CREATE CONTEXTS FOR THE OTHER TASKS ON THEIR STACKS )
  ( TASK 1 START UP CONTEXT )
  FE54   C0FE  ! ( PROGRAM COUNTER = NEXT2 )
  C010 @ C0FC  ! ( Y REGISTER = XXXX )
  C002 @ C0FA  ! ( X REGISTER = CFA TO RUN )
  0000   C0F8  ! ( B&A REGISTERS )
  00     C0F7 C! ( CC REGISTER )
  0000   C0F5  ! ( IP )
  0000   C0F3  ! ( W )
  C0F2   C00C  ! ( RPSAVE )
  ( TASK 2 START UP CONTEXT )
  FE54   C8FE  ! ( PROGRAM COUNTER = NEXT2 )
  C810 @ C8FC  ! ( Y REGISTER = XXXX )
  C802 @ C8FA  ! ( X REGISTER = CFA TO RUN )
  0000   C8F8  ! ( B&A REGISTERS )
  00     C8F7 C! ( CC REGISTER )
  0000   C8F5  ! ( IP )
  0000   C8F3  ! ( W )
  C8F2   C80C  ! ( RPSAVE )

  ( ENABLE THE HARDWARE TO GIVE REAL TIME IRQ'S )
  40 B024 C! ( RTII SET )
  CLEAR-CC-MASKS
;

: TASK1? BEGIN C00A @ U. ?TERMINAL UNTIL ;
: TASK2? BEGIN C80A @ U. ?TERMINAL UNTIL ;
```

PREPARING A PROGRAM FOR PROM

The defining word VARIABLE works fine if your application is run in RAM, but when the program is moved to ROM the variable space allotted by VARIABLE becomes "frozen" in read only memory. The value of the variables can never be altered and the program will fail. An alternative to using VARIABLE is shown below.

```
HEX
VARIABLE RAM_POINTER      ( DEFINE A PIECE OF RAM TO HOLD          )
                          ( A POINTER TO NEXT FREE BYTE OF RAM     )
HWORD                     ( MOVE IT TO HEADS IF SEPERATE FROM CODE)
: IS CONSTANT ;           ( RENAME CONSTANT FOR EASY READING       )
HWORD                     ( MOVE IT TO HEADS IF SEPERATE FROM CODE)

200 RAM_POINTER !         ( IN A BUFFER STARTING AT 200            )

: RAM ( n --- addr )
  RAM_POINTER @           ( GET THE FREE RAM POINTER               )
  SWAP                    ( BRING THE # OF BYTES TO ALLOT UP       )
  RAM_POINTER +!          ( ADD THAT # TO POINTER IE: ALLOT RAM    )
;                         ( RETURN ADDRESS OF ALLOTTED RAM         )
```

Examples of how to use RAM:

```
2 RAM IS VARIABLE1
2 RAM IS VARIABLE2
4 RAM IS DVARIABLE
0 RAM IS SHOW-TABLE
2 RAM IS PARAMETER1
2 RAM IS PARAMETER2
2 RAM IS PARAMETER3
2 RAM IS PARAMETER4
0 RAM IS SHOW-END

VARIABLE1 @ VARIABLE2 @ + PARAMETER1 !
DVARIABLE @ DVARIABLE 2+ @ VARIABLE1 @ U/
PARAMETER3 ! PARAMETER 4 !

: INITI SHOW-TABLE SHOW-END OVER - ERASE ; ( CLEARS PARAMETER1-4)
```

ERROR MESSAGES

Standard Error Message

? (question mark) is the standard error message in Max-FORTH. An error exists when Max-FORTH responds with a ? prefixed with one of the following:

- the most recently entered word which is not part of the Max-FORTH dictionary

or

- the most recently entered number which is not valid under the current BASE.

Example: Enter WRONG . WRONG is not a part of the Max-FORTH dictionary, therefore, Max-FORTH will respond with WRONG ? .

Enter HEX . Max-FORTH will respond with OK . TYPE 10H . Max-FORTH will respond with 10H ? since 10H is not a valid hexadecimal number.

STANDARD ERROR MESSAGE ROUTINE

Max-FORTH has a standard routine for handling errors depending on the value of the user variable WARNING which is not named in the dictionary:

WARNING value	Max-FORTH action
less than 0	executes the word ABORT
0	prints an error message number n
greater than 0	assumes that a disk (RAMdisk) is in use

ERROR MESSAGE DEFINITIONS

When Max-FORTH detects an error condition, it may respond with an error message which corresponds to an error message number shown in the Table below. Max-FORTH clears the stacks and executes QUIT as its last actions when an error is processed, except for the message, NOT UNIQUE , which has no effect on stacks allowing Max-FORTH to continue execution normally.

Max-FORTH ERROR MESSAGES

Number	Message	Definition	Recovery Action
0	?	Echoed word was the most recently interpreted. The word is not in the dictionary or is not a valid number.	Check the word's name for spelling error or define the named word. Check if the number is valid under the current BASE or change BASE
1	STACK EMPTY	Parameter stack is empty.	Put more numbers into the stack or quit pulling out number from the empty stack.
2	DICTIONARY FULL	Dictionary space is used up. FIRST HERE is less than $A0	Increase dictionary space by moving FIRST or by FORGETing disposable word entries.
3	—	Not assigned	—
4	NOT UNIQUE	The <name> of the word just defined already exist in the dictionary.	Max-FORTH uses the latest definition of <name>. Reminder: previous definition is still in the system and is accessible by FORGETing the recent <name>.
5-6	—	Not assigned	—
7	FULL STACK	The parameter stack is full. The maximum stack entry is 39.	DROP or output some stack item.
8-16	—	Not assigned.	—
17	COMPILATION ONLY	The word just interpreted must be used inside of a definition.	Do not use the word for interpreting.
18	EXECUTION ONLY	The word just interpreted must be used outside of a definition.	Do not use the word in defining.
19	CONDITIONALS NOT PAIRED	Omitted words or incorrect nesting of conditionals	Correct or add the conditional pair.
20	DEFINITION NOT FINISHED	Definition is not finished or delimiter is missing.	Finish the definiton or add delimiter.

21	IN PROTECTED DICTIONARY	The word in question is below the FENCE .	Quit trying to FORGET a protected wor or move FENCE .
22	USE ONLY WHEN LOADING	Incorrect use of the word —> .	Use —> only when loading.
23	NO NAME	Attempt to create definition with 0 length name.	Use appropriate name.

Appendix B - NMI board manual

"100 Squared"TM System Documentation

By NEW MICROS INC.
1601 Chalk Hill Rd.
Dallas, Texas 75212

Covers: NMIX-0021 Rev. 1.0 10/10/86
NMIX-0022 Rev. 1.0
NMIT-0021 Rev. 1.0
NMIT-0022 Rev. 1.0
NMIX-0021 Rev. 2.0 & 2.1 12/13/87
NMIX-0022 Rev. 2.0 & 2.1
NMIT-0021 Rev. 2.0 & 2.1
NMIT-0022 Rev. 2.0 & 2.1

Getting Started

The "100 Squared"TM, when purchased in development configuration, is complete and ready to run. To operate the system, plug in the wall transformer and connect a terminal to the serial RS-232 DB25F connector. Most terminals should plug in directly, with a straight through cable (ie: pin 1 to pin 1, 2 to 2, 3 to 3, etc.). The"100 Squared"TM uses only lines 2 and 3 for serial in and serial out respectively, and pins 1 and 7 for ground. Many terminals require additional handshaking signals to work, so pins 4 and 5 are hooked together on the DB25F connector, as are pins 6 and 20. In this way the terminals that require the additional handshake signal have their own " clear to send" / "ready to send" and "data terminal ready" / "data set ready" signals wrapped back around, indicating "always ready".

In order to talk to the "100 Squared"TM the terminal must have the correct bit settings. The baud rate should be set at 9600 baud for 2 Mhz systems (8 Mhz crystal), 4800 for 1 Mhz systems (4 Mhz crystal). The "100 Squared"TM sends and receives a bit protocol of one start bit, eight data bits and one stop bits.

```
 ---+   +---+---+---+---+---+---+---+--------------
    | S | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | S |
    +---+---+---+---+---+---+---+---+---+
```

When the terminal is set correctly, every time you depress and release the red reset button the "100 Squared"TM should respond with:

```
Max-FORTH Vx.x
```

Seeing that message means the terminal. can see the "100 Squared"TM. Press "return" on your terminal several times. If the "100 Squared"TM responds with "OK" each time, communications are established.

Now you will want to see the system do something. Type WORDS followed by a return. This will cause the system to list its entire vocabulary, some 200+ words. The listing can be stopped at any time by pressing a key, like the space bar.

When the F68HC11 powers up, it assumes nothing else on the board is working, so it defaults to its own internal RAM. As a result there is a limited terminal input buffer area (16 characters) and dictionary space. The "100 Squared"TM provides external memory expansion. You now need to tell the system to move its terminal input buffer and dictionary to external memory. If the RAM is installed at 0100-1FFF (factory default for single 8K RAM) the following will accomplish that.

```
HEX
100 TIB !
50 TIB 2+ !
200 DP !
```

Now try a simple program to exercise some of these words. Enter:

```
: TYPE-LETTERS 5B 41 DO I EMIT LOOP ;
TYPE-LETTERS
```

to which the machine will respond:

```
: TYPE-LETTERS 5B 41 DO I EMIT LOOP ; OK
TYPE-LETTERS ABCDEFGHIJKLMNOPQRSTUVWXYZOK
```

Now have a look at memory with the DUMP command. Type:

```
0000 80 DUMP
```

and examine the results (remember we put the machine in HEX). Try another WORDS and observe the first word displayed. It has become the word TYPE-LETTERS entered above.

Your "100 Squared"TM is now running and communicating as it should. Its time to begin your design project by learning more about how to use the "100 Squared"TM.

The "100 Squared"TM, when purchased in the generic target configuration, is a minimum, 5 Volt only, configuration. The F68HC11, Xtal, reset circuit, various HC "glue" components and three 28 pin JEDEC sockets. Typically, a program developed in the "development configured" board will be installed in the "generic target configured" board for production of a dedicated application. The user must install the appropriate jumpers, which are not provided in the target configuration.

All configurations of the F68HC11 based "100 Squared"TM boards use the same base PC board. This includes the NMIX-0021, the NMIX-0022, the NMIT-0021 and the NMIT-0022. Configuration differences refer to the extent to which the board is filled with components.

PARALLEL PORTS

The F68HC11 has five parallel ports, Port A, B, C, D and E. Although some port lines have special multiplexed functions, they can all be used as inputs or as outputs according to their individual designs. Some of the port lines have direction registers allowing them to be used as either inputs or outputs. Two ports of the F86HC11 are sacrificed to create an 64K address and data bus. The 68HC24 simulates the replacement of those ports. Three ports of the F68HC11 and two replacement ports of the 68HC24 are brought out to connector J2. Power and ground are also available on J2.

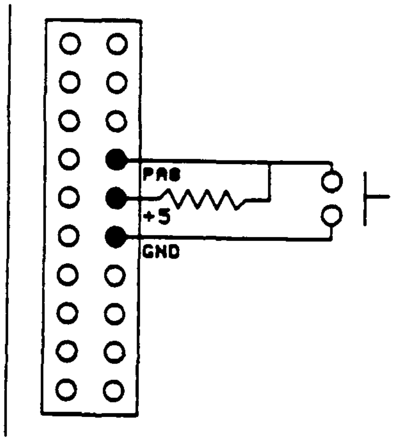
"100 SQUARED"TM DOCUMENTATION
INPUT/OUTPUT JACKS J2

```
                                 TOP VIEW
     FRONT (EDGE) OF CARD v
                             - X PA7 o o PA6 O
                             | O PA5 o o PA4 O
                             | O PA3 o o PA2 I
                             | I PA1 o o PA0 I
      20 pin header          |    +5 o o +5
      group                  |   GND o o GND
                             | O PB7 o o PB6 O
                             | O PB5 o o PB4 O
                             | O PB3 o o PB2 O
                             - O PB1 o o PB0 O
                                  +5 o o +5
                                 GND o o GND
                             - X PC7 o o PC6 X
                             | X PC5 o o PC4 X
                             | X PC3 o o PC2 X
                             | X PC1 o o PC0 X
                             |    +5 o o +5
                             |   GND o o GND
                             | O PD7 o o PD6 I
                             | X PD5 o o PD4 X
      34 pin header          | X PD3 o o PD2 X
      group                  | X PD1 o o PD0 X
                             |    +5 o o +5
                             |   GND o o GND
                             | I PE7 o o PE6 I
                             | I PE5 o o PE4 I
                             | I PE3 o o PE2 I
                             | I PE1 o o PE0 I
                             -    +5 o o +5
                                 GND o o GND
```

I=INPUT O=OUTPUT X=EITHER

The lines can be used as individual inputs or outputs or in combination. There are very few applications, however, where pins are switched dynamically, sometimes used as inputs, sometimes as outputs.

The simplest form of input device is a switch to ground, to create a low level when the switch is closed, with a pullup to give a high level when the switch is open. This switch can be breaker points, reed switch, the contacts of a relay, microswitch, etc. To try an example of this type input, hook up a simple push button switch to Port A Line 0 (PA0) with a 10K ohm pull up resistor to +5.

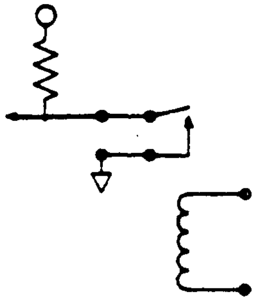


The following program will show the current state of the switch. Enter LOOK after pushing reset. (Reset sets the ports to all "ones".)

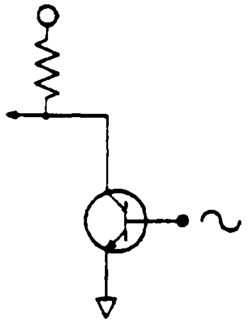
```
B000 CONSTANT PA
: SWITCH PA C@ 1 AND ;
: CHECK-STATE IF ." OPEN" ELSE ." CLOSED" THEN CR ;
: LOOK SWITCH BEGIN SWITCH 2DUP = IF DROP ELSE SWAP 0=
  CHECK-STATE THEN ?TERMINAL UNTIL ;
LOOK
```

Whenever the switch changes state, open or closed, the computer follows with a written report.

Other possible input devices are shown here.



RELAY



TRANSISTOR

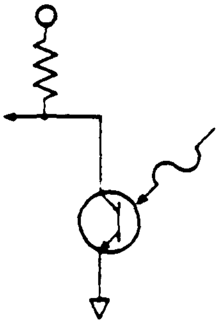
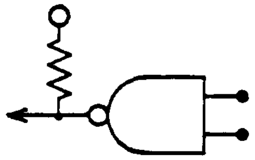
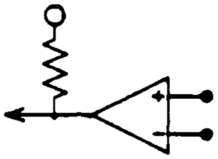


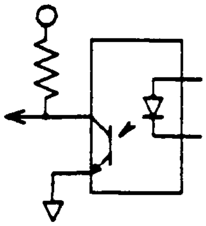
PHOTO TRANSISTOR



TTL LOGIC



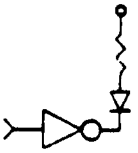
COMPARATOR



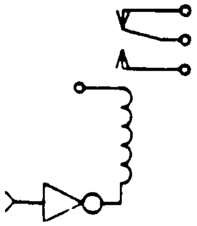
OPTO ISOLATOR

Note that due to the 10K pull up on the port, the "switch" must sink .5 ma to ground with no more voltage rise than an HC low level (2/10ths of Vcc) at the pin. (A voltage of 7/10 Vcc will always be recognised as a logical one.) Voltages applied above Vdd or below 0 Volts can damage the computer.

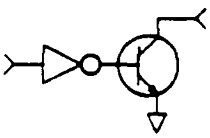
The outputs of the F68HC11 and 68HC24 can sink 1.6 ma to ground while letting the pin go no higher than 0.4 Volts for a "zero" and source about .8 ma at 4.5 Volts for a "one". In terms of control, this is a very small signal. Most relays require over 50 times more current to operate. LED's typically take 5 ma to be visible. HC levels are such that the output is sufficient to drive the input on one pin of one TTL device or about a dozen of the lower power LSTTL inputs. The output is sufficient to drive VMOS FET's and Darlingtons with an external pull up which can in turn control several amps of current. Usually, however, a buffer will be needed to do serious non-HC interfacing.



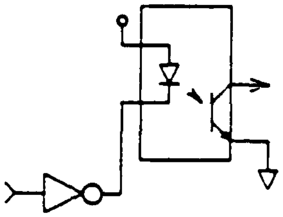
LED



RELAY

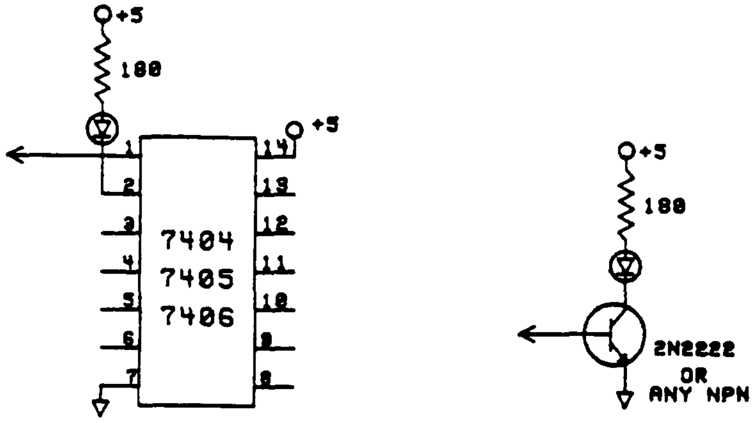


TRANSISTOR



OPTO ISOLATOR

To test the output capabilities, wire one of the two circuits shown here or use an oscilloscope or logic probe.



When the output is a "1" the LED will be on. When the output is a "0" the LED will be off. The following program will exercise the outputs of the 68HC24.

```
: RUN-UP FF B007 C! 0 BEGIN 1+ DUP B003 ! ?TERMINAL UNTIL ;
```

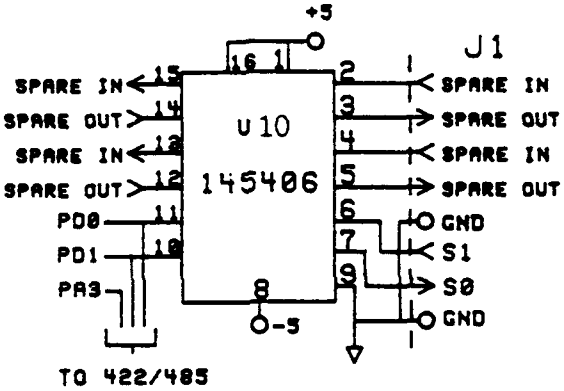
Notice that the low lines of Port B are changing so fast the LED appears to be on continuously at low brightness. Higher numbered Port B lines and Port C lines toggle at slower rates. Each bit position toggles at 1/2 the speed of the next lower bit.

SERIAL I/O

The F68HC11 has a full duplex hardware serial channel that operates at HC levels. To use this serial channel with most standard communications interfaces, level converters are needed. Drivers for RS-232C and IEEE 422/485 drivers are on the boards. (It should be noted that only one combination of RS-232 driver, RS-422 drivers or RS-485 driver should be used at one time to avoid contention of their receiver outputs.)

A zero by RS-232C specification is any voltage from +3 to +15 Volts, a one is between -3 and -15 Volts. To convert the HC signals to the voltage ranges of that interface standard, the "100 Squared"TM Rev. 1.0 uses a single 16 pin device, the MC145406.

The circuit is shown here.



The 145406 is ideally suited for this use. It not only provides an RS-232 receiver and transmitter pair for the F68HC11 processor, but also two spare RS-232 receiver and transmitter pairs which can be used with port lines for handshaking or software driven UARTS, etc..

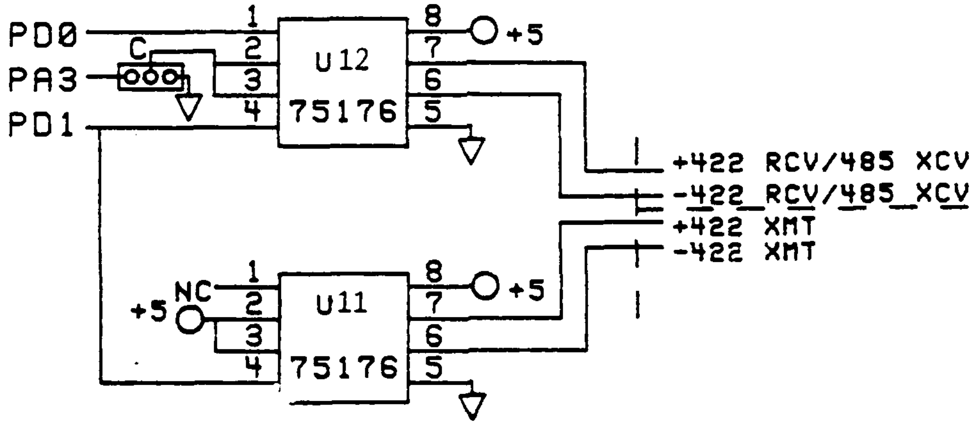
The RS-422 standard represents a relatively new interface now coming into popularity, and with good reason. Unlike the RS-232 requirements which specify a single wire voltage transmission referenced to ground, the RS-422 standard uses a voltage differential on a pair of conductors. While the RS-232 at full volatge drive levels in electrically noisy environments is barely reliable at distances to 1000 feet, RS-422 signals are considered reliable at distances up to 4000 feet. The 422 drivers operate, requiring only a single sided 5 Volt supply, over twisted pairs of wires. A full duplex connection for RS-422 requires two twisted pairs, one for transmit, one for recieve.

The RS-485 interface uses the same specifications for its transmitters and receivers. It, however, allows a single twisted pair to be used for incoming and outgoing messages. This is accomplished by having both a transmitter (with 3 state abiltity) and a reciever tied in parallel to the same twisted pair. Multiple drop point communications are possible under this scheme (up to 64 pairs by specification). Of course, in application the transmitter turns on and takes control of the lines only under software control. The actual implementation of this control will be determined by the particular protocol being used in the communication network. Usually one master sends an addresses message to one of multiple slaves and then turns off its master transmitter. The addressed slave, recognizing its address will turn on its transmitter and respond with the requested data.

These two interfaces are accomodated on the "100 Squared"TM by the addition of two 8 pin 75176's, which each contain a transmitter/receiver pair. Whether the transmitter of the pair is active, or not, is controlled by a signal on one of its pins.

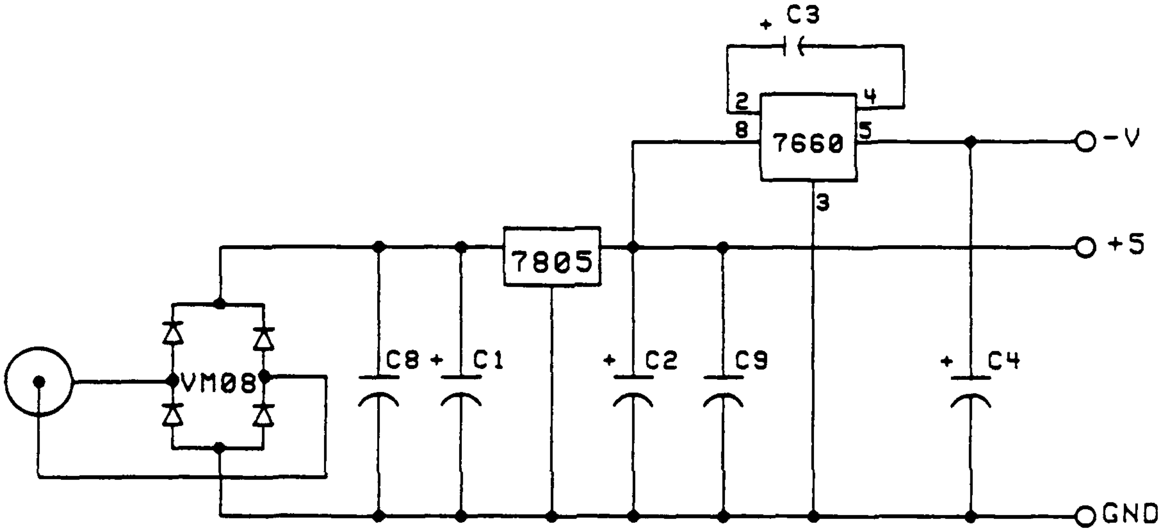
One of the 75176's (U11) has its receiver always enabled. It is used exclusively as the RS-422 receiver. The other 75176 (U12) can be used as the RS-422 transmitter if jumper C on the "100 Squared"TM is grounded (ie: in 422 position), or it can be used as the receiver and transmitter for the RS-485 interface as controlled by PA3 (ie: in 485 position). In this case if PA3 is high, the 75176's transmitter is not active. If PA3 is low its transmitter is active.

The RS-422/485 interface circuit is shown below.



POWER SUPPLY

The power supply circuit on the "100 Squared"TM is designed to allow the board to operate from a simple AC wall transformer. It has three major sub circuits - rectification, regulation and DC to DC conversion. Rev 2.x added battery backup capabilities to the 28 pin JEDEC sockets and the F68HC11 internal RAM, and an improved power-up power-down reset circuit.



The bridge rectifier converts the AC to DC. The 7805 regulates this rectified incoming voltage to a constant 5 Volts.

The most unusual feature of the power supply is the use of the DC to DC converter, the ICL 7660. On NMIx-002x boards the 7660 is fed from the 5 Volt rail. The two voltages are used to power the RS-232 converter circuit. This means the maximum output from the RS-232 converter would be + and - 5 Volts.

The upper limit of +V is set by the ability of the 7805 to dissipate heat. If a heat sink is added to the 7805, voltages in excess of 20 Volts are possible. Driving the 7805 to hard, however, will cause it to enter thermal overload and "shut down" its output.

The typical current required by the "100 Squared"TM with 8K CMOS RAM and the Max-FORTH ROM at 2 Mhz from 9 VAC is 20 ma.

The power terminal, J3, can be used as an alternate power source instead of the AC supply. The 5 Volts applied at the terminal is also applied to the 7660. The 5 Volt +/- rails are usually sufficient to generate more than the +/- 3 Volts needed to meet the RS-232 specification. Some terminals, however, may not fully meet those requirements.

BATTERY BACK UP AND RESET
(Rev 2.x only)

The battery backup capability added to the Rev 2.x boards to allow data retention in otherwise volitale CMOS RAMs and the processors own internal RAM through main board power downs. A third terminal has been added to the power connector, J3, marked VBB for Voltage Battery Backup.

The VBB terminal on J3 is connected to the VBB supply rail on the board by diode, D1. The VBB supply rail supplied the three 28 pin JEDEC sockets, the 8054HN low voltage indicator in the reset circuit, the 74HC00 gate and the 74HC138 decoder. If no power is applied to the VBB terminal, the VBB rail is supplied through the intrinsic diode of P channel FET, Q1, to within a diode drop of the suppling 5 volt rail (~4.4 Volts). When the 8054HN low voltage indicator releases the reset line, Q1 is turned on and the VBB comes almost completely up to the 5 volt rail (~4.95 Volts). (This may cause some problem with the Dallas Semiconductor DS1223 battery sockets, as they "write protect" their RAMs at 4.75 Volts. Running an elevated 5 Volt supply may be necessary to accomodate these parts. The purpose of this new feature is, however, to do away with the need for those devices in final system configurations.)

When the 8054HN low voltage indicator holds the reset line low (when VBB is below 3.8-4.2 Volts), Q1 is turned off and the address decoder is disabled through the same input that is used by MEMDIS. This "access" protects the memories during the power down cycle.

To meet the full letter of the specifications of the parts involved the correct backup voltage on the VBB pin is critical. This supply must be low enough to ensure that after the diode drop of D1, the VBB rail cause the 8054HN to issue a reset (~4.0 Volts), otherwise Q1 will remain on and the whole system will be powered by VBB. It must also be high enough to ensure that after the diode drop of D1, the VBB rail will meet the processors required backup volatge (listed as 4.0 Volts). Therefore, the ideal voltage for the VBB supply is 4.3-4.5 Volts. It should be pointed out however that the Motorola specification appears to be overly conservative. By empirical test, VBB supplies below 3 Volts appear to be quite adequate. Most CMOS RAMs will retain data down to 2.2 Volts. Accounting for the diode drop under such low currents, the VBB supply may work as low as 2.5 Volts.

The proccess battery backup supply enters the chip via the MODB pin. Jumper block D controls the setting of MODB, either to ground or to VBB. For backup of the processor's RAM to be successful jumpers D and E must be in the Single Chip or Expanded Multiplexed settings. When the VBB supply is used on the processor, it will retain its User Area through power down ano remember its linkages to the external FORTH dictionary.

ADDRESS DECODING

The chip selects of the three JEDEC sockets are generated by a 74HC138. When jumpers A and B are in the 8K position, address lines A15 to A13 are brought to this part. This means that each of the eight generated chip selects represent a single 8K byte segment out of the 64K byte memory map.

When jumpers A and B are in the 16K position, address lines A15 and A14 are brought to this part. The A13 is held high. This means that the upper four generated chip selects represent a single 16K byte segment out of the 64K byte memory map.

When jumpers A and B are in the 32K position, address lines A15 alone controls the part. The A14 and A13 are held high. This means that each of the two upper chip selects represent a 32K byte segments out of the 64K byte memory map.

Two other signals control the decoder - Address Strobe (AS) and On Board Memory Disable (MEMDIS). The Address Strobe (AS) signal must be active low before any chip selects are enabled. This is the processor's signal indicating the address on the bus is valid for the off-chip memory. The On Board Memory Disable (MEMDIS) signal allows an offboard open collector source to disable the on board decoder, so offboard components can usurp a memory segment from on board memory, even if the entire 64K is filled with RAM on the main board.

```
                           74HC138
         A13  A  +5V   +----u----+
          o   o   o  --|A     Vcc|-+5V   +------+
         A14  B  +5V   |         |       |      |
          o   o   o  --|B      00|-00----| o  o |-+-U2 CHIP SELECT
                       |         |       |      | |
REV 2.x           A15-|C      01|-01----| o  o |-+
     v                 |         |       |      | |
AS                  E-|E      02|-02----| o  o |-+
MEMDIS                 |         |       |      | |
  +                AS-|E      03|-03----| o  o |-+
RESET                  |         |       |      | |
E              MEMDIS-|E      04|-04----| o  o |-+
                       |         |       |      |
              +---07-|07     05|-05----| o  o |-+-U3 CHIP SELECT
              |        |         |       |      | |
              |  GND-|GND    06|-06----| o  o |-+
              |        +---------+       |      |
              +--------------------------| *--* |---U4 CHIP SELECT
                                         +------+
```

* Rev 1.0 boards do not have a jumper block in this postion - U4 Chip Select is hard wired to the socket. On 2.x boards this jumper block is installed - the jumpered connection of the high order chip select to U4 is user selectable.

TROUBLESHOOTING

As always the first thing to do when troubleshooting is to check the power and ground connections. An oscilliscope should be used to check signals. The heat sink of the 7805 is a convenient place to hook a ground clip. If +5 Volts is present at J3 and the board is not operational, the next item to check is the oscillator. Putting the scope on EXTAL (Pin 7) should show a 8 Mhz sine wave (4 Mhz F68HC11 parts running 4 Mhz XTAL's) running from about .5 Volt lows to 4.5 Volt peaks. XTAL (F68HC11 Pin 8) should have an identical signal, but of a much smaller amplitude. If the sine waves are not present and there is 5V present at the power pin Vcc (Pins 26), and ground at Vss (Pin 52), then either the F68HC11 or the crystal are bad and require replacement. There is one exception. If the processor has executed a STOP instruction, the oscillator will stop. When the oscillator is functioning correctly a 2 Mhz (1 Mhz) clean running square wave should be present at the E output (Pin 5). The E signal drives the timing for all external memory transfers. This signal should transition nearly rail to rail, a 0.4V low and a 4.6V high are normal. Less amplitude can indicate a board short or an excessive load on the line external to the F68HC11.

The serial channel should send a sign on message if no autostart ROM interferes. If not, the reset circuit could be bad, the serial converter could have failed, or the F68HC11 could be defective. With the reset button depressed the RES pin (Pin 17) should be at ground. When release, the pin should rise to 5 Volts in about a quarter second. If the reset pin is working and still no message is seen on the terminal, check PD1, the serial output line (Pin 33). When reset is exercised, this line should go from normally high through a multitude of toggles back to a high state. The periods of the toggle transitions are multiples of approximately 100 microseconds. If this signal is not present, and there are no user ROMs in the board, the F68HC11 is suspect. If the signal is present, check pin 3 of the DB25F connector. It should normally be at -V (-5 Volts nominally) and should toggle to +V (+5 Volts nominally) at the same rate as the serial output line. If this is happening and no message is seen, the RS-232 wiring or the terminal is suspect. Check to see if J1 is connected to the DB25F RS-232 connector as follows:

DB25F	Signal Name
1	Case ground
2	Serial in (to "100 Squared"TM)
3	Serial out (from "100 Squared"TM)
7	Electrical ground

Check the voltages on pins 2 and 3. If pin 3 is very negative and pin 2 is floating, both systems are trying to talk on the same line. Pins 2 and 3 need to be swapped. Usually this is done with a "null modem" inserted where the two systems connect.

If the -V/+V signal was not found at pin 3, the RS-232 converter is not working. Check pin 1 of the 145406 for +V and pin 8 of the 145406 for -V. If -V is not present at the -V pin, the 7660 has failed. Pin 7 of the 145406, the output, should look the same as pin 3 of J1.

Check pin 2 of J1 which is the serial into the board from the terminal. It should normally be at a negative voltage between -3 and -15 Volts. When a key is pressed on the terminal it should pulse to positive voltages between +3 and +15 Volts. If it doesn't, the terminal or the RS-232 wiring are suspect. The same signals at inverted TTL levels, should also be at PD0, which is the serial input line of the processor (Pin 34).

The most common error in trying to use the "100 Squared"TM is mismatched baud rates or bit settings. Verify that the terminal is set for 9600 baud with one start bit, eigth data bits and one stop bits, with no parity generated. (Review this discussion in the Getting Started section.)

MEMORY MAP

```
K#  HEX
--  -----
64  $FFFF   +-------------+
63             RUN TIME   |
62              KERNEL    |
61                        |
60          NON RUN TIME  |  Max-FORTH ROM
59             CODES      |
58                        |
57             HEADS      |
56  $E000                 |
    $DFFF   _____________ |
            |             |
            |             |
            |             |
            |             |
   $B800    |             |
   $B600    |_____________|
            |_____________| EEPROM

   $B000    |=============|  REGISTERS
            |             |
            -             -



            -             -
            |             |
5           |             |
4  $1000    |$0B_AT_$103B |
3  $0C00    |             |
2  $0800    |             |
1  $0400    |             |
0  $0000    +ON=CHIP=RAM=+
```

"100 SQUARED"TM DOCUMENTATION
MISCELLANEOUS JUMPERS

#	SOURCE	DESTINATION	NORMALLY
A			
A13-A	ADDRESS LINE 13	ADDRESS DECODER INPUT	
A-5	+5 VOLT RAIL		
B			
A13-A	ADDRESS LINE 13	ADDRESS DECODER INPUT	
A-5	+5 VOLT RAIL		
C			
00-U2	DECODER OUTPUT 0	U2 JEDEC SOCKET	
01-U2	DECODER OUTPUT 1	U2 JEDEC SOCKET	
02-U2	DECODER OUTPUT 2	U2 JEDEC SOCKET	
03-U2	DECODER OUTPUT 3	U2 JEDEC SOCKET	
04-U2	DECODER OUTPUT 4	U2 JEDEC SOCKET	
05-U3	DECODER OUTPUT 5	U3 JEDEC SOCKET	
06-U3	DECODER OUTPUT 6	U3 JEDEC SOCKET	
07-U4	DECODER OUTPUT 7	U4 JEDEC SOCKET *	
D			
GND-D	GROUND	MODB PIN	OPEN
D-5	MODB PIN	+5 VOLT RAIL	CLOSED
E			
GND-E	GROUND	MODA PIN	OPEN
E-5	MODA PIN	+5 VOLT RAIL	CLOSED
F			
XIRQ-B	NMI	INT FROM J4	OPEN
B-IRQ	INT FROM J4	PA3 EDGE SENSITIVE LINE	OPEN
G			
485-C	PA3	U12 PINS 2 & 3	
C-422	U12 PINS 2 & 3	GROUND	
I			
U2	U2 PIN 27 R/W LINE	U2 PIN 28 SUPPLY	OPEN**
J			
U3	U3 PIN 27 R/W LINE	U3 PIN 28 SUPPLY	OPEN**
K			
U4	U4 PIN 27 R/W LINE	U4 PIN 28 SUPPLY	OPEN**

* Rev 1.0 is hard wired to U4, Rev 2.x is jumper selectable

** Rev 2.x has option of pullups on R/W lines to write protect RAMs in socket. To use install 100K pullup resistor & remove jumper from 28 pin JEDEC selection socket for pin 27. If battery backup is in use, RAM will then emulate ROM.

"100 SQUARED"TM DOCUMENTATION
GENERAL PURPOSE SOCKET

Jumper Assignments for JEDEC 28 Pin Sockets

```
                                                      +---+
JUMPER  1 o                       o 28 +5             | o |  *
                                                      |   |
   A12  2                           27 JUMPER         | o |
                                                      +---+
    A7  3 o                       o 26 JUMPER

    A6  4 o                       o 25 A8

    A5  5 o                       o 24 A9

    A4  6 o                       o 23 A11

    A3  7 o                       o 22 OE

    A2  8 o                       o 21 A10
                                       ___________
    A1  9 o                       o 20 CHIP SELECT

    A0 10 o                       o 19 D7

    D0 11 o                       o 18 D6

    D1 12 o                       o 17 D5

    D2 13 o                       o 16 D4

   GND 14 o                       o 15 D3

                PIN 1    PIN 26   PIN 27
                0---0    0---0    0---0

                0    0   0    0   0    0
               A14  +5  +5   A13 A14  RR/W
```

*   Rev 2.x has option of pullups on R/W lines to write protect RAMs in socket. To use, install 100K pullup resistor & remove jumper for pin 27. If battery back up is in use, RAM will then emulate ROM.

"100 SQUARED"TM DOCUMENTATION
GENERAL PURPOSE SOCKET - U6, U7, U8

Jumper Settings for Standard JEDEC 24/28 Pin Devices

ALL 8K X 8 DEVICES
2764
2864
6264

```
 PIN 1   PIN 26  PIN 27
+---+---+---+---+---+---+
|   | X | X |   |   | X | *
|   | X | X |   |   | X |
+---+---+---+---+---+---+
 A14 +5V +5V A13 A14 RR/W
```

16K X 8 EPROM
27128

```
 PIN 1   PIN 26  PIN 27
+---+---+---+---+---+---+
|   | X |   | X |   | X |
|   | X |   | X |   | X |
+---+---+---+---+---+---+
 A14 +5V +5V A13 A14 RR/W
```

32K X 8 EPROM
27256

```
 PIN 1   PIN 26  PIN 27
+---+---+---+---+---+---+
|   | X |   | X | X |   |
|   | X |   | X | X |   |
+---+---+---+---+---+---+
 A14 +5V +5V A13 A14 RR/W
```

32K X 8 RAM
62256

```
 PIN 1   PIN 26  PIN 27
+---+---+---+---+---+---+
| X |   |   | X |   | X | *
| X |   |   | X |   | X |
+---+---+---+---+---+---+
 A14 +5V +5V A13 A14 RR/W
```

* Rev 2.x has option of pullups on R/W lines to write protect RAMs in socket. To use, install 100K pullup resistor & remove jumper for pin 27. If battery backup is in use, RAM will then emulate ROM.

"100 SQUARED"TM DOCUMENTATION
GENERAL PURPOSE SOCKET - U6, U7, U8

Jumper Settings for Various Addressing Schemes

3 8K DEVICES

```
           A13 A +5V
          +---------+
          | XXXX  o |
          +---------+   8K POSITION
          | XXXX  o |
          +---------+
           A14 B +5V

             C
          +-------+                       0000
0000-1FFF 00 | XXXX |-+-U2 CHIP SELECT  ----
          |       | |                    1FFF
2000-3FFF 01 | o  o |-+
          |       | |
4000-5FFF 02 | o  o |-+
          |       | |
6000-7FFF 03 | o  o |-+
          |       | |
8000-9FFF 04 | o  o |-+
          |       |                       C000
A000-BFFF 05 | o  o |-+-U3 CHIP SELECT  ----
          |       | |                    DFFF
C000-DFFF 06 | XXXX |-+
          |       |                       E000
E000-FFFF 07 | *--* |---U4 CHIP SELECT  ----
          +-------+                       FFFF
```

* Rev 1.0 is hard wired to U4, Rev 2.x is jumper selectable

```
                      3  16K DEVICES

               A13 A +5V
              +---------+
              | o  XXXX |
              + --------+ 16K POSITION
              | XXXX  o |
              +---------+
               A14 B +5V

                C
             +------+
          00 | o  o |-+
             |      | |                0000
0000-3FFF 01 | XXXX |-+-U2 CHIP SELECT ----
             |      | |                3FFF
          02 | o  o |-+
             |      | |
4000-7FFF 03 | o  o |-+
             |      | |
          04 | o  o |-+
             |      |                  8000 **
8000-BFFF 05 | XXXX |-+-U3 CHIP SELECT ----
             |      | |                BFFF
          06 | o  o |-+
             |      |                  C000
C000-FFFF 07 | *--* |---U4 CHIP SELECT ----
             +------+                  FFFF
```

* Rev 1.0 is hard wired to U4, Rev 2.x is jumper selectable

** See appnote on PRU overmapping cautions

2 32K DEVICES

```
                  A13 A +5V
                 +---------+
                 | o  XXXX |
                 +---------+ 32K POSITION
                 | o  XXXX |
                 +---------+
                  A14 B +5V

                     C
                 +------+
            00   | o  o |-+
                 |      | |
            01   | o  o |-+
                 |      | |
            02   | o  o |-+
                 |      | |                  0000
 0000-7FFF  03   | XXXX |-+-U2 CHIP SELECT   ----
                 |      | |                  7FFF
            04   | o  o |-+
                 |      |
            05   | o  o |-+-U3 CHIP SELECT   ----
                 |      | |
            06   | o  o |-+
                 |      |                    8000 **
 8000-FFFF  07   | *--* |---U4 CHIP SELECT   ----
                 +------+                    FFFF
```

* Rev 1.0 is hard wired to U4, Rev 2.x is jumper selectable
** See appnote on PRU overmapping cautions

"100 SQUARED"TM DOCUMENTATION
SERIAL INPUT/OUTPUT JACKS J1

TOP VIEW
NUMBERED LEFT TO RIGHT

```
1  2  3  4  5  6  7  8  9 10 11 12 13 14
----------------------------------------
o  o  o  o  o  o  o  o  o  o  o  o  o  o
```

DB25F	J1	Signal Name
	1	Spare RS-232 in
	2	Spare RS-232 out
	3	Spare RS-232 in
	4	Spare RS-232 out
1	5	Case ground
2	6	Serial into "100 Squared"TM
3	7	Serial out of "100 Squared"TM
7	8	Electrical ground
	9	Reset line in or out
	10	Electrical ground
	11	RS-422 Receive + Differential input or 485 xcv
	12	RS-422 Receive - Differential input or 485 xcv
	13	RS-422 Receive + Differential output
	14	RS-422 Receive - Differential output

"100 SQUARED"TM DOCUMENTATION
INPUT/OUTPUT JACKS J2

```
                                     TOP VIEW
FRONT (EDGE) OF CARD v
                            - X PA7 o o PA6 O
                            | O PA5 o o PA4 O
                            | O PA3 o o PA2 I
                            | I PA1 o o PA0 I
 20 pin header              |    +5 o o +5
 group                      |   GND o o GND
                            | O PB7 o o PB6 O
                            | O PB5 o o PB4 O
                            | O PB3 o o PB2 O
                            - O PB1 o o PB0 O
                                 +5 o o +5
                                GND o o GND
                            - X PC7 o o PC6 X
                            | X PC5 o o PC4 X
                            | X PC3 o o PC2 X
                            | X PC1 o o PC0 X
                            |    +5 o o +5
                            |   GND o o GND
                            | O PD7 o o PD6 I
                            | X PD5 o o PD4 X
 34 pin header              | X PD3 o o PD2 X
 group                      | X PD1 o o PD0 X
                            |    +5 o o +5
                            |   GND o o GND
                            | I PE7 o o PE6 I
                            | I PE5 o o PE4 I
                            | I PE3 o o PE2 I
                            | I PE1 o o PE0 I
                            -    +5 o o +5
                                GND o o GND
```

I=INPUT O=OUTPUT X=EITHER

"100 SQUARED"TM DOCUMENTATION
EXPANSION JACK J4

MEMDIS	o	o	N.C.
E	o	o	RST
A15	o	o	INT
A14	o	o	+5
A12	o	o	R/W
A7	o	o	A13
A6	o	o	A8
A5	o	o	A9
A4	o	o	A11
A3	o	o	OE
A2	o	o	A10
A1	o	o	AS
A0	o	o	D7
D0	o	o	D6
D1	o	o	D5
D2	o	o	D4
GND	o	o	D3

The J4 expansion connector was designed to follow the JEDEC standard for byte sized memory parts in the 8, 16 and 32K Byte varieties. The J4 connector on these boards are made to most closely match the more recently available 32K JEDEC parts.

NEW MICROS, INC.

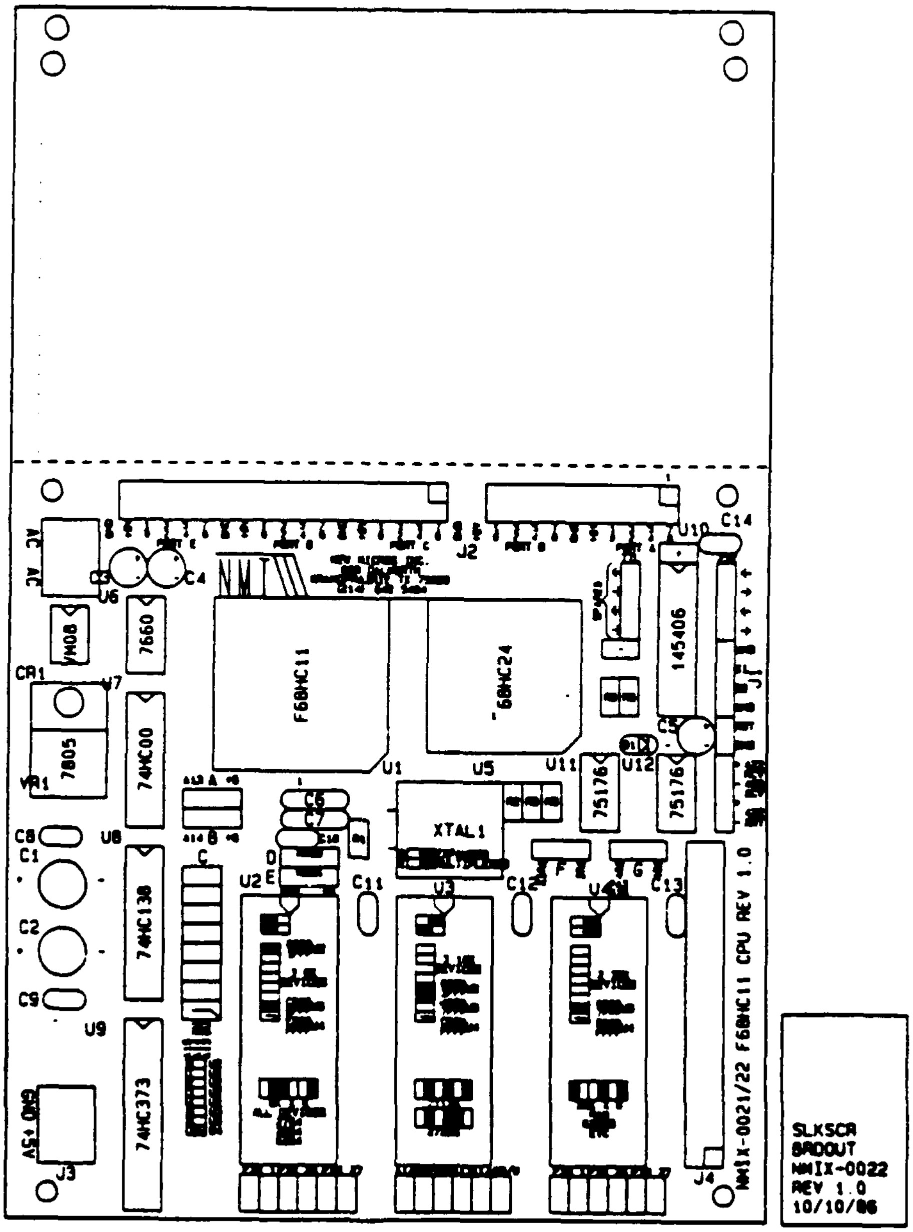
NMIX-0021/2 F68HC11 "100 SQUARED"TM PARTS LIST REV 1.0 & 2.x

PART#	GENERIC	DESCRIPTION
U1	52 PIN SOCKET	
	F68HC11	FORTH CPU
U2	28 PIN SOCKET	
	2064	8K x 8 RAM
U3	28 PIN SOCKET	
U4	28 PIN SOCKET	
	2064/20256	OPTIONAL MEMORIES U4,5
U5	44 PIN SOCKET	
	F68HC24 PRU	PRT RPLCMNT UNT (NMIX-0022 ONLY)
U6	8 PIN SOCKET	
	7660	DC/DC CONVERTER
U7	14 PIN SOCKET	
	74HC00	NAND GATE
U8	16 PIN SOCKET	
	74HC138	ADDRESS DECODING PROM
U9	20 PIN SOCKET	
	74HC373	8 BIT LATCH
U10	16 PIN SOCKET	
	145406	RS-232 CONVERTOR
U11,12	8 PIN SOCKETS	
	75176	RS-422/485 DRIVERS
Y1	8 MHZ XTAL	
J3	SCREW TERMINAL	2 PIN .194" CONN (Rev 1.0 only)
J3	SCREW TERMINAL	3 PIN .194" CONN (Rev 2.x only)
J4	34 PIN VSC HEADER	.1" DUAL INLINE
R0	10K	1/8 WATT RESISTOR (Rev 2.x only)
R1	1Meg	1/8 WATT RESISTOR
R2-5	10K	1/8 WATT RESISTOR
R6	10K	1/8 WATT RESISTOR (Rev 1.0 only)
C1,2	220uf	16V ELECTROLYTIC CAP
C3,4	10uf	16V ELECTROLYTIC CAP
C5	10uf	16V ELECTROLYTIC CAP (Rev 1.0 only)
C6,7	20 pf	CERAMIC DISC
C8-14	.1uf	MONOLYTHIC BYPASS
C15	.1uf	MONOLYTHIC BYPASS (Rev 2.x only)
LVI1	8054HN	POWER ON RESET MONITOR (Rev 2.x)
Q1	VP0300L	P CHANNEL FET (Rev 2.x only)
D1	1N4148 OR 1N914	SIGNAL DIODE
CR1	VM08	BRIDGE RECTIFIER
VR1	7805	5V REGULATOR
PCB	100 SQUARED	NMIX-0021/2/3 PCB REV 1.0 or 2.x
	JUMPER PINS	BERG STYLE .1" CENTER JUMPERS
	JUMPER SHUNTS	BERG STYLE .1" CENTER SHUNTS
S1	MOMENTARY PUSH	RESET SWITCH
	9V WALL PLUG	A.C. POWER TRANSFORMER
	CASE	ALUMINUM EXTRUDED METAL CASE
	FRONT PANEL	ALUMINUM MOUNTING FACE PLATE
	BACK PANEL	STEEL BLACK END PLATE

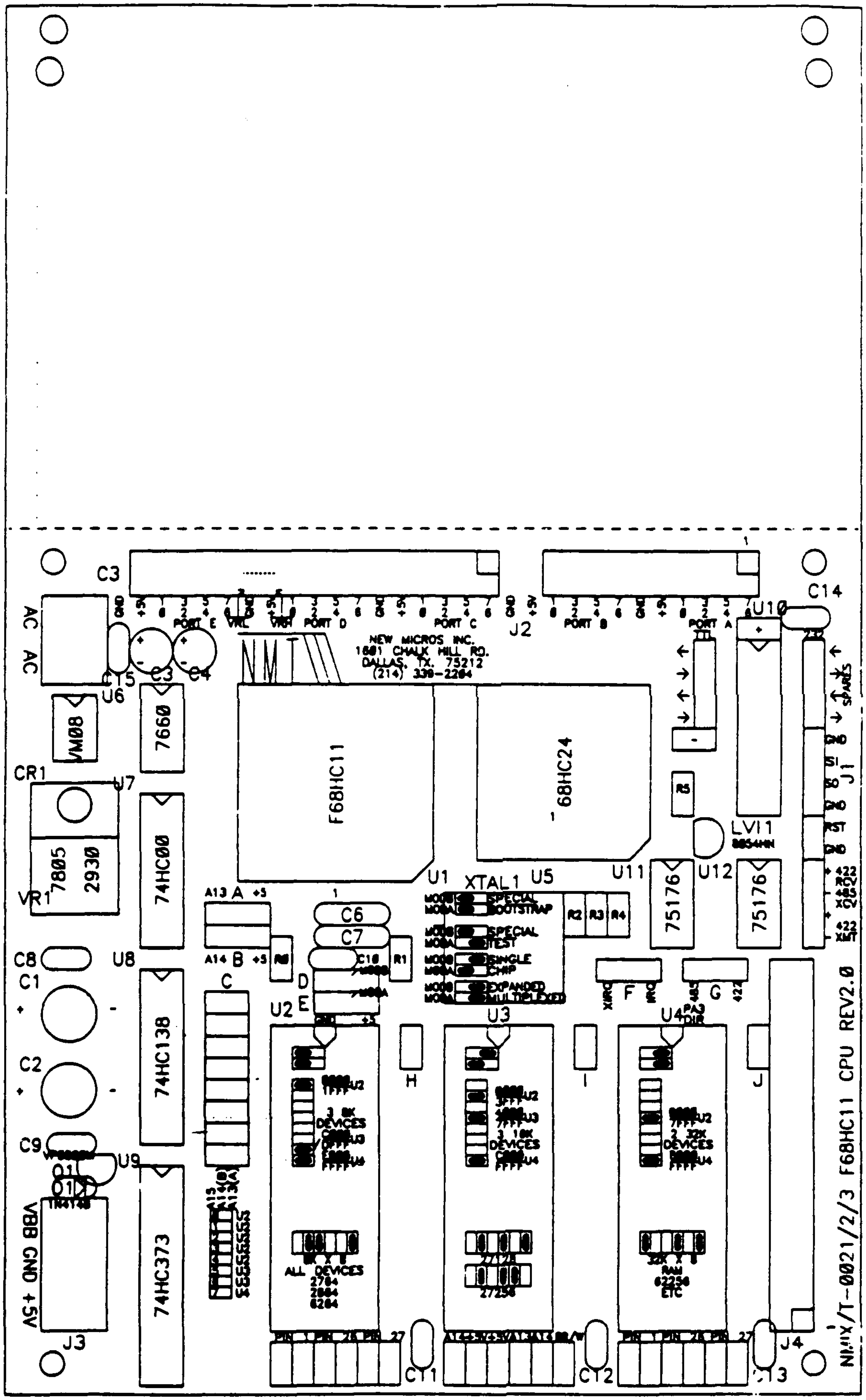
NEW MICROS, INC.
NMIT-0021/2 F68HC11 "GENERIC TARGET COMPUTER"TM PARTS LIST
REV 1.0 & 2.x

PART#	GENERIC	DESCRIPTION
U1	52 PIN SOCKET	
	F68HC11	FORTH CPU
U2	28 PIN SOCKET	
U3	28 PIN SOCKET	
U4	28 PIN SOCKET	
U7	14 PIN SOCKET	
	74HC00	NAND GATE
U8	16 PIN SOCKET	
	74HC138	ADDRESS DECODING PROM
U9	20 PIN SOCKET	
	74HC373	8 BIT LATCH
Y1	8 OR 4 MHZ XTAL	
R0	10K	1/8 WATT RESISTOR (Rev 2.x only)
R1	1Meg	1/8 WATT RESISTOR
R2-5	10K	1/8 WATT RESISTOR
R6	10K	1/8 WATT RESISTOR (Rev 1.0 only)
C6,7	20 pf	CERAMIC DISC
C9-13	.1uf	MONOLYTHIC BYPASS
C15	.1uf	MONOLYTHIC BYPASS (Rev 2.x only)
LVI1	8054HN	POWER ON RESET MONITOR (Rev 2.x)
Q1	VP0300L	P CHANNEL FET (Rev 2.x only)
D1	1N4148 OR 1N914	SIGNAL DIODE
PCB	100 SQUARED	NMIX-0021/2/3 PCB REV 1.0 or 2.x

NMIX-0022 REV 1.0 SILKSCREEN

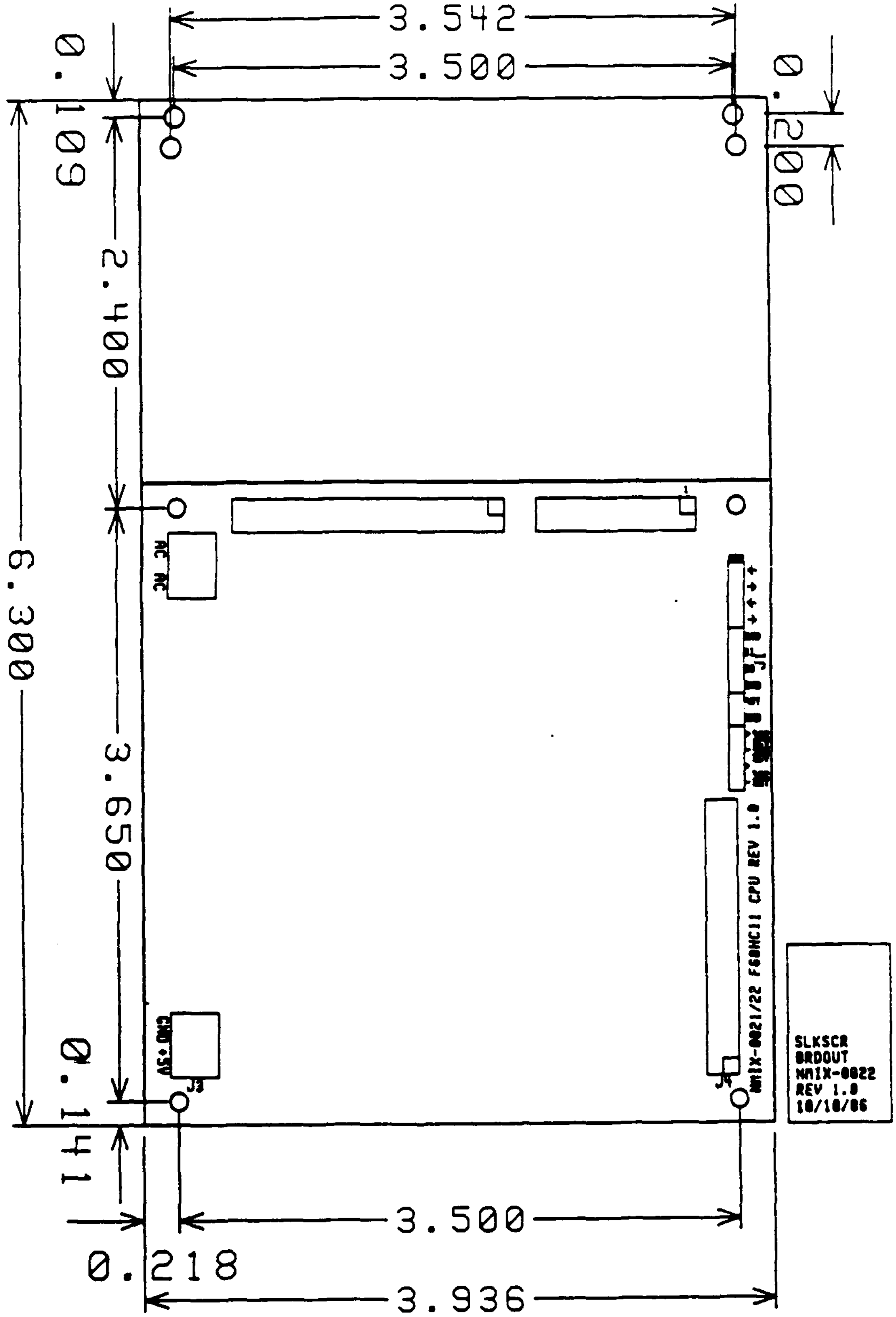


SLKSCR
BRDOUT
NMIX-0022
REV 1.0
10/10/86

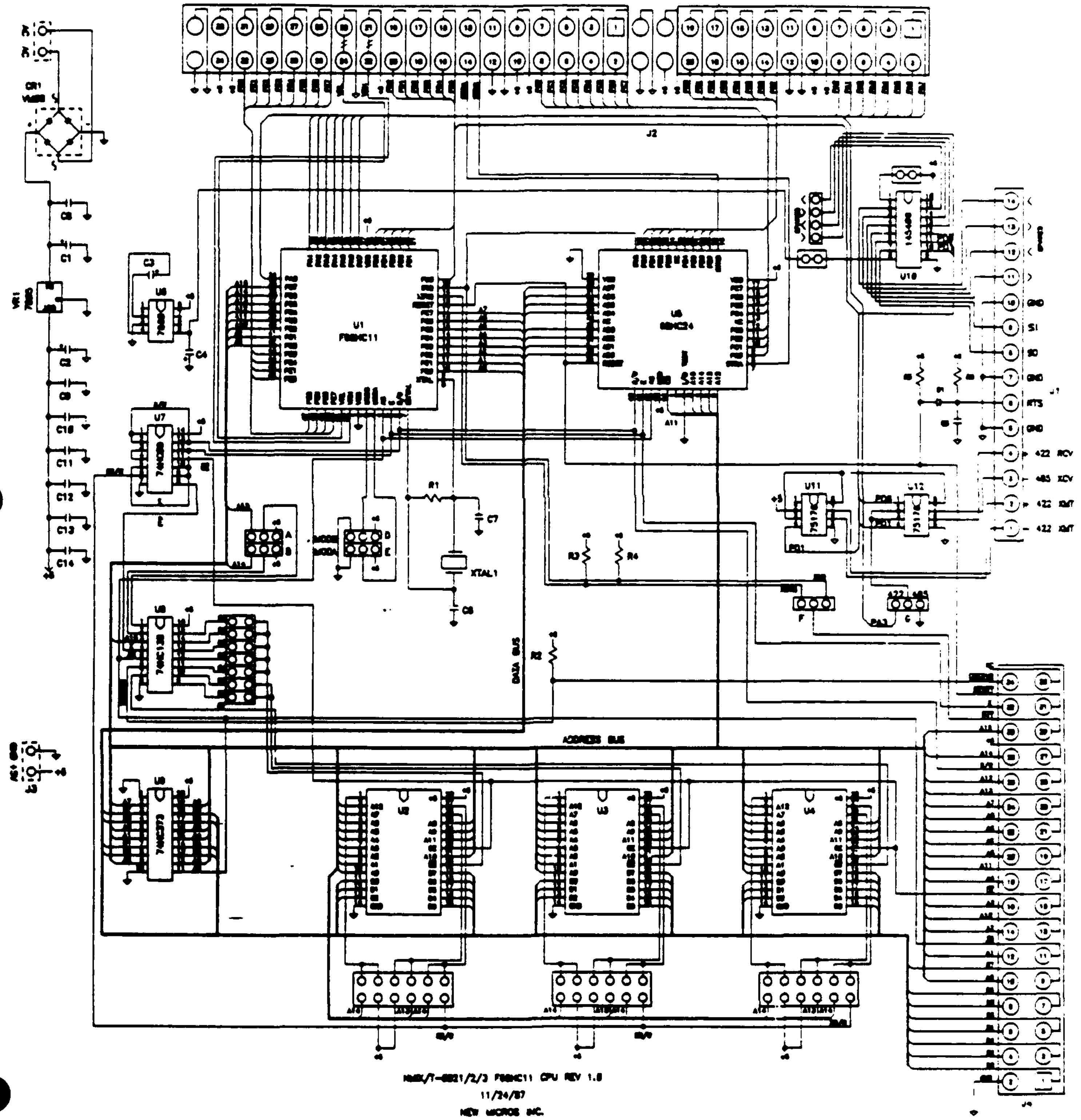


SLKSCR
BRDOUT
NMIx-002x
REV 2.0
9/28/87

NMIX-0023 REV 1.0 & 2.x MECHANICALS

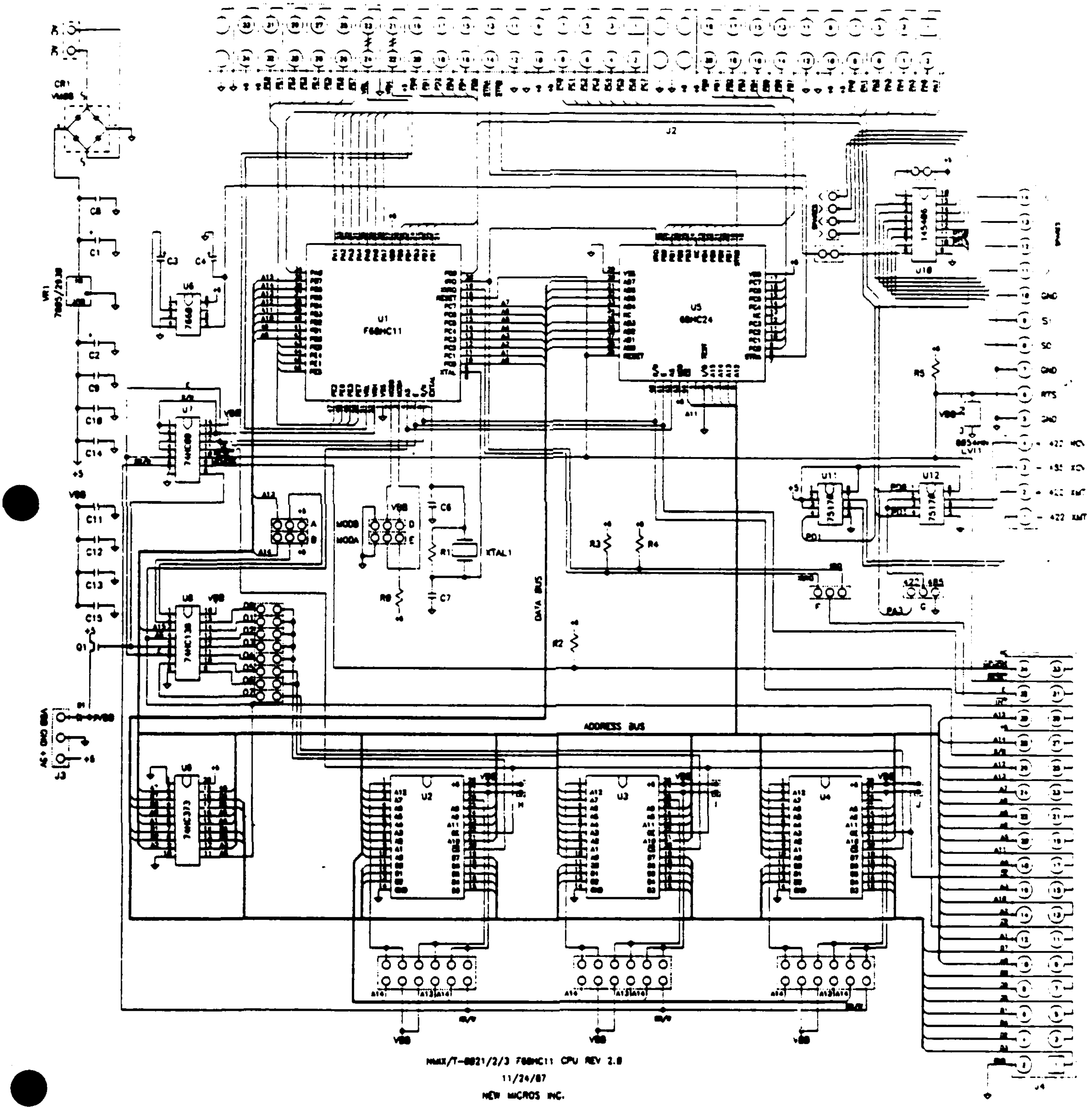


"schematic REV 1.0"



NMIX/T-0021/2/3 F68HC11 CPU REV 1.0
11/24/87
NEW MICROS INC.
1601 CHALK HILL RD.
DALLAS TX. 75212
214-339-2204

"schematic REV 2.x"



NMIX/T-0021/2/3 F68HC11 CPU REV 2.0

11/24/87

NEW MICROS INC.

1601 CHALK HILL RD.

DALLAS TX. 75212

· 4-330-2204

"100 SQUARED"TM DOCUMENTATION
APPLICATION NOTE

CONNECTING A PARALLEL PRINTER TO THE "100 SQUARED"TM

Being able to keep a hard copy of entered or displayed text can be a very desirable feature during development. Further, the ability to make written reports from a run time application may be required of the finished system. The hook-up below shows connections between the "100 SQUARED"TM and a Centronics style printer.

This example works only on the NMIX and NMIT-0022 with the Port Replacement Unit installed.

PRINTER	SIGNAL	CABLE	J2	F68HC11
CONNECTOR	NAME	CONDUCTOR		68HC24
PIN #		NUMBER	34	SIGNAL
1	STB	1	13	STRB (PD7)
19	GND	2	11	GND
2	D0	3	8	PC0
20	GND	4		GND
3	D1	5	7	PC1
21	GND	6		GND
4	D2	7	6	PC2
22	GND	8		GND
5	D3	9	5	PC3
23	GND	10		GND
6	D4	11	4	PC4
24	GND	12		GND
7	D5	13	3	PC5
25	GND	14		GND
8	D6	15	2	PC6
26	GND	16		GND
9	D7	17	1	PC7
27	GND	18		GND
10	ACK	19	14	STRA (PD6)
28	GND	20		GND
11-18	N.C.			
29-36	N.C.			

"100 SQUARED"TM DOCUMENTATION
APPLICATION NOTE

INTEL FORMAT DUMP COMMAND

The following program allows a section of memory to be dumped out the serial channel in the Intel hex format which is a standard used by many of the commercially available PROM programmers. This program should allow the use of such programmers to capture programs and data in EPROMs, which are not supported for programming by the "100 SQUARED"TM directly.

```
HEX

VARIABLE CHKSUM

: CE DUP A < IF 30 ELSE 37 THEN + EMIT ; ( CONVERT AND EMIT )
: 2.R FF AND 10 /MOD CE CE ;
: 4.R 0 100 UM/MOD 2.R 2.R ;

: INTEL-DUMP ( addr count --- )
  OVER + SWAP ( CONVERTS ADDR & COUNT TO UPPER, LOWER ADDR )
  BEGIN
    CR
    2DUP 20 + MIN ( MAKE NEXT LINE OF OUTPUT UP TO 32 BYTES LONG)
    SWAP ( BRING UP START ADDRESS, MOVE DOWN END ADDRESS )
    ." :" ( BEGIN THE RECORD )
    2DUP - ( FIND OUT # OF BYTES IN THIS RECORD )
    DUP CHKSUM ! ( BEGIN CHKSUM COMPUTATION )
    2.R ( PRINT # OF BYTES IN RECORD IN TWO DIGIT FIELD )
    DUP 100 /MOD + CHKSUM +! ( ADD START ADDRESS TO CHKSUM )
    DUP 4.R ( PRINT START ADDRESS IN FOUR DIGIT FIELD )
    ." 00" ( PRINT RECORD TYPE, NO NEED TO ADD TO CHKSUM )
    >R DUP R> ( MAKE START STOP #S FOR DO LOOP )
    DO
      I C@ 2.R ( PRINT HEX BYTE IN TWO DIGIT FIELD )
      I C@ CHKSUM +! ( UPDATE CHKSUM )
    LOOP
    CHKSUM C@ NEGATE 2.R ( PRINT CHKSUM NEGATED TWO DIGIT FIELD )
    2DUP =
  UNTIL ( KEEP GOING TILL LINE END IS = TO BLOCK END )
  CR ." ;00000001FF" CR ( TACK ON END RECORD )
  2DROP
;
```

Program and application courtesy of Danny Barger, International Computing Scale.

Special consideration needs to be given to the address requirements of the Port Replacement Part (PRU) if it is optionally installed in a board by the user. Since the PRU is outside the F68HC11, it must compete for its address space with other devices on the bus, while the internal registers do not. It is possible to accidentally "over-map" the 68HC11 PRU with another external memory device unless some care is given to where the Max-FORTH system is mapping its registers.

Particular care must be taken when setting one of these boards for the 32K addressing mode which will always cause problems for the upper 32K device and the PRU. Generally the 8K address mode is the safest mode to use when a PRU is installed, if care is given to the U2 jumper to prohibit "over-mapping".

In Max-FORTH revisions x.1 (and prior) the registers are at $9000. Later revisions put the registers at $B000. Only address lines A15 - A11 are attached to the PRU, so it will over map a 2 Kbyte area (i.e. 9000-97FF or B000-B7FF) with a skipping pattern that repeats every 64 (40 hex) locations corresponding to the registers it provides.

An off board address decoder could be used to disable the on board memory using MEMDIS. It would need to generate a low signal on MEMDIS when ever there was any address in the area occupied by the PRU. In this way the memories in the 28 pin sockets would be "notched out" to allow the PRU to function normally.

```
           74HC688
       + ---u----+
  GND- |E     Vcc| -+5V

  GND- |A0    A=B| -MEMDIS

  GND- |B0     B7| -+5V

  GND- |A1     A7| -A15

  GND- |B1     B6| -+5V

  GND- |A2     A6| -A14

  GND- |B2     B5| -GND

  A11- |A3     A5| -A13

  GND- |B3     B4| -+5V

  GND- |GND    A4| -A12
       +---------+
```

Above 74HC688 detects addresses B000-B7FF and controls MEMDIS.

Appendix C - R65C52

Rockwell

R65C52
Dual Asynchronous Communications Interface Adapter (DACIA)

DESCRIPTION

The Rockwell CMOS R65C52 Dual Asynchronous Communications Interface Adapter (DACIA) provides an easily implemented, program controlled two-channel interface between 8-bit microprocessor-based systems and serial communication data sets and modems.

The DACIA is designed for maximum programmed control from the microprocessor (MPU) to simplify hardware implementation. Dual sets of registers allow independent control and monitoring of each channel.

Transmitter and Receiver bit rates may be controlled by an internal baud rate generator or external times 16 clocks. The baud rate generator accepts either a crystal or a clock input, and provides 15 programmable baud rates. When a 3.6864 MHz crystal is used, the baud rates range from 50 bps to 38,400 bps.

The DACIA may be programmed to transmit and receive frames having word lengths of 5, 6, 7 or 8 bits; even, odd, space, mark or no parity; and 1 or 2 stop bits.

A Compare Register, and the ability to detect address frames, facilitate address recognition in a multidrop mode.

FEATURES

- Low power CMOS N-well silicon gate technology
- Two independent full duplex channels with buffered receivers and transmitters.
- Data set/modem control functions
- Internal baud rate generator with 15 programmable baud rates (50 bps to 38,400 bps)
- Program-selectable internally or externally controlled receiver and transmitter bit rates
- Programmable word lengths, number of stop bits, and parity bit generation and detection
- Programmable interrupt control
- Edge detect for $\overline{DCD}$, $\overline{DSR}$, and $\overline{CTS}$
- Program-selectable echo mode for each channel
- Compare Register
- Address/Data frame recognition
- 5.0 Vdc ± 5% supply requirements
- 40-pin plastic or ceramic DIP or 44-pin PLCC
- Full TTL or CMOS input/output compatibility
- Compatible with R6500 and R65C00 microprocessors and R6500/* microcomputers

ORDERING INFORMATION

Part Number:
R65C52 _ _ _

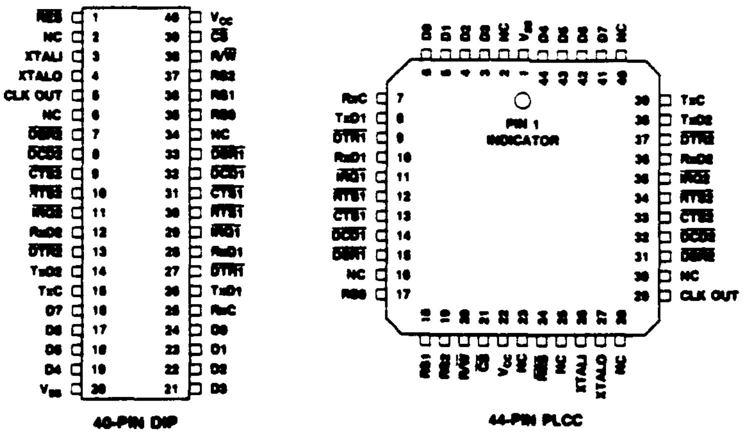
Temperature Range (T_L to T_H):
Blank = 0°C to +70°C
E = −40°C to +85°C

Frequency Range:
1 = 1 MHz
2 = 2 MHz
3 = 3 MHz

Package
C = 40-Pin Ceramic DIP
P = 40-Pin Plastic DIP
J = 44-Pin Plastic Leaded Chip Carrier (PLCC)

INTERFACE SIGNALS

The DACIA is available in a 40-pin DIP or a 44-pin PLCC. Figure 1 shows the pin assignments for each package. The DACIA interface signals are shown in Figure 2. Table 1 contains a description of each signal.



40-PIN DIP

44-PIN PLCC

NC = NO CONNECTION. NO SIGNAL SHOULD BE CONNECTED TO THIS PIN.

Figure 1. R65C52 Pin Assignments

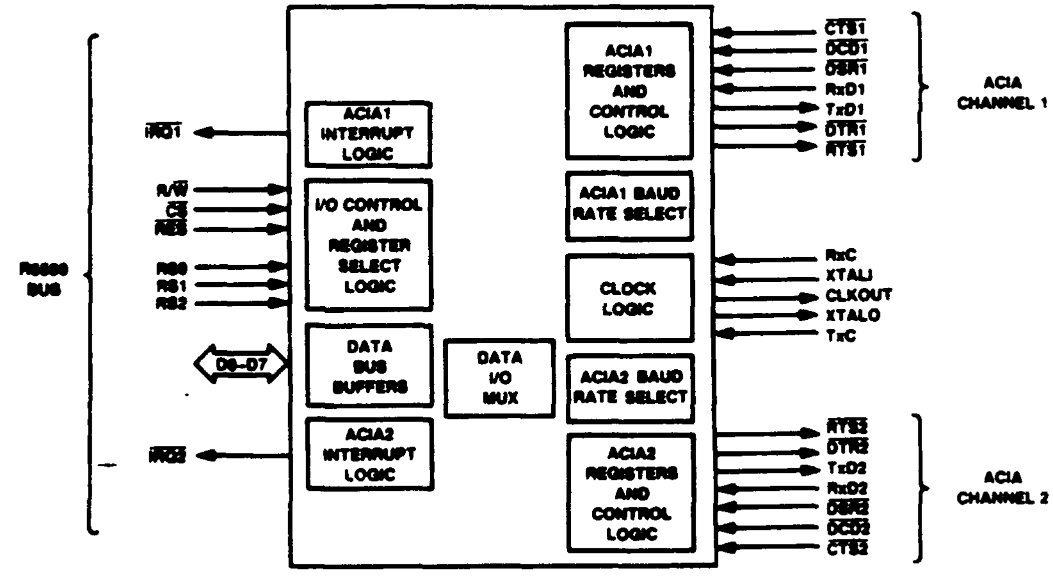


Figure 2. R65C52 DACIA Interface Signals

Table 1. DACIA Interface Signal Definitions

	Pin No.			
Signal	**DIP**	**PLCC**	**I/O**	**Name/Description**
Host Interface				
$\overline{RES}$	1	24	I	**Reset.** Active low input controlling the reset function. This signal must be driven low for a minimum of 4 μs for a valid reset to occur. It is driven high during normal operation.
R/$\overline{W}$	38	20	I	**Read/Write.** Input controlling the direction of data transfer. It is driven low during write cycles, and is driven high at all other times.
$\overline{CS}$	39	21	I	**Chip Select.** Active low input enabling data transfers between the host CPU and the DACIA. The DACIA latches register selects and the R/$\overline{W}$ input on the falling edge of $\overline{CS}$. It latches input data on the rising edge of $\overline{CS}$.
RS0–RS3	35–37	17–19	I	**Register Select.** Three inputs controlling access to the DACIA internal registers. Table 3 lists the coding for each register.
D0–D3 D4–D7	24–21 19–16	6–3 44–41	I/O	**Data Bus.** Eight bidirectional lines used to transfer data between the host and the DACIA. These lines output data during READ cycles when $\overline{CS}$ is low. At all other times, they are in the high impedance state.
$\overline{IRQ1}$ $\overline{IRQ2}$	29 11	11 35	O	**Interrupt Request.** Two active low, open-drain outputs from the interrupt control logic. These outputs are *normally high. An* $\overline{IRQ}$ *line goes low when one of the flags of the associated* ISR is set if the corresponding enable bit is set in the IER.
Clock Interface				
XTALI XTALO	3 4	26 27	I O	**Crystal Input/Output.** One input and one output through which the reference signal for the internal clock oscillator is supplied. A parallel resonant crystal may be connected across the pins or a clock may be input at XTALI. When a clock is used, XTALO must be left open.
CLK OUT	5	29	O	**Clock Out.** A buffered output from the internal clock oscillator which is in phase with XTALI. This output may be used to drive the XTALI input of another DACIA. Therefore, several DACIA chips may be driven with one crystal.
RxC	25	. 7	I	**Receiver Clock.** Input for external 16x receiver clock.
TxC	15	39	I	**Transmitter Clock.** Input for external 16x transmitter clock.
Serial Channel Interface				
$\overline{DTR1}$ $\overline{DTR2}$	27 13	9 37	O	**Data Terminal Ready.** Two general purpose outputs which are set high upon reset. The output level is programmed by setting the appropriate bit in the associated Format Register (FR) high or low. The state of each $\overline{DTR}$ line is reflected by the $\overline{DTR}$ LVL bit in the associated Control Status Register (CSR).
$\overline{DSR1}$ $\overline{DSR2}$	33 7	15 31	I	**Data Set Ready.** Two general purpose inputs. An active transition sets the $\overline{DSR}$T bit in the Interrupt Status Register (ISR). The $\overline{DSR}$ LVL bit in the associated CSR reflects the current state of a $\overline{DSR}$ line.
$\overline{RTS1}$ $\overline{RTS2}$	30 10	12 34	O	**Request To Send.** Two general purpose outputs which are set high upon reset. The output level is programmed by *setting the appropriate bit in the associated FR high or low. The state of an* $\overline{RTS}$ *line is* reflected by the $\overline{RTS}$ LVL bit in the associated CSR.
$\overline{CTS1}$ $\overline{CTS2}$	31 9	13 33	I	**Clear To Send.** The $\overline{CTS}$ control line inputs allow handshaking by the transmitters. When $\overline{CTS}$ is low, the data is transmitted continuously. When $\overline{CTS}$ is high, the Transmit Data Register Empty bit (TDRE) in the associated ISR is not set. The word presently in the Transmit Shift Register is sent normally. *Any active transition on a* $\overline{CTS}$ line sets the $\overline{CTS}$T bit in the appropriate ISR. The $\overline{CTS}$ LVL bit in the associated CSR reflects the current state of $\overline{CTS}$.
TxD1 TxD2	26 14	8 38	O	**Transmit Data.** The TxD outputs transfer serial non-return to zero (NRZ) data to the data communications equipment (DCE). The data is transferred, LSB first, at a rate determined by the baud rate generator or external clock.
$\overline{DCD1}$ $\overline{DCD2}$	32 8	14 32	I	**Data Carrier Detect.** Two general purpose inputs. An active transition sets the $\overline{DCD}$T bit in the appropriate ISR. The $\overline{DCD}$ LVL bit in the associated CSR reflects the current state of a $\overline{DCD}$ line.
RxD1 RxD2	28 12	10 36	I	**Receive Data.** *The RxD inputs transfer serial NRZ data into the DACIA from the DCE*, LSB first. The receiver baud rate is determined by the baud rate generator or external clock.
Power				
VCC	40	22	I	**DC Power Input.** 5.0V ± 5%.
VSS	20	1	I	**Power and Signal Reference.**

FUNCTIONAL DESCRIPTION

Figure 3 is a block diagram of the DACIA which consists of two asynchronous communications interface adapters with common microprocessor interface control logic and data bus buffers. The individual functional elements of the DACIA are described in the following paragraphs.

RESET LOGIC

The Reset Logic sets various internal registers, status bits and control lines to a known state. The $\overline{RES}$ input must be driven low for a minimum of 4 μs for a valid reset to occur. At this time, the IERs are set to $80, the RDRs and ACRs are cleared, and the compare mode is disabled. Also, the $\overline{DTR}$ and $\overline{RTS}$ outputs are driven high and the $\overline{CTS}$, $\overline{DCD}$ and $\overline{DSR}$ transition detect flags are cleared. No other bits are affected.

DATA BUS BUFFER

The Data Bus Buffer is a bidirectional interface between the data lines and the internal data bus. The state of the Data Bus Buffer is controlled by the I/O Control Logic and the Interrupt Logic. Table 2 summarizes the Data Bus Buffer states.

I/O CONTROL LOGIC

The I/O Control Logic controls data transfers between the Internal Registers and the Data Bus Buffer. Internal Register selection is determined by the Register Select inputs as shown in Table 3. When R/$\overline{W}$ is high and $\overline{CS}$ is low, data from the selected register is transferred from the internal data bus to the data lines. When $\overline{CS}$ is high, the DACIA is deselected and the data lines are tri-stated.

INTERRUPT LOGIC

The interrupt logic causes the $\overline{IRQ}$ lines ($\overline{IRQ1}$ or $\overline{IRQ2}$) to go low when conditions are met that require the attention of the MPU. *There are two registers (the Interrupt Enable Register and the Interrupt* Status Register) involved in the control of interrupts in the DACIA. An $\overline{IRQ}$ will be asserted *on the transition* of one of the flags in an ISR from 0 to 1 if the corresponding bit in the associated IER is set. The $\overline{IRQ}$ line is negated when the ISR is read or when the interrupting condition is cleared. CAUTION: When the interrupt is generated by TDRE, 1/16 of a bit time must elapse before $\overline{IRQ}$ can be cleared by reading the ISR.

CLOCK OSCILLATOR LOGIC

The internal clock oscillator supplies the time base for the baud rate generator. The oscillator can be driven by a crystal or an external clock.

The baud rate generator may be disabled by connecting XTALI to ground and leaving XTALO open. When this is done, a transmitter times 16 clock must be input at TxC, a receiver times 16 clock must be input at RxC and the Control Registers must be programmed to select TxC and RxC clocks.

Table 2. Data Bus Buffer Summary

Control Signals		Data Bus Buffer State
R/$\overline{W}$	$\overline{CS}$	
L	L	Write Mode — Tri-State
H	L	Read Mode — Output Data
X	H	Deselected — Tri State

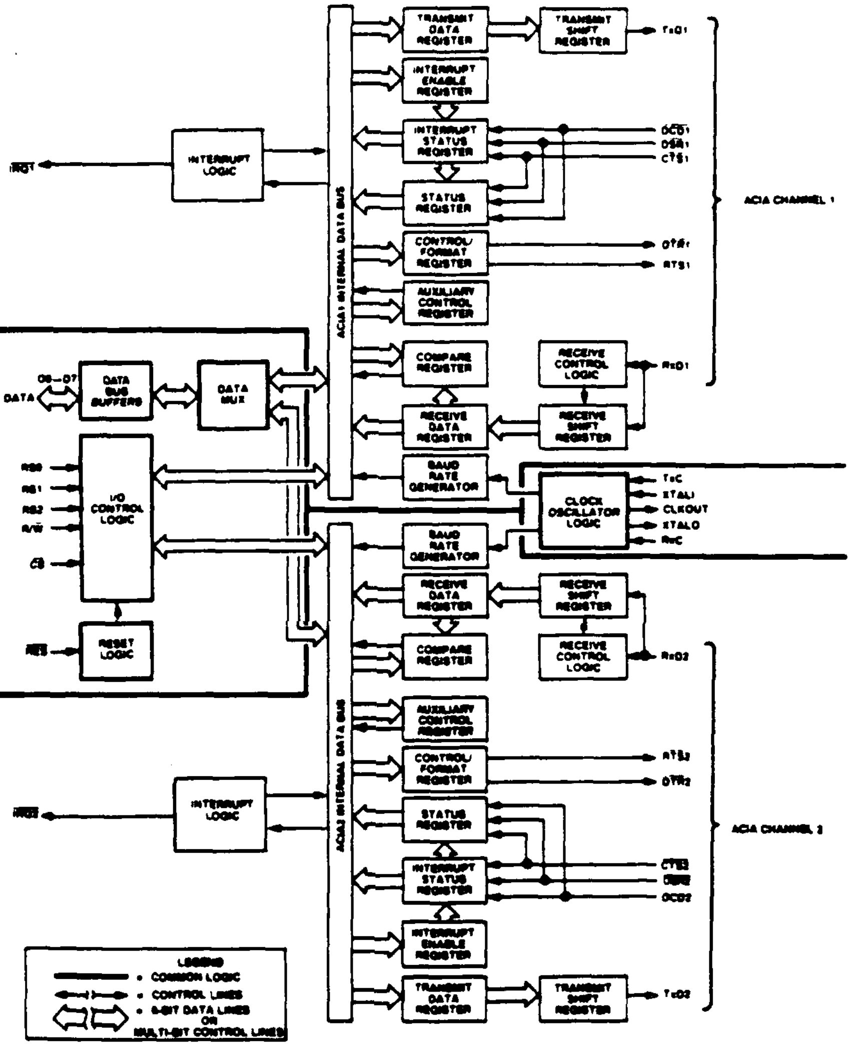


Figure 3. DACIA Block Diagram

Table 3. DACIA Register Selection

Register Select Lines				Register Accessed			
				Write		Read	
HEX	RS2	RS1	RS0	Symbol	Name	Symbol	Name
0	L	L	L	IER1	Interrupt Enable Register 1	ISR1	Interrupt Status Register 1
1	L	L	H	CR1	Control Register 1[1]	CSR1	Control Status Register 1
				FR1	Format Register 1[2]		
2	L	H	L	CDR1	Compare Data Register 1[3]		Not Used
				ACR1	Auxiliary Control Register 1[4]		
3	L	H	H	TDR1	Transmit Data Register 1	RDR1	Receive Data Register 1
4	H	L	L	IER2	Interrupt Enable Register 2	ISR2	Interrupt Status Register 2
5	H	L	H	CR2	Control Register 2[1]	CSR2	Control Status Register 2
				FR2	Format Register 2[2]		
6	H	H	L	CDR2	Compare Data Register 2[3]		Not Used
				ACR2	Auxiliary Control Register 2[4]		
7	H	H	H	TDR2	Transmit Data Register 2	RDR2	Receive Data Register 2

Notes:

1. D7 must be set low to write to the Control Registers.
2. D7 must be set high to write to the Format Registers.
3. Control Register bit 6 must be set to 0 to access the Compare Register.
4. Control Register bit 6 must be set to 1 to access the Auxiliary Control Register.

SERIAL DATA CHANNELS

Two independent serial data channels are available for the full duplex (simultaneous transmit and receive) transfer of asynchronous frames. Separate internal registers are provided for each channel for the selection of frame parameters (number of bits per character, parity options, etc.), status flags, interrupt control and handshake. The asynchronous frame format is shown in Figure 4.

Transmit data from the host system is loaded into the Transmit Data Register. From there, it is transferred to the Transmit Shift Register where it is shifted, LSB first, onto the TxD line. All transmissions begin with a start bit and end with the user selected number of stop bits. A parity bit is transmitted before the stop bit(s) if parity is enabled.

Receive data is shifted into the Receive Shift Register from the associated RxD line. Start and stop bits are stripped from the frame and the data is transferred to the Receive Data Register. Parity bits may be discarded or stored in the ISR.

Five I/O lines are provided for each channel for handshake with the data communications equipment (DCE). Four of these signals ($\overline{RTS}$, $\overline{DTR}$, $\overline{DSR}$ and $\overline{DCD}$) are general purpose inputs or outputs. The fifth signal, $\overline{CTS}$, enables/disables the transmitter. When $\overline{CTS}$ is high and the Transmit Shift Register is empty, the transmitter (except for Echo Mode) is inhibited. When $\overline{CTS}$ is low, the transmitter is enabled.

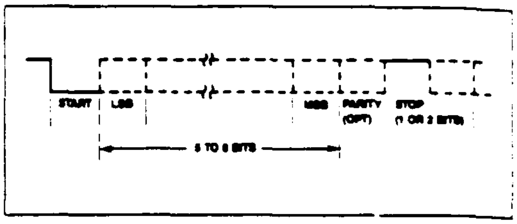


Figure 4. Asynchronous Frame Format

INTERNAL REGISTERS

The DACIA contains ten control registers and four status registers in addition to the transmit and receive registers. The Control Registers provide for control of frame parameters, baud rate, interrupt generation, handshake lines, transmission and reception. The status registers provide status information on transmit and receive registers, error conditions and interrupt sources. Table 4 summarizes the bit definitions of these registers. A detailed description follows.

Table 4. Register Formats

Register Select (Hex)	Register	R/$\overline{W}$	Bit 7	6	5	4	3	2	1	0	Reset Value 76543210
0 4	ISR1 ISR2	R	ANY BIT SET	TDRE	$\overline{CTS}$T	$\overline{DCD}$T	$\overline{DSR}$T	PAR	F/O/B	RDRF	1 - 00000 -
0 4	IER1 IER2	W	CLR/SET BITS	TDRE IE	$\overline{CTS}$T IE	$\overline{DCD}$T IE	$\overline{DSR}$T IE	PAR IE	F/O/B IE	RDRF IE	- 0000000
1 5	CSR1 CSR2	R	FE	TUR	$\overline{CTS}$ LVL	$\overline{DCD}$ LVL	$\overline{DSR}$ LVL	BRK	$\overline{DTR}$ LVL	$\overline{RTS}$ LVL	1 - - - - 011
1 5	CR1 CR2	W	0	CDR/ ACR	STOP BITS	ECHO	BIT RATE SEL				0 - - - - - - -
1 5	FR1 FR2	W	1	DATA BITS		PAR SEL		PAR EN	$\overline{DTR}$ CNTL	$\overline{RTS}$ CNTL	1 - - - - - - -
2 6	CDR1 CDR2	W (CR6 = 0)	COMPARE DATA								- - - - - - - -
2 6	ACR1 ACR2	W (CR6 = 1)	UNUSED						TRNS BRK	PAR ERR/ST	- - - - - - 00
3 7	RDR1 RDR2	R	RECEIVE DATA REGISTER								00000000
3 7	TDR1 TDR2	W	TRANSMIT DATA REGISTER								- - - - - - - -

INTERRUPT STATUS REGISTERS (ISR1, ISR2)

The Interrupt Status Registers are read-only registers indicating the status of each interrupt source. Bits 6 through 0 are set when the indicated $\overline{\text{IRQ}}$ c ndition has occurred. Bit 7 is set to a 1 when any $\overline{\text{IRQ}}$ source bit is set, or if Echo Mode is disabled, when $\overline{\text{CTS}}$ is high.

7	6	5	4	3	2	1	0
ANY BIT SET	TDRE	$\overline{\text{CTS}}$T	$\overline{\text{DCD}}$T	$\overline{\text{DSR}}$T	PAR	F/O/B	RDRF

Address = 0,4 Reset Value = 1 - 00000 -

Bit 7 **Any Bit Set**
1 Any bit (6 through 0) has been set to a 1 or $\overline{\text{CTS}}$ is high with echo disabled
0 No bits have been set to a 1 or echo is enabled

Bit 6 **Transmit Data Register Empty (TDRE)**
1 Transmit Data Register is empty and $\overline{\text{CTS}}$ is low
0 Transmit Data Register is full or $\overline{\text{CTS}}$ is high

Bit 5 **Transition On $\overline{\text{CTS}}$ Line ($\overline{\text{CTS}}$T)**
1 A positive or negative transition has occurred on $\overline{\text{CTS}}$
0 No transition has occurred on $\overline{\text{CTS}}$, *or ISR has* been Read

Bit 4 **Transition On $\overline{\text{DCD}}$ Line ($\overline{\text{DCD}}$T)**
1 A positive or negative transition has occurred on $\overline{\text{DCD}}$
0 No transition has occurred on $\overline{\text{DCD}}$, or ISR has been Read

Bit 3 **Transition On $\overline{\text{DSR}}$ Line ($\overline{\text{DSR}}$T)**
1 A positive or negative transition has occurred on $\overline{\text{DSR}}$
0 No transition has occurred on $\overline{\text{DSR}}$, or ISR has been Read

Bit 2 **Parity Status (PAR)**
ACR bit 0 = 0
1 A parity error has occurred in received data
0 No parity error has occurred, or the Receive Data Register (RDR) has been Read
ACR bit 0 = 1
1 Parity bit = 1
0 Parity bit = 0

Bit 1 **Frame Error, Overrun, Break**
1 *A framing error, receive overrun, or receive break* has occurred or has been detected
0 *No error, overrun, break has occurred or RDR* has been Read

Bit 0 **Receive Data Register Full (RDRF)**
1 Receive Data Register is full
0 Receive Data Register is empty

INTERRUPT ENABLE REGISTERS (IER1, IER2)

The Interrupt Enable Registers are write-only registers that enable/disable the $\overline{\text{IRQ}}$ sources. $\overline{\text{IRQ}}$ sources are enabled by writing to an IER with bit 7 set to a 1 and the bit for every $\overline{\text{IRQ}}$ source to be enabled set to a 1. $\overline{\text{IRQ}}$ sources are disabled by writing to an IER with bit 7 reset to a 0 and the bit for every source to be disabled set to a 1. Any source bit reset to 0 is unaffected and remains in its original state. *Thus, writing $7F to an IER disables all of that* channel's interrupts and writing an $FF to an IER enables all of that channel's interrupts.

7	6	5	4	3	2	1	0
SET BITS	TDRE IE	$\overline{\text{CTS}}$T IE	$\overline{\text{DCD}}$T IE	$\overline{\text{DSR}}$T IE	PAR IE	F/O/B IE	RDRF IE

Address = 0,4 Reset Value = - 0000000

Bit 7 **Enable/Disable**
1 Enable selected IRQ source
0 Disable selected IRQ source

Bits 0-6
1 Select for enable/disable
0 No change

CONTROL STATUS REGISTERS (CSR1, CSR2)

The Control Status Registers are read-only registers that provide I/O status and error condition information. A CSR is normally read after an $\overline{\text{IRQ}}$ has occurred to determine the exact cause of the interrupt condition.

7	6	5	4	3	2	1	0
FE	TUR	$\overline{\text{CTS}}$ LVL	$\overline{\text{DCD}}$ LVL	$\overline{\text{DSR}}$ LVL	BRK	$\overline{\text{DTR}}$ LVL	$\overline{\text{RTS}}$ LVL

Address = 1,5 Reset Value = 1 - - - - 011

Bit 7 **Framing Error (FE)**
1 A framing error occurred in receive data
0 No framing error occurred, or the RDR was read

Bit 6 **Transmitter Underrun (TUR)**
1 Transmit Shift Register is empty and TDRE is set
0 Transmitter Shift Register is not empty

Bit 5 **$\overline{\text{CTS}}$ Level ($\overline{\text{CTS}}$ LVL)**
1 $\overline{\text{CTS}}$ line is high
0 $\overline{\text{CTS}}$ line is low

Bit 4 **$\overline{\text{DCD}}$ Level ($\overline{\text{DCD}}$ LVL)**
1 $\overline{\text{DCD}}$ line is high
0 $\overline{\text{DCD}}$ line is low

Bit 3 **$\overline{\text{DSR}}$ Level ($\overline{\text{DSR}}$ LVL)**
1 $\overline{\text{DSR}}$ line is high
0 $\overline{\text{DSR}}$ line is low

Bit 2 **Receive Break (BRK)**
1 *A Receive Break has occurred*
0 No Receive Break occurred, or RDR was read

Bit 1 **$\overline{\text{DTR}}$ Level ($\overline{\text{DTR}}$ LVL)**
1 $\overline{\text{DTR}}$ line is high
0 $\overline{\text{DTR}}$ line is low

Bit 0 **$\overline{\text{RTS}}$ Level ($\overline{\text{RTS}}$ LVL)**
1 $\overline{\text{RTS}}$ line is high
0 $\overline{\text{RTS}}$ line is low

CONTROL REGISTERS (CR1, CR2)

The Control Registers are write-only registers. They control access to the Auxiliary Control Register and the Compare Data Register. They select the number of stop bits, control Echo Mode, and select the data rate.

(Accessed when Bit 7 = 0)

7	6	5	4	3	2	1	0
0	CDR/ACR	STOP BITS	ECHO	BAUD RATE SEL			

Address = 1,5 Reset Value = 0 - - - - - - -

Bit 7	Control or Format Register
0	Access Control Register

Bit 6	CDR/ACR
1	Access the Auxiliary Control Register (ACR)
0	Access the Compare Data Register (CDR)

Bit 5	Number of Stop Bits Per Character
1	Two stop bits
0	One stop bit

Bit 4	Echo Mode Selection
1	Echo Mode enabled
0	Echo Mode disabled

Bits 3-0				Baud Rate Selection
3	2	1	0	(bits per second with 3.6864 MHz crystal)
0	0	0	0	50
0	0	0	1	109.2
0	0	1	0	134.58
0	0	1	1	150
0	1	0	0	300
0	1	0	1	600
0	1	1	0	1200
0	1	1	1	1800
1	0	0	0	2400
1	0	0	1	3600
1	0	1	0	4800
1	0	1	1	7200
1	1	0	0	9600
1	1	0	1	19200
1	1	1	0	38400
1	1	1	1	External TxC and RxC X16 Clocks

FORMAT REGISTERS (FR1, FR2)

The Format Registers are write-only registers. They select the number of data bits per character and parity generation/checking options. They also control $\overline{\text{RTS}}$ and $\overline{\text{DTR}}$.

(Accessed when Bit 7 = 1)

7	6	5	4	3	2	1	0
1	DATA BITS		PAR SEL		PAR EN	$\overline{\text{DTR}}$ CNTL	$\overline{\text{RTS}}$ CNTL

Address = 1,5 Reset Value = 1 - - - - - - -

Bit 7	Control or Format Register
1	Access Format Register

Bits 6-5		Number of Data Bits Per Character
6	5	
0	0	5
0	1	6
1	0	7
1	1	8

Bits 4-3		Parity Mode Selection
4	3	
0	0	Odd Parity
0	1	Even Parity
1	0	Mark in Parity bit
1	1	Space in Parity bit

Bit 2	Parity Enable
1	Parity as specified by bits 4-3
0	No Parity

Bit 1	$\overline{\text{DTR}}$ Control
1	Set $\overline{\text{DTR}}$ high
0	Set $\overline{\text{DTR}}$ low

Bit 0	$\overline{\text{RTS}}$ Control
1	Set $\overline{\text{RTS}}$ high
0	Set $\overline{\text{RTS}}$ low

COMPARE DATA REGISTERS (CDR1, CDR2)

The Compare Data Registers are write-only registers which can be accessed when CR bit 6 = 0. By writing a value into the CDR, the DACIA is put in the compare mode. In this mode, setting of the RDRF bit is inhibited until a character is received which matches the value in the CDR. The next character is then received and the RDRF bit is set. The receiver will now operate normally until the CDR is again loaded.

(Control Register bit 6 = 0)

7	6	5	4	3	2	1	0
COMPARE DATA							

Address = 2,6 Reset Value = - - - - - - - -

AUXILIARY CONTROL REGISTERS (ACR1, ACR2)

The Auxiliary Control Registers are write-only registers. Bits 7-2 are unused. Bit 1 causes the transmitter to transmit a BREAK. Bit 0 determines whether parity error or the parity bit is displayed in ISR bit 2.

(Control Register bit 6 = 1)

7	6	5	4	3	2	1	0
NOT USED						TRNS BRK	PAR ERR/ST

Address = 2,6 Reset Value = - - - - - - 00

Bits 7-2	**Not Used**
Bit 1	**Transmit Break (TRNS BRK)**
1	Transmit continuous Break
0	Normal transmission
Bit 0	**Parity Error/State (PAR ERR/ST)**
1	Send value of parity bit to ISR bit 2 (Address Recognition mode)
0	Send Parity Error status to ISR bit 2

RECEIVE DATA REGISTERS (RDR1, RDR2)

The Receive Data Registers are read-only registers which are loaded with the received data character of each frame. Start bits, stop bits and parity bits are stripped off of incoming frames before the data is transferred from the Receive Shift Register to the Receive Data Register. For characters of less than eight bits, the unused bits are the high order bits which are set to 0.

MSB **LSB**

7	6	5	4	3	2	1	0
RECEIVE DATA							

Address = 3,7 Reset Value = 00000000

TRANSMIT DATA REGISTERS (TDR1, TDR2)

The Transmit Data Registers are write-only registers which are loaded from the CPU with data to be transmitted. For data characters of less than eight bits, the unused bits are the high order bits which are "don't care".

MSB **LSB**

7	6	5	4	3	2	1	0
TRANSMIT DATA							

Address = 3,7 Reset Value = - - - - - - - -

OPERATION

TERMINATION OF UNUSED INPUTS

Noise on floating inputs can affect chip operation. All unused inputs must be terminated. If the baud rate generator is bypassed, XTALI must be connected to ground (XTALO is an output and must be left open). If the external clock mode is not used, RxC and TxC may be tied either to +5V or to ground. If the handshake inputs are not needed, the $\overline{CTS}$ inputs should be tied low to enable the transmitters. The $\overline{DCD}$ and $\overline{DSR}$ inputs may either be tied high or low.

RESET INITIALIZATION

During power on initialization, all readable registers should be read to assure that the status registers are initialized. Specifically, the RDRF bit of the Interrupt Status Registers is not initialized by reset. The Receiver Data Registers must be read to clear this bit.

TDRE $\overline{IRQ}$ is generated only on the transition of the corresponding TDR from full to empty. Initialization software must account for this occurrence.

BAUD RATE CLOCK OPTIONS

The receiver and transmitter clocks may be supplied either by the internal Baud Rate Generator or by user supplied external clocks. Both channels may use the same clock source or one may use the Baud Rate Generator and the other channel external clocks. If both channels use the Baud Rate Generator, each channel may have a different bit rate. The options are shown in Figure 5.

An internal clock oscillator supplies the time base for the Baud Rate Generator. The oscillator can be driven by a crystal or an external clock.

If the on-chip oscillator is driven by a crystal, a parallel resonant crystal is connected between the XTALI and XTALO pins. The equivalent oscillator circuit is shown in Figure 6.

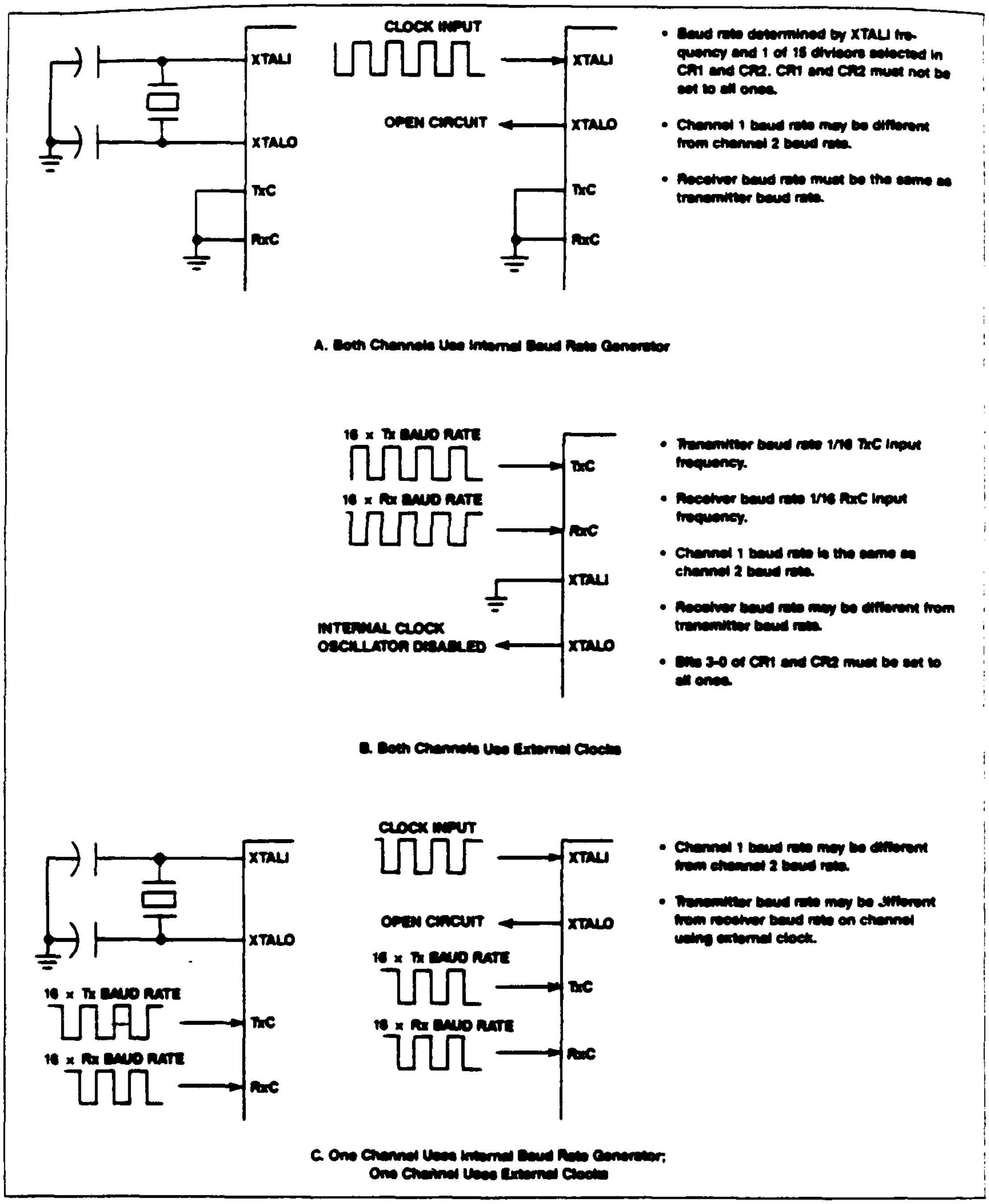
A parallel resonant crystal is specified by its load capacitance and series resonant resistance. For proper oscillator operation, the load capacitance (C_L), series resistance (R_s) and the crystal resonant frequency (F) must meet the following two relations:

$$(C + 2) = 2C_L \quad \text{or} \quad C = 2C_L - 2$$

$$R_s \leq R_{smax} = \frac{2 \times 10^6}{(FC_L)^2}$$

where: F is in MHz; C and C_L are in pF; R is in ohms.

To select a parallel resonant crystal for the oscillator, first select the load capacitance from a crystal manufacturer's catalog. Next, calculate R_{smax} based on F and C_L. The selected crystal must have a R_s less than the R_{smax}.



- Baud rate determined by XTALI frequency and 1 of 15 divisors selected in CR1 and CR2. CR1 and CR2 must not be set to all ones.
- Channel 1 baud rate may be different from channel 2 baud rate.
- Receiver baud rate must be the same as transmitter baud rate.

A. Both Channels Use Internal Baud Rate Generator

- Transmitter baud rate 1/16 TxC input frequency.
- Receiver baud rate 1/16 RxC input frequency.
- Channel 1 baud rate is the same as channel 2 baud rate.
- Receiver baud rate may be different from transmitter baud rate.
- Bits 3-0 of CR1 and CR2 must be set to all ones.

B. Both Channels Use External Clocks

- Channel 1 baud rate may be different from channel 2 baud rate.
- Transmitter baud rate may be different from receiver baud rate on channel using external clock.

C. One Channel Uses Internal Baud Rate Generator;
One Channel Uses External Clocks

Figure 5. Baud Rate Clock Options

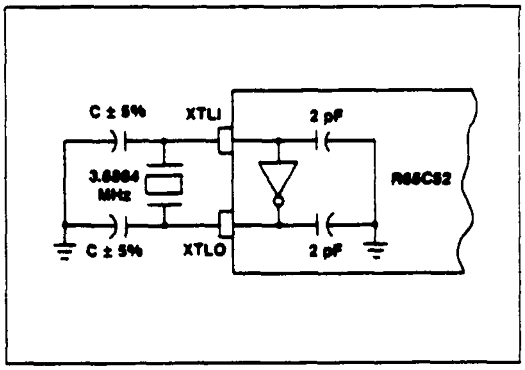


Figure 6.

For example, if $C_L = 22$ pF for a 3.6864 MHz parallel resonant crystal, then

$$C = (2 \times 22) - 2 = 42 \text{ pF (use standard value of 43 pF)}$$

The series resistance of the crystal must be less than

$$R_{smax} = \frac{2 \times 10^6}{(3.6864 \times 22)^2} = 304 \text{ ohms}$$

If the on-chip oscillator is driven by an external clock, the clock is input at XTALI and XTALO is left open.

An internal counter/divider circuit divides the frequency input at XTALI by the divisor selected in bits 3 through 0 of the Control Registers. Table 5 lists the divisors that may be selected and shows the bit rates generated with a 3.6864 MHz crystal or clock input. Other bit rates may be generated by changing the clock or crystal frequency. However, the input frequency must not exceed 4 MHz.

For external clock operation, a transmitter times 16 clock must be supplied at TxC and a receiver times 16 clock must be input at RxC. Since there are separate receiver and transmitter clock inputs, the receiver data rate may be different from the transmitter data rate.

Table 5. Baud Rate Generator Divisor Selection

Control Register Bits				Divisor Selected For The Internal Counter	Baud Rate Generated With 3.6864 MHz Crystal or Clock	Baud Rate Generated* With a Crystal or Clock of Frequency (f)
3	2	1	0			
0	0	0	0	73,728	$(3.6864 \times 10^6)/73,728 = 50$	f/73,728
0	0	0	1	33,536	$(3.6864 \times 10^6)/33,536 = 109.92$	f/33,536
0	0	1	0	27,408	$(3.6864 \times 10^6)/27,408 = 134.58$	f/27,408
0	0	1	1	24,576	$(3.6864 \times 10^6)/24,576 = 150$	f/24,576
0	1	0	0	12,288	$(3.6864 \times 10^6)/12,288 = 300$	f/12,288
0	1	0	1	6,144	$(3.6864 \times 10^6)/6,144 = 600$	f/6,144
0	1	1	0	3,072	$(3.6864 \times 10^6)/3,072 = 1,200$	f/3,072
0	1	1	1	2,048	$(3.6864 \times 10^6)/2,048 = 1,800$	f/2,048
1	0	0	0	1,536	$(3.6864 \times 10^6)/1,536 = 2,400$	f/1,536
1	0	0	1	1,024	$(3.6864 \times 10^6)/1,024 = 3,600$	f/1,024
1	0	1	0	768	$(3.6864 \times 10^6)/768 = 4,800$	f/768
1	0	1	1	512	$(3.6864 \times 10^6)/512 = 7,200$	f/512
1	1	0	0	384	$(3.6864 \times 10^6)/384 = 9,600$	f/384
1	1	0	1	192	$(3.6864 \times 10^6)/192 = 19,200$	f/192
1	1	1	0	96	$(3.6864 \times 10^6)/96 = 38,400$	f/96
1	1	1	1	16	*Transmitter Baud Rate = TxC/16*	*Receiver Baud Rate = RxC/16*

$$\text{*Baud Rate} = \frac{\text{Frequency}}{\text{Divisor}}$$

6

CONTINUOUS DATA TRANSMIT

In the normal operating mode, the TDRE bit in the ISR signals the MPU that the DACIA is ready to accept the next data word. An $\overline{IRQ}$ occurs on the transition of the TDR from full to empty if the corresponding TDRE $\overline{IRQ}$ enable bit is set in the IER. The TDRE bit is set at the beginning of the start bit. When the MPU writes a word to the TDR the TDRE bit is cleared. In order to maintain continuous transmission the TDR must be loaded before the stop bit(s are ended. 1/16 of a bit time after $\overline{IRQ}$ goes low, the $\overline{IRQ}$ line may be reset by reading the ISR. $\overline{IRQ}$ will always reset when data is written to the TDR. Figure 7 shows the relationship between $\overline{IRQ}$ and TxD for the Continuous Data Transmit mode.

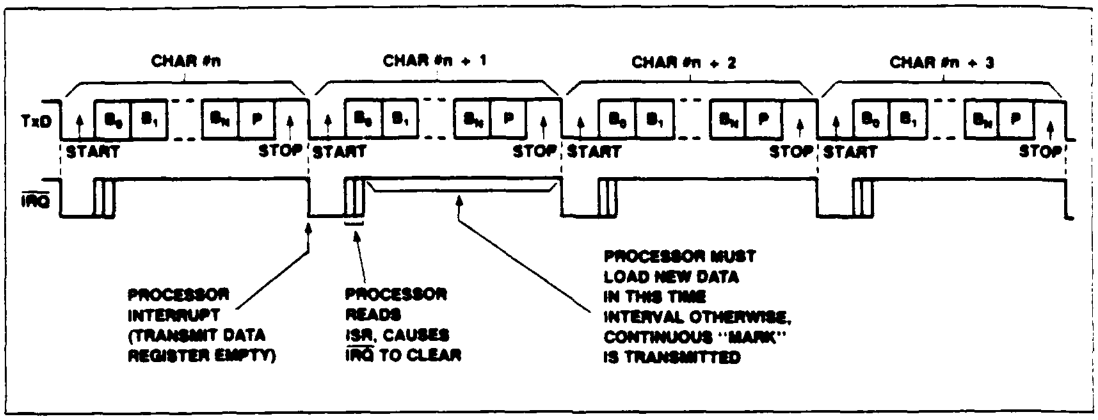

CHAR #n
CHAR #n + 1
CHAR #n + 2
CHAR #n + 3
TxD
B0
B1
BN
P
START
STOP
IRQ
PROCESSOR INTERRUPT (TRANSMIT DATA REGISTER EMPTY)
PROCESSOR READS ISR, CAUSES IRQ TO CLEAR
PROCESSOR MUST LOAD NEW DATA IN THIS TIME INTERVAL OTHERWISE, CONTINUOUS "MARK" IS TRANSMITTED

Figure 7. Continuous Data Transmit

TRANSMIT UNDERRUN CONDITION

If the MPU is unable to load the TDR before the last stop bit is sent, the TxD line goes to the MARK condition and the underrun flag (TUR) is set. This condition persists until the TDR is loaded with a new word. Figure 8 shows the relation between $\overline{IRQ}$ and TxD for the Transmit Underrun Condition.

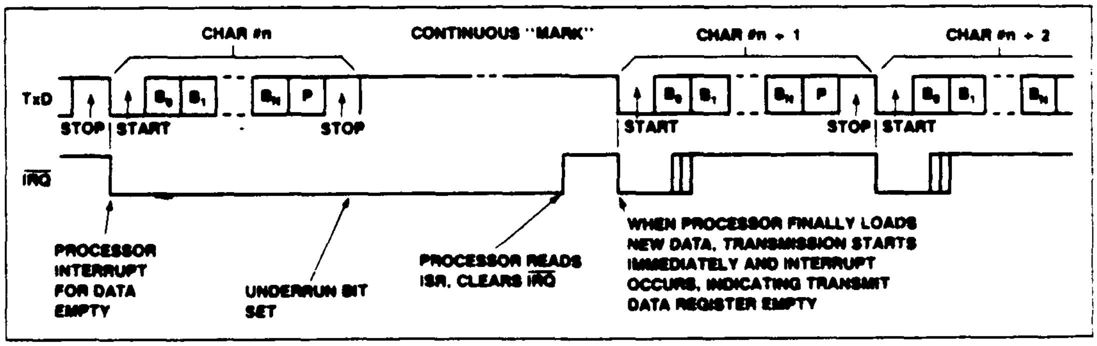
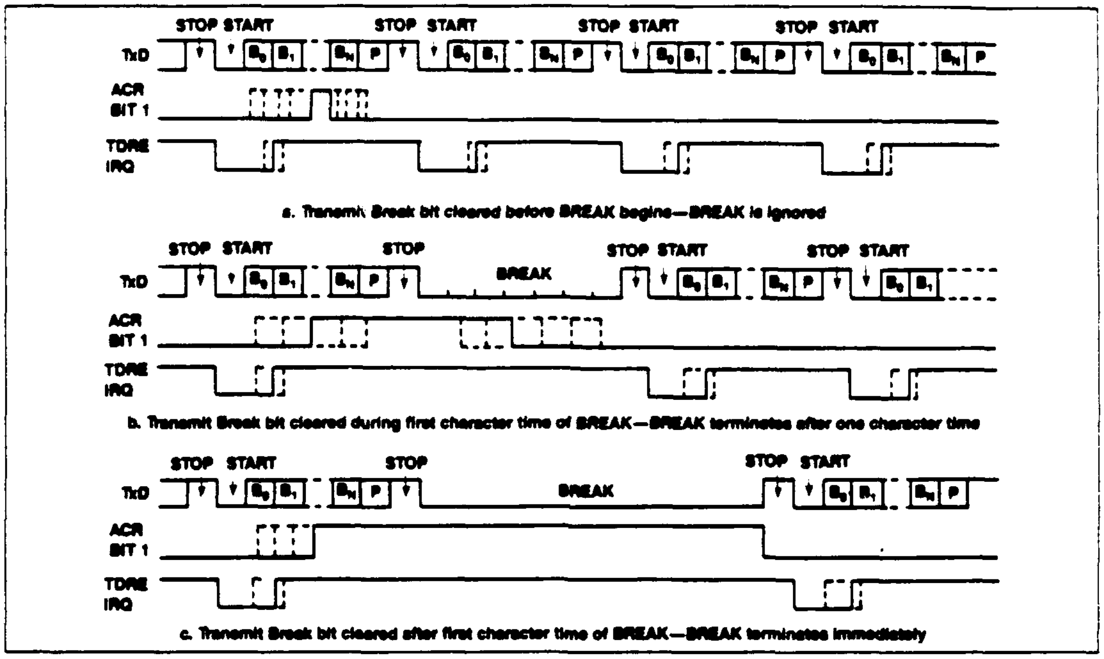

CHAR #n
CONTINUOUS "MARK"
CHAR #n + 1
CHAR #n + 2
TxD
B0
B1
BN
P
STOP
START
IRQ
PROCESSOR INTERRUPT FOR DATA EMPTY
UNDERRUN BIT SET
PROCESSOR READS ISR, CLEARS IRQ
WHEN PROCESSOR FINALLY LOADS NEW DATA, TRANSMISSION STARTS IMMEDIATELY AND INTERRUPT OCCURS, INDICATING TRANSMIT DATA REGISTER EMPTY

Figure 8. Transmit Underrun Condition Relationship

TRANSMIT BREAK CHARACTER

A BREAK may be transmitted by setting bit 1 of the ACR (Transmit Break bit) to a 1. The BREAK is transmitted after the character in the Transmit Shift Register is sent. If there is a character in the Transmit Data Register, it will be transmitted after the BREAK is terminated. The Transmit Break bit must remain set for at least one character time to assure that a proper BREAK is transmitted. If the Transmit Break bit is cleared before one character time of BREAK has been transmitted, the BREAK will be terminated after one character time has elapsed. If the Transmit Break bit is cleared after one character time of BREAK has been transmitted, the BREAK will be terminated immediately. Figure 9 shows the relationship of TxD, $\overline{\text{IRQ}}$ and ACR bit 1 for various BREAK options.



a. Transmit Break bit cleared before BREAK begins—BREAK is ignored

b. Transmit Break bit cleared during first character time of BREAK—BREAK terminates after one character time

c. Transmit Break bit cleared after first character time of BREAK—BREAK terminates immediately

Figure 9. Transmit BREAK

EFFECTS OF $\overline{\text{CTS}}$ ON TRANSMITTER

The $\overline{\text{CTS}}$ control line controls the transmission of data or the handshaking of data to a "busy" device (such as a printer). When the $\overline{\text{CTS}}$ line is low, the transmitter operates normally. A high condition inhibits the TDRE bit in the ISR from becoming set. Transmission of the word currently in the shift register is completed but any word in the TDR is held until $\overline{\text{CTS}}$ goes low.

Any transition on $\overline{\text{CTS}}$ sets bit 5 ($\overline{\text{CTST}}$) of the ISR A high on $\overline{\text{CTS}}$ forces bit 6 (TDRE) of the ISR to a 0. Bit 7 of the ISR also goes to a 1 when $\overline{\text{CTS}}$ is high, if Echo Mode is disabled. Thus, when the ISR is $80, it means that $\overline{\text{CTS}}$ is high and no interrupt source requires service. A processor interrupt will not be generated under these circumstances, but an ISR polling routine should accommodate this.

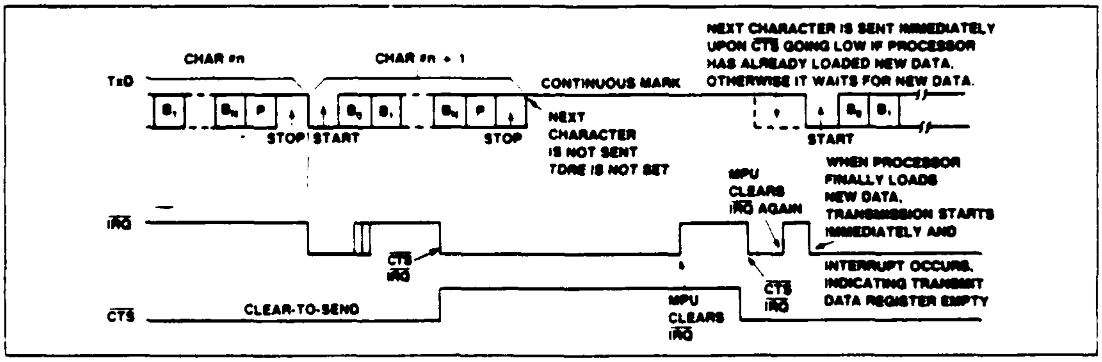

TxD
CHAR #n
CHAR #n + 1
B_7
B_N
P
STOP
START
B_0
B_1
CONTINUOUS MARK
NEXT CHARACTER IS NOT SENT TDRE IS NOT SET
NEXT CHARACTER IS SENT IMMEDIATELY UPON $\overline{\text{CTS}}$ GOING LOW IF PROCESSOR HAS ALREADY LOADED NEW DATA. OTHERWISE IT WAITS FOR NEW DATA.
$\overline{\text{IRQ}}$
$\overline{\text{CTS}}$ $\overline{\text{IRQ}}$
MPU CLEARS $\overline{\text{IRQ}}$ AGAIN
WHEN PROCESSOR FINALLY LOADS NEW DATA, TRANSMISSION STARTS IMMEDIATELY AND
INTERRUPT OCCURS, INDICATING TRANSMIT DATA REGISTER EMPTY
$\overline{\text{CTS}}$
CLEAR-TO-SEND
MPU CLEARS $\overline{\text{IRQ}}$

Figure 10. Effects of $\overline{\text{CTS}}$ on Transmitter

ECHO MODE TIMING

In the Echo Mode, the TxD line re-transmits the data received on the RxD line, delayed by 1/2 of a bit time. An internal underrun mode must occur before Echo Mode will start transmitting. In normal transmit mode if TDRE occurs (indicating end of data) an underflow flag would be set and continuous Mark transmitted. If Echo is initiated, the underflow flag will not be set at end of data and continuous Mark will not be transmitted. Figure 11 shows the relationship of RxD and TxD for Echo Mode.

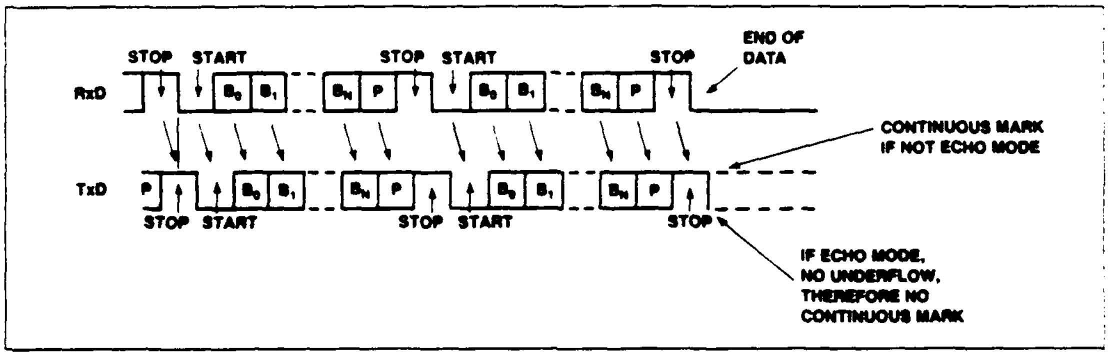

STOP
START
STOP
START
STOP
END OF DATA
RxD
B_0
B_1
B_N
P
CONTINUOUS MARK IF NOT ECHO MODE
TxD
STOP
START
STOP
START
STOP
IF ECHO MODE, NO UNDERFLOW, THEREFORE NO CONTINUOUS MARK

Figure 11. Echo Mode Timing

CONTINUOUS DATA RECEIVE

The normal receive mode sets the RDRF bit in the ISR when the DACIA channel has received a full data word. This occurs at about the 9/16 point through the stop bit. The processor must read the RDR before the next stop bit, or an overrun error occurs. Figure 12 shows the relationship between $\overline{IRQ}$ and RxD for the continuous Data Receive mode.

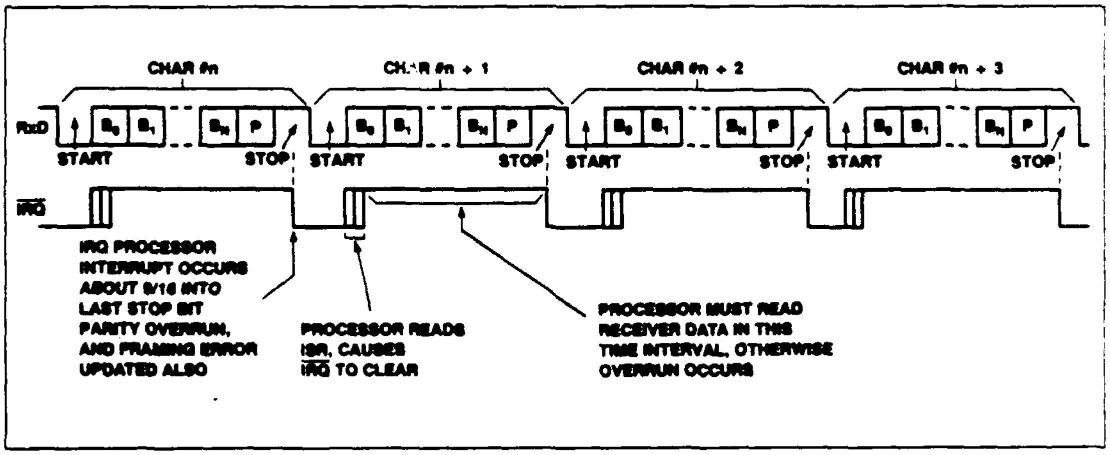

CHAR #n
CHAR #n + 1
CHAR #n + 2
CHAR #n + 3
RxD
B_0
B_1
B_N
P
START
STOP
$\overline{IRQ}$
IRQ PROCESSOR INTERRUPT OCCURS ABOUT 9/16 INTO LAST STOP BIT PARITY OVERRUN, AND FRAMING ERROR UPDATED ALSO
PROCESSOR READS ISR, CAUSES $\overline{IRQ}$ TO CLEAR
PROCESSOR MUST READ RECEIVER DATA IN THIS TIME INTERVAL, OTHERWISE OVERRUN OCCURS

Figure 12. Continuous Data Receive

EFFECTS OF OVERRUN ON RECEIVER

If the processor does not read the RDR before the stop bit of the next word, an overrun error occurs, the overrun bit is set in the ISR, and the new data word is not transferred to the RDR. The RDR contains the last word not read by the MPU and all following data is lost. The receiver will return to normal operation when the RDR is read. Figure 13 shows the relationship of $\overline{IRQ}$ and RxD when overrun occurs.

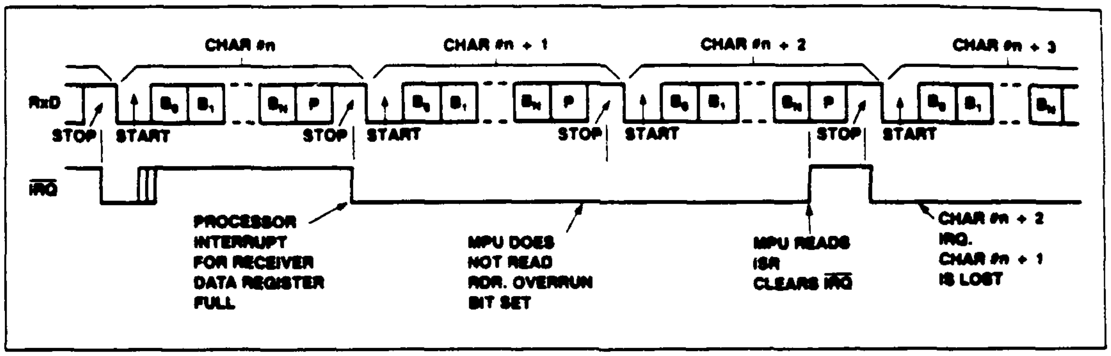


Figure 13. Effects of Overrun on Receiver

RECEIVE BREAK CHARACTER

When a Break character is received, the Break bit is set. The receiver does not set the RDRF bit and remains in this state until a stop bit is received. *At this time the next character is received* normally. Figure 14 shows the relationship of $\overline{IRQ}$ and RxD for a Receive Break Character.

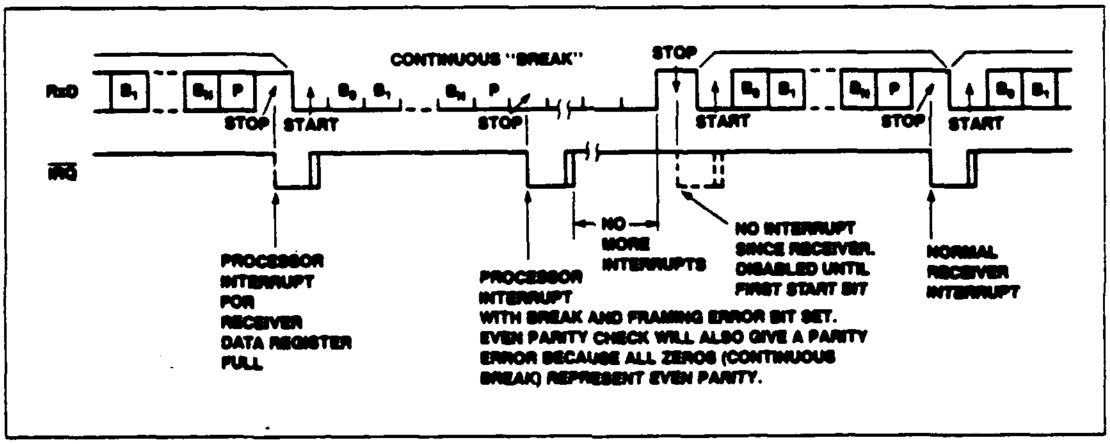
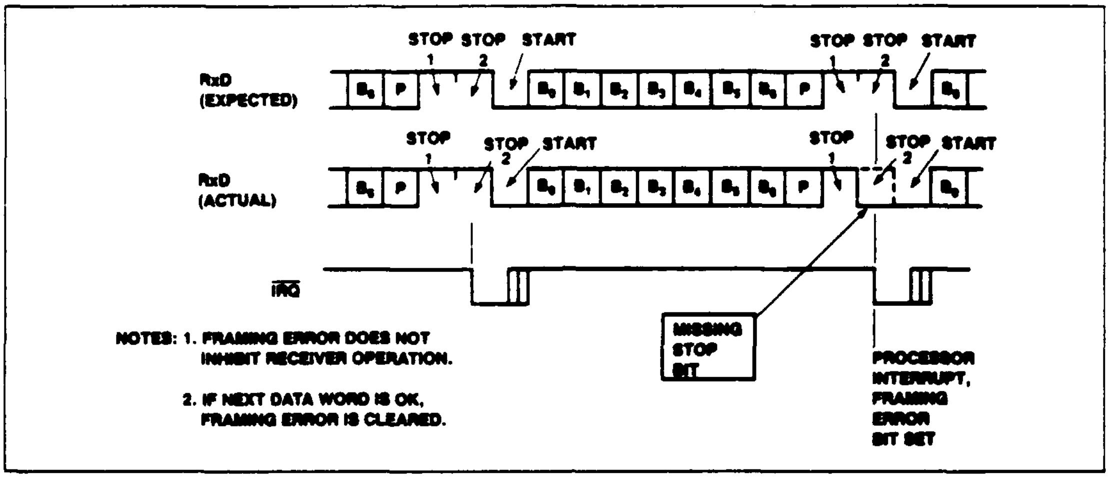

CONTINUOUS "BREAK"
RxD
B_0
B_1
B_N
P
STOP
START
IRQ
PROCESSOR INTERRUPT FOR RECEIVER DATA REGISTER FULL
PROCESSOR INTERRUPT WITH BREAK AND FRAMING ERROR BIT SET. EVEN PARITY CHECK WILL ALSO GIVE A PARITY ERROR BECAUSE ALL ZEROS (CONTINUOUS BREAK) REPRESENT EVEN PARITY.
NO MORE INTERRUPTS
NO INTERRUPT SINCE RECEIVER. DISABLED UNTIL FIRST START BIT
NORMAL RECEIVER INTERRUPT

Figure 14. Receive Break Character

FRAMING ERROR

Framing error is caused by the absence of stop bit(s) on received data. The framing error bit is set when the RDRF bit is set. Subsequent data words are tested separately, so the status bit always reflects the last data word received. Figure 15 shows the relationship of $\overline{IRQ}$ and RxD when a framing error occurs.



NOTES: 1. FRAMING ERROR DOES NOT INHIBIT RECEIVER OPERATION.

2. IF NEXT DATA WORD IS OK, FRAMING ERROR IS CLEARED.

Figure 15. Framing Error

PARITY ERROR DETECT/ADDRESS FRAME RECOGNITION

The Parity Status bit (ISR bit 2) may be programmed to indicate parity errors (ACR bit 0 = 0) or to display the parity bit received (ACR bit 0 = 1).

In applications where parity checking is used, one of the parity checking modes is enabled by setting bits 2, 3 and 4 of the Format Register to the desired option and bit 0 of the Auxiliary Control Register is reset to 0. Then, when the RDRF bit (bit 0) is set in the ISR, the PAR bit (bit 2) will be set when a parity error is detected.

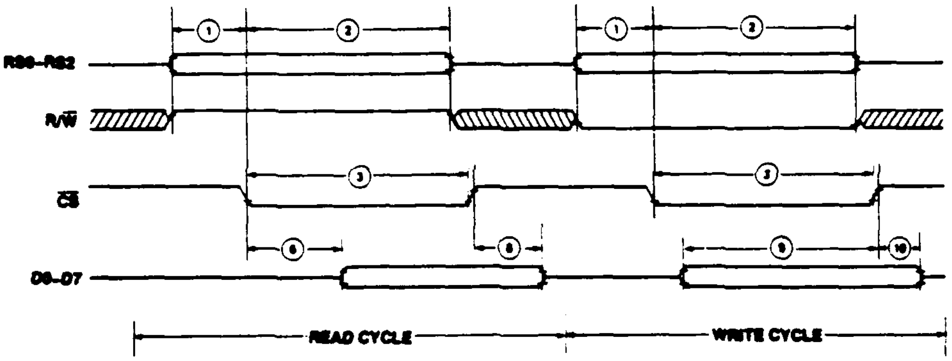
In multi-drop applications, the parity bit is used as an address/data flag. It is set to 1 for address frames and is 0 on data frames. For this type of operation, bit 0 of the ACR is set to a 1 and bits 2, 3 and 4 of the FR select a parity checking mode. Then, ISR bit 2 will be set to a 1 by incoming address frames and it will be a 0 on data frames.

COMPARE MODE

The Compare Mode is automatically enabled, i.e., the channel is put to sleep, whenever data is written to the Compare Data Register. NOTE: Bit 6 of the Control Register must be set to 0 to enable access to the Compare Data Register. When the channel is in the compare mode, the RDRF bit (bit 0 of the ISR) is forced to a 0. Upon receipt of a matching character, normal receiver operation resumes and the RDRF bit (bit 0 of the ISR) will be set upon receipt of the *next* character.

SPECIFICATIONS

DACIA READ/WRITE WAVEFORMS



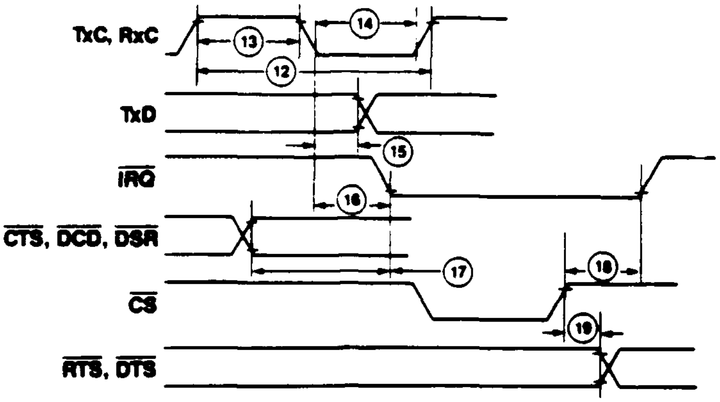
NOTE: TIMING MEASUREMENTS ARE REFERENCED TO AND FROM A LOW VOLTAGE OF 0.8V AND A HIGH VOLTAGE OF 2.0V, UNLESS OTHERWISE NOTED.

DACIA READ/WRITE CYCLE TIMING

(V_{CC} = 5 Vdc ±5%, V_{SS} = 0 Vdc, T_A = T_L to T_H, unless otherwise noted)

			1 MHz		2 MHz		3 MHz		
Number	Characteristic	Symbol	Min.	Max.	Min.	Max.	Min.	Max.	Unit
1	R/$\overline{W}$, RS0–RS2 Valid to $\overline{CS}$ Low (Setup)	T_{RSU}	5	—	5	—	5	—	ns
2	$\overline{CS}$ Low to R/$\overline{W}$, RS0–RS2 Invalid (Hold)	T_{RH}	45	—	45	—	45	—	ns
3	$\overline{CS}$ Pulse Width	T_{CP}	410	—	340	—	210	—	ns
6	$\overline{CS}$ Low to Data Valid (Read)	T_{CDV}	—	380	—	280	—	170	ns
8	$\overline{CS}$ High to Data Invalid (Read)	T_{CDR}	10	50	10	50	10	50	ns
9	Data Valid to $\overline{CS}$ High (Write, Setup)	T_{DSU}	30	—	30	—	30	—	ns
10	$\overline{CS}$ High to Data Invalid (Write Hold)	T_{CDW}	10	—	10	—	10	—	ns

DACIA TRANSMIT/RECEIVER WAVEFORMS


TxC, RxC
13
14
12
TxD
15
$\overline{IRQ}$
16
$\overline{CTS}$, $\overline{DCD}$, $\overline{DSR}$
17
18
$\overline{CS}$
19
$\overline{RTS}$, $\overline{DTS}$

NOTE: $\overline{IRQ}$ WILL NOT CHANGE STATE WHEN $\overline{CS}$ IS LOW.

NOTE: TIMING MEASUREMENTS ARE REFERENCED TO AND FROM A LOW VOLTAGE OF 0.8V AND A HIGH VOLTAGE OF 2.0V, UNLESS OTHERWISE NOTED.

TRANSMIT/RECEIVE AND INTERRUPT ACKNOWLEDGE TIMING

(V_{CC} = 5 Vdc ±5%, V_{SS} = 0 Vdc, T_A = T_L to T_H, unless otherwise noted)

Number	Characteristic	Symbol	Min.	Max.	Unit
TRANSMIT/RECEIVE TIMING					
12	Transmit/Receive Clock Rate	t_{CY}	300	—	ns
13	Transmit/Receive Clock High	t_{CH}	125	—	ns
14	Transmit/Receive Clock Low	t_{CL}	125	—	ns
15	TxC, RxC to TxD Propagation Delay	t_{DD}	—	285	ns
16	TxC, RxC to $\overline{IRQ}$ Propagation Delay	t_{DI}	—	285	ns
17	$\overline{CTS}$, $\overline{DCD}$, $\overline{DSR}$ Valid to $\overline{IRQ}$ Low	t_{CTI}	—	150	ns
18	$\overline{IRQ}$ Propagation Delay (Clear)	t_{IRQ}	—	150	ns
19	$\overline{RTS}$, $\overline{DTR}$ Propagation Delay	t_{DLY}	—	150	ns

ABSOLUTE MAXIMUM RATINGS*

Parameter	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	Vdc
Input Voltage	V_{IN}	−0.3 to V_{CC} +0.3	Vdc
Output Voltage	V_{OUT}	−0.3 to V_{CC} +0.3	Vdc
Operating Temperature Commercial Industrial	T_A	 0 to +70 −40 to +85	°C
Storage Temperature	T_{STG}	−55 to +150	°C

*NOTE: Stresses above those listed 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 indicated in other sections of this document is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

OPERATING CONDITIONS

Parameter	Symbol	Value
Supply Voltage	V_{CC}	5V ± 5%
Temperature Range Commercial Industrial	T_A	 0 to 70°C −40°C to +85°C

DC CHARACTERISTICS

(V_{CC} = 5.0 V ± 5%, V_{SS} = 0, T_A = T_L to T_H, unless otherwise noted)

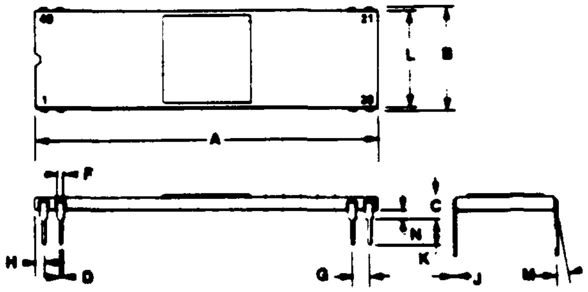
Parameter	Symbol	Min	Typ	Max	Unit	Test Conditions
Input High Voltage Except XTALI XTALI	V_{IH}	 +2.0 +3.0	 — —	 V_{CC} + 0.3 V_{CC} + 0.3	V	
Input Low Voltage Except XTALI XTALI	V_{IL}	 −0.3 −0.3	 — —	 +0.8 +0.4	V	
Input Leakage Current R/$\overline{W}$, $\overline{RES}$, RS0, RS1, RS2, RxD, $\overline{CTS}$, $\overline{DCD}$, $\overline{DSR}$, RxC, TxC, $\overline{CS}$	I_{IN}	—	10	50	μA	V_{IN} = 0V to 5.0V V_{CC} = 5.25V
Input Leakage Current for Three-State Off D0–D7	I_{TSI}	—	±2	10	μA	V_{IN} = 0.4V to 2.4V V_{CC} = 5.25V
Output High Voltage D0–D7, TxD, CLK OUT, $\overline{RTS}$, $\overline{DTR}$	V_{OH}	+2.4	—	—	V	V_{CC} = 4.75V I_{LOAD} = −100 μA
Output Low Voltage D0–D7, TxD, CLK OUT, $\overline{RTS}$, $\overline{DTR}$	V_{OL}	—	—	+0.4	V	V_{CC} = 4.75V I_{LOAD} = 1.6 mA
Output Leakage Current (Off State) $\overline{IRQ}$	I_{OFF}	—	±2	±10	μA	V_{CC} = 5.25V V_{OUT} = 0 to 2.4V
Power Dissipation	P_D	—	—	10	mW/MHz	
Input Capacitance Except XTALI XTALI	C_{IN}	 — —	 — —	 5 10	 pF pF	V_{CC} = 5.0V V_{IN} = 0V f = 2 MHz T_A = 25°C
Output Capacitance	C_{OUT}	—		10	pF	

Notes:

1. All units are-direct current (dc) except for capacitance.
2. Negative sign indicates outward current flow, positive indicates inward flow.
3. Typical values are shown for V_{CC} = 5.0V and T_A = 25°C.

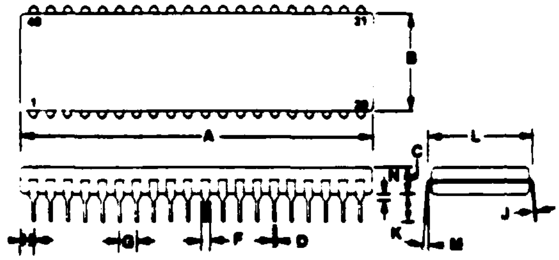
PACKAGE DIMENSIONS

40-PIN CERAMIC DIP



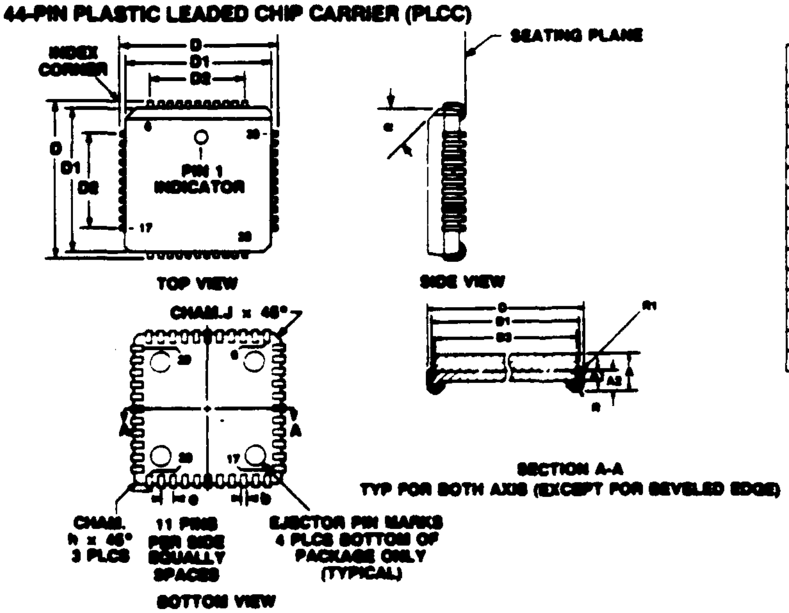
DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	50.29	51.31	1.980	2.020
B	15.11	15.88	0.595	0.625
C	2.54	4.19	0.100	0.165
D	0.38	0.53	0.015	0.021
F	0.76	1.27	0.030	0.050
G	2.54 BSC		0.100 BSC	
H	0.76	1.78	0.030	0.070
J	0.20	0.33	0.008	0.013
K	2.54	4.19	0.100	0.165
L	14.60	15.37	0.575	0.605
M	0°	10°	0°	10°
N	0.51	1.52	0.020	0.060

40-PIN PLASTIC DIP



DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	51.82	52.32	2.040	2.060
B	13.46	13.97	0.530	0.550
C	3.56	5.08	0.140	0.200
D	0.38	0.53	0.015	0.021
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.30	0.008	0.012
K	3.30	4.32	0.130	0.170
L	15.24 BSC		0.600 BSC	
M	7°	10°	7°	10°
N	0.51	1.02	0.020	0.040

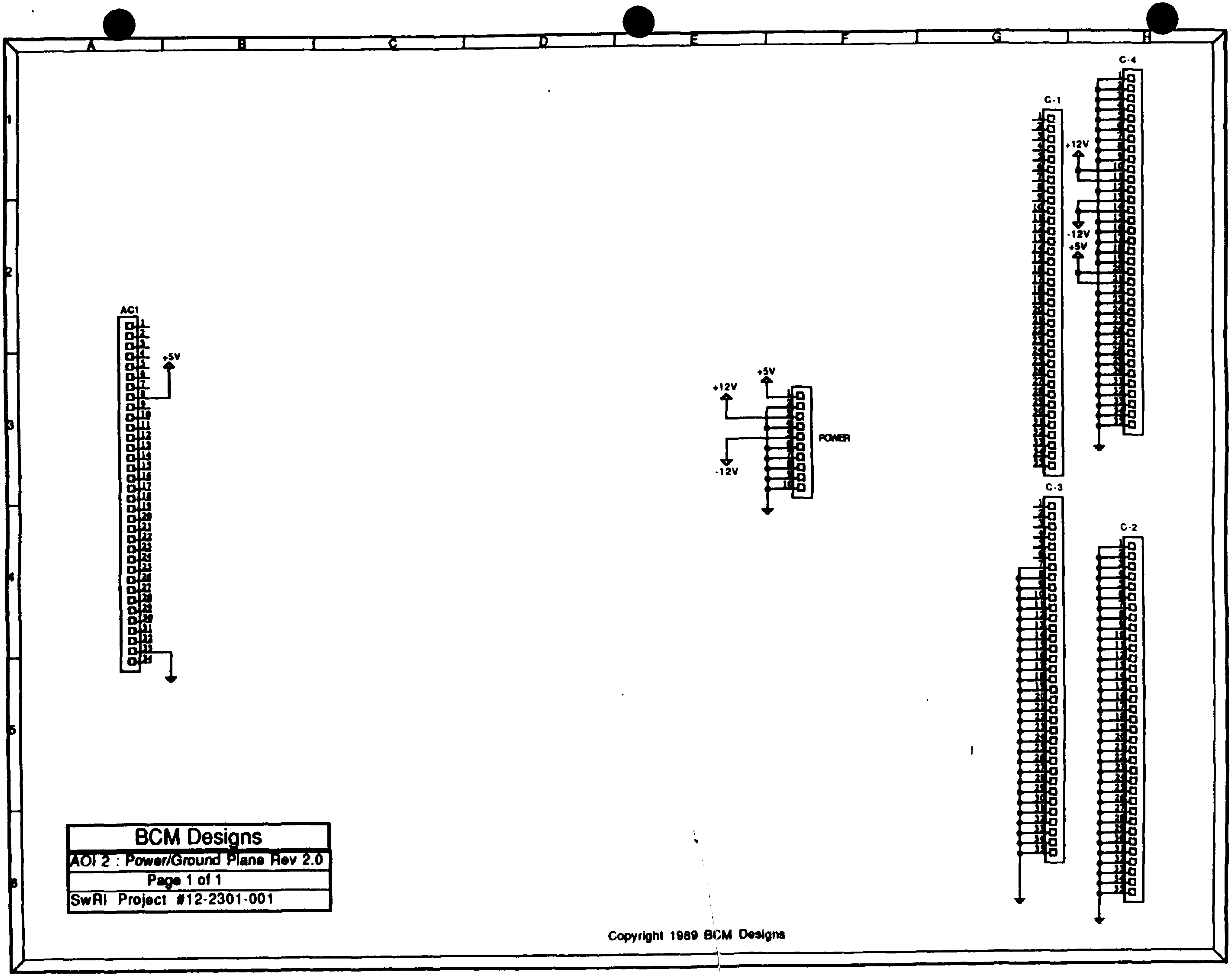
44-PIN PLASTIC LEADED CHIP CARRIER (PLCC)



DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	4.14	4.39	0.163	0.173
A1	1.37	1.47	0.054	0.058
A2	2.31	2.46	0.091	0.097
b	0.457 TYP		0.018 TYP	
D	17.40	17.65	0.685	0.695
D1	16.46	16.66	0.648	0.656
D2	12.62	12.78	0.497	0.503
D3	15.75 REF		0.620 REF	
e	1.27 BSC		0.050 BSC	
h	1.15 TYP		0.045 TYP	
J	0.25 TYP		0.010 TYP	
α	45° TYP		45° TYP	
R	0.89 TYP		0.035 TYP	
R1	0.25 TYP		0.010 TYP	

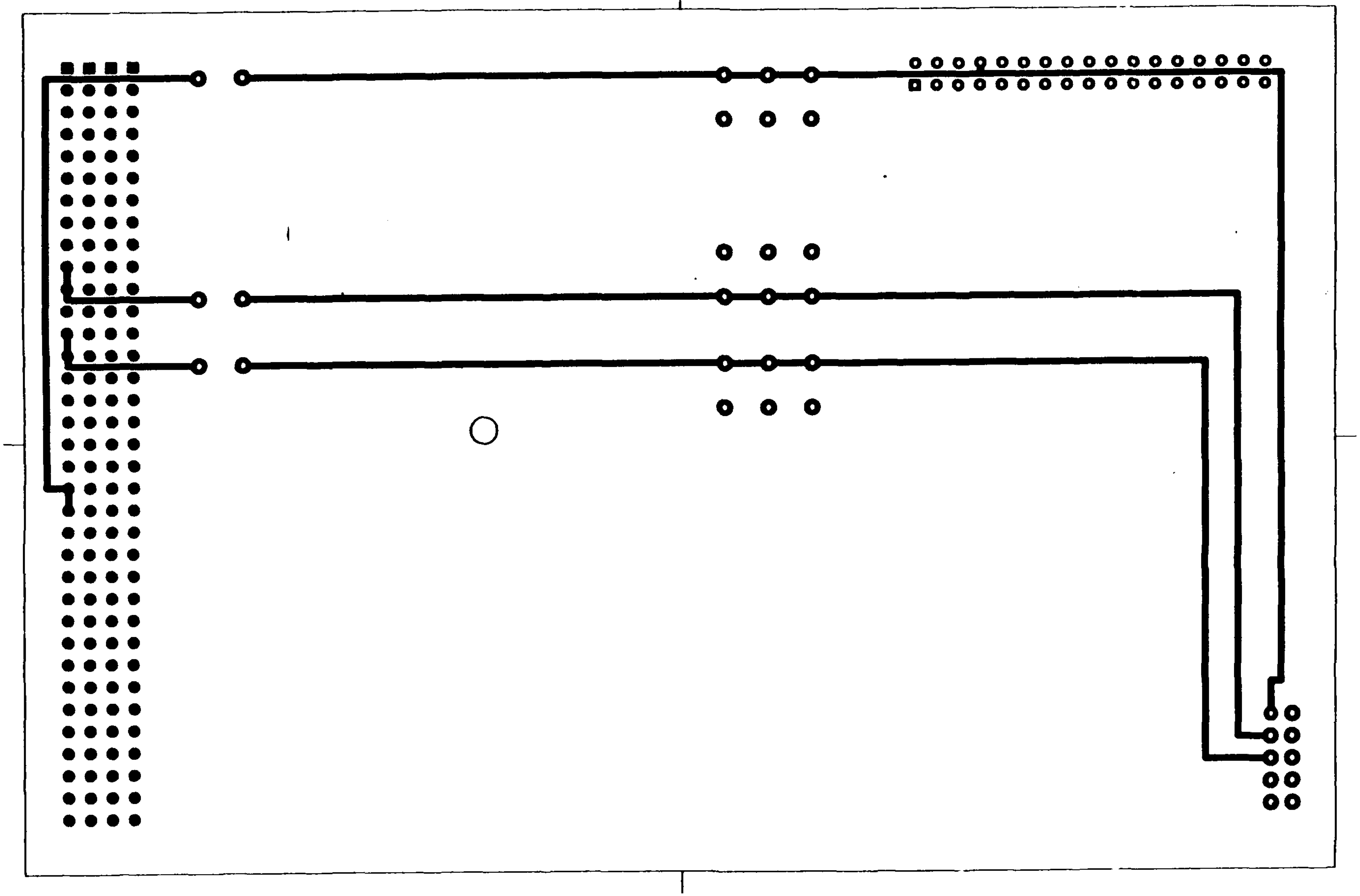
Appendix D - Ground plane/Power board

Appendix D - Schematic



BCM Designs
AOI 2 : Power/Ground Plane Rev 2.0
Page 1 of 1
SwRI Project #12-2301-001

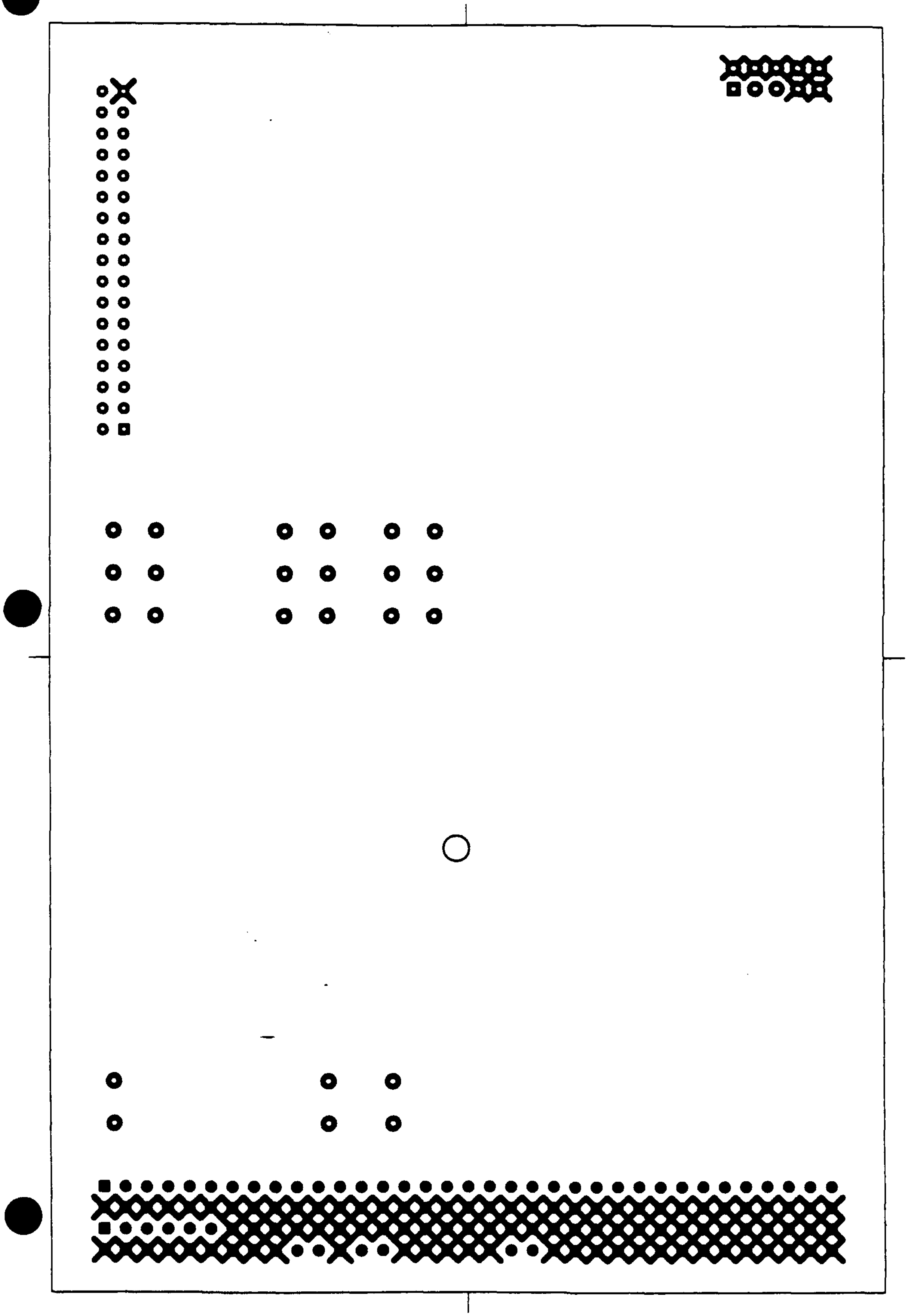
Appendix D - PCB layouts



Double Stack Final PCB

Tue, Sep 12, 1989 5:54 PM

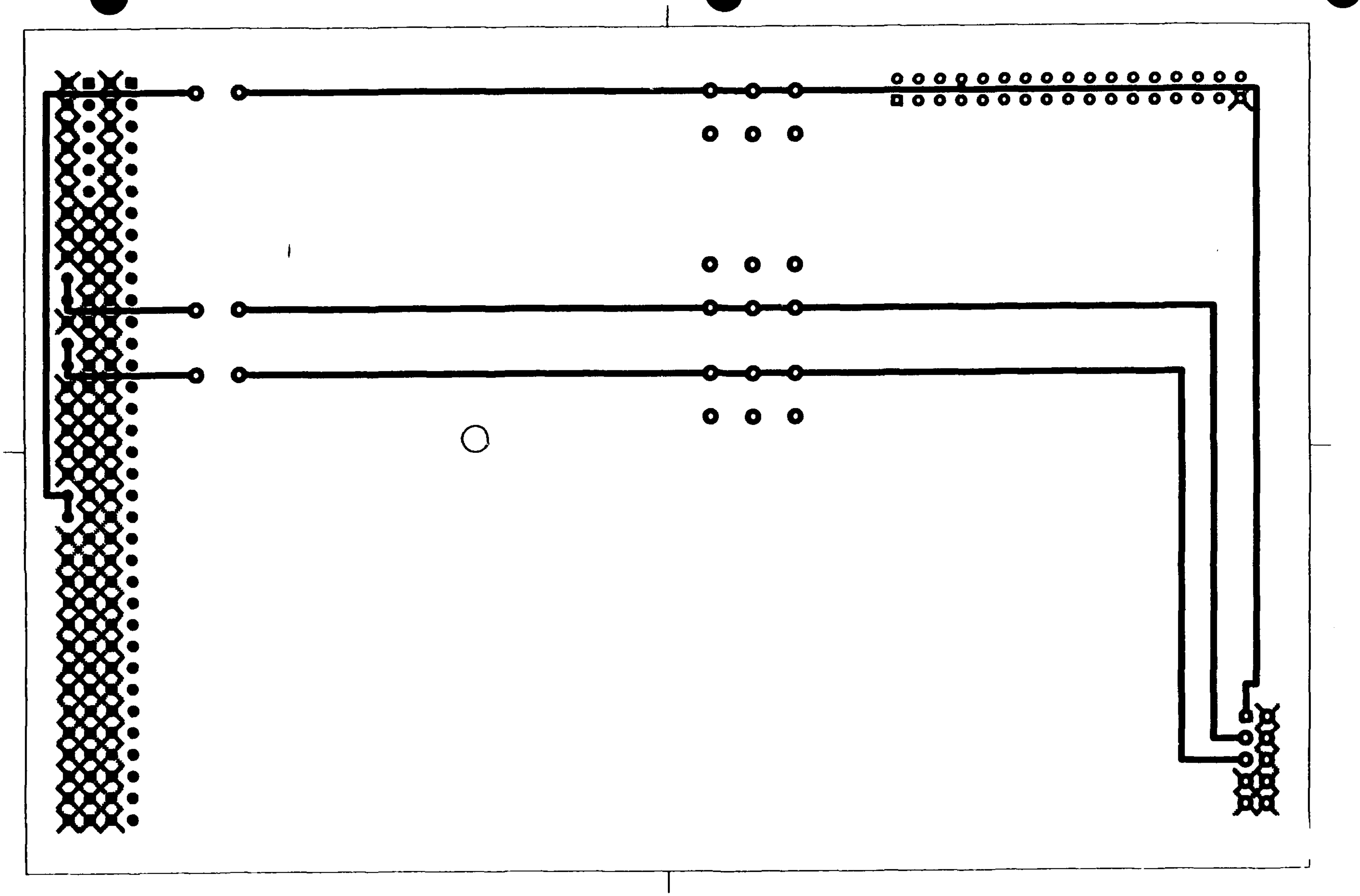
Component Side Scale150%



Double Stack Final PCB

Tue, Sep 12, 1989 5:54 PM

Solder Side Scale150%



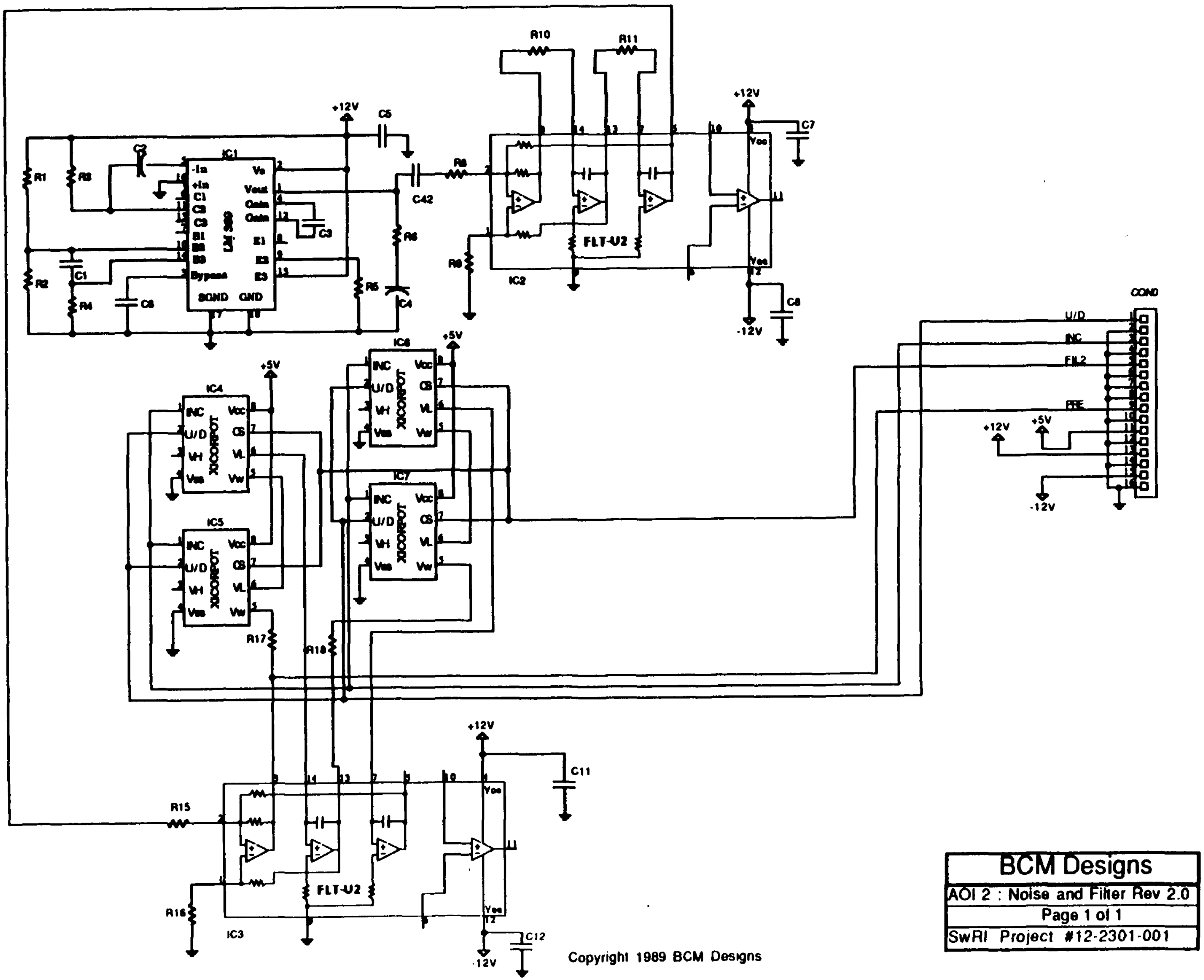
Double Stack Final PCB

Tue, Sep 12, 1989 5:54 PM

Component Xray Scale150%

Appendix E - Filter board

Appendix E - Schematic



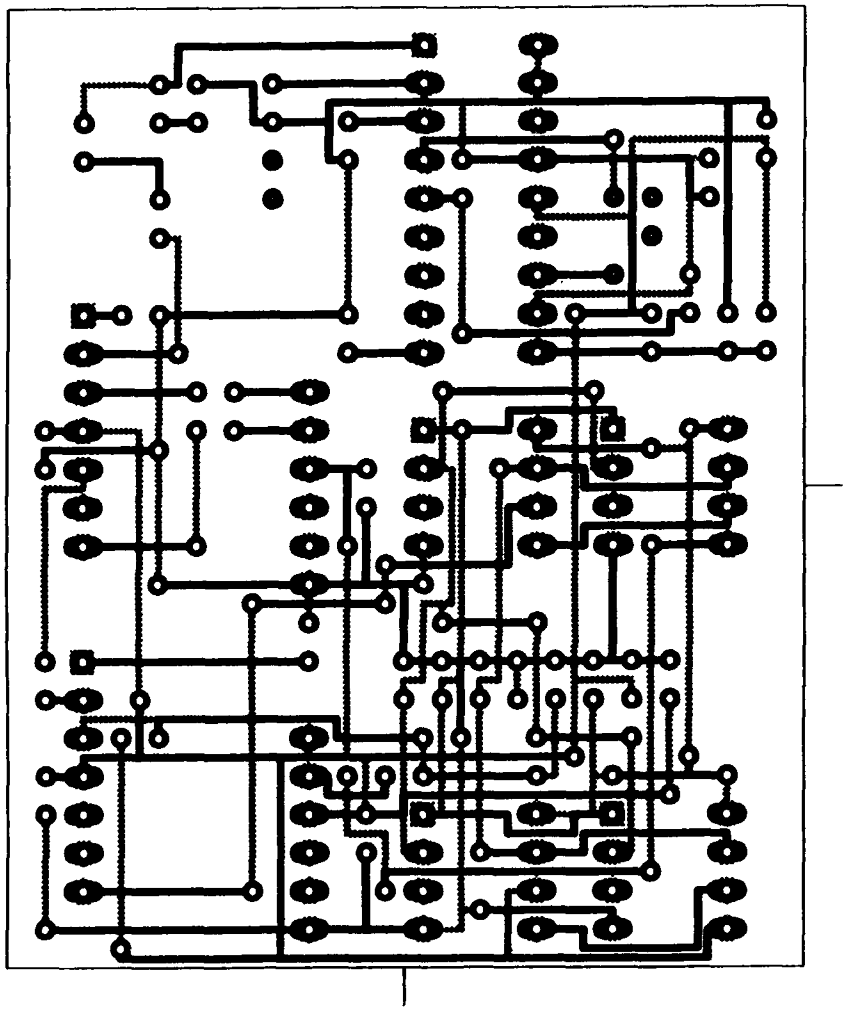
BCM Designs

AOI 2 : Noise and Filter Rev 2.0

Page 1 of 1

SwRI Project #12-2301-001

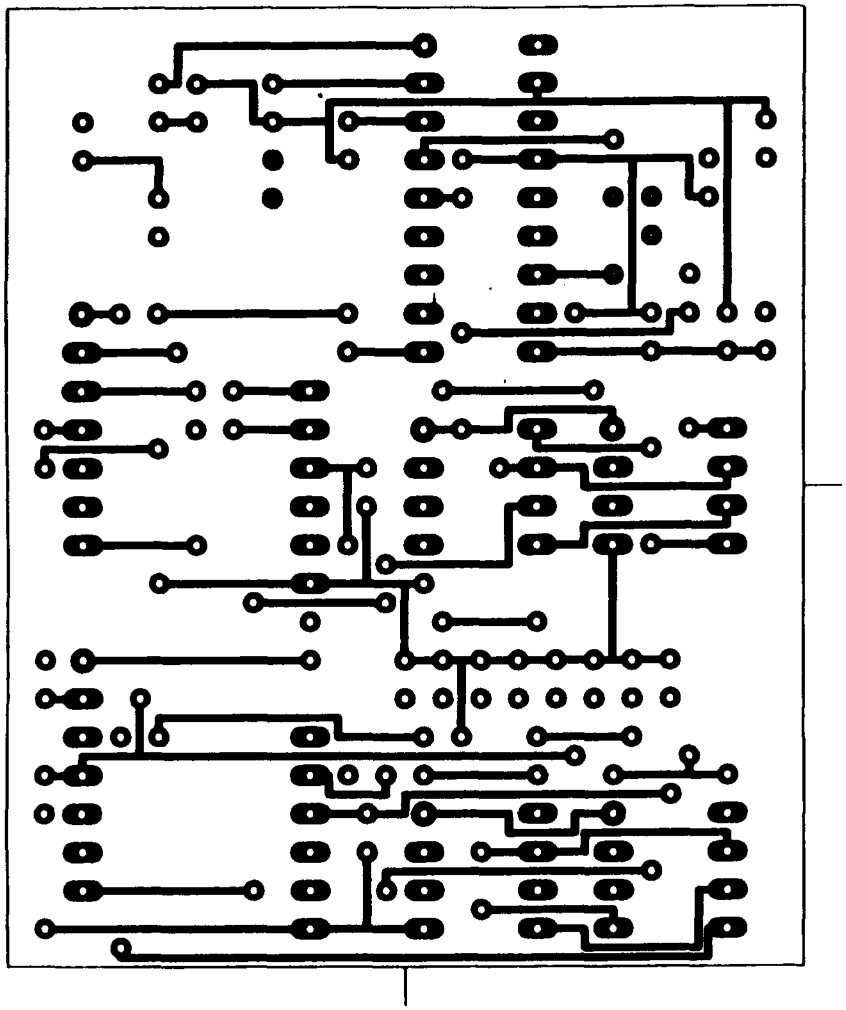
Appendix E - PCB layouts



Noise Board 1.0 PCB

Tue, Aug 15, 1989 12:05 AM

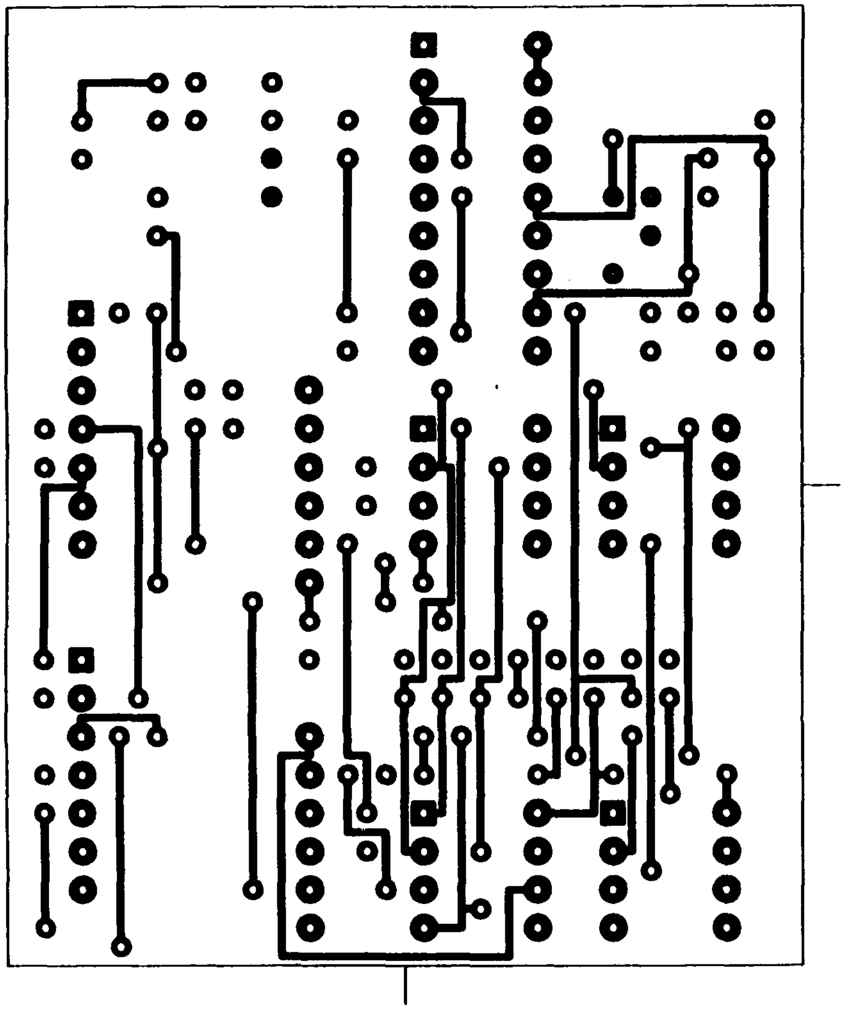
Component Xray Scale2:1



Noise Board 1.0 PCB

Tue, Aug 15, 1989 12:05 AM

Component Side   Scale2:1



Noise Board 1.0 PCB

Tue, Aug 15, 1989 12:05 AM

Solder Side Scale2:1

Appendix E - FLT-U2 data

FLT-U2
Microelectronic Universal Active Filter

FEATURES

- State variable filter
- LP, BP, or HP functions
- 2-Pole response
- Low-noise operational amplifiers
- −55°C to +125°C Operation
- Low cost

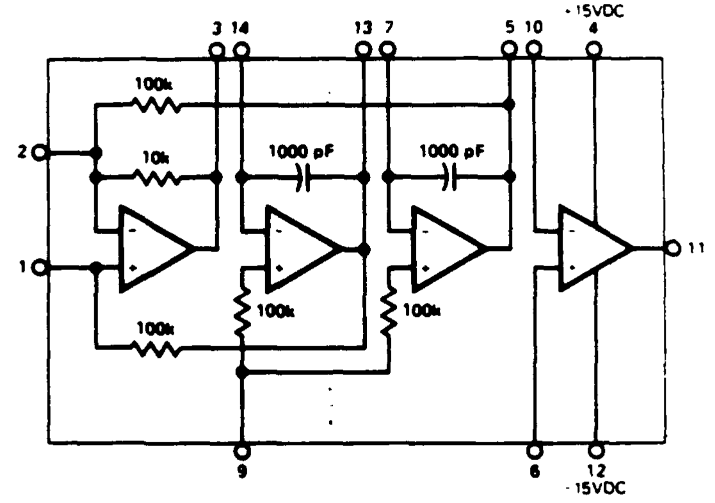
GENERAL DESCRIPTION

The FLT-U2 is a universal active filter manufactured with thick-film hybrid technology. It uses the state variable active filter principle to implement a second order transfer function. Three committed operational amplifiers are used for the second order function while a fourth uncommitted operational amplifier can be used as a gain stage, summing amplifier, buffer amplifier, or to add another independent real pole.

Two-pole lowpass, bandpass, and highpass output functions are available simultaneously from three different outputs, and notch and allpass functions are available by combining these outputs in the uncommitted operational amplifier. To realize higher order filters, several FLT-U2's can be cascaded. Q range is from 0.1 to 1,000 and resonant frequency range is 0.001 Hz to 200 kHz. Frequency stability is 0.01%/°C and resonant frequency accuracy is within ±5% of calculated values. Frequency tuning is done by two external resistors and Q tuning by a third external resistor. For resonant frequencies below 50 Hz, two external tuning capacitors must be added. Exact tuning of the resonant frequency is done by varying one of the resistors around its calculated value.

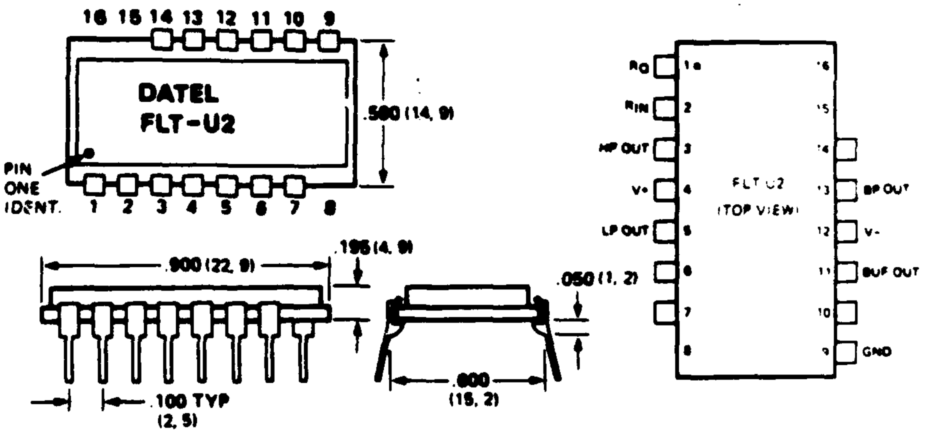
The internal operational amplifiers in the FLT-U2 have 3 MHz gain bandwidth products and a wideband input noise specification of only 10 nV/√Hz. This results in considerably improved operation over most other competitive active filters which employ lower performance amplifiers. By proper selection of external components any of the popular filter types such as Butterworth, Bessel, Chebyshev, or Elliptic may be designed. Applications include audio, tone signalling, sonar, data acquisition, and feedback control systems.

Two models are available for operation over the commercial, 0°C to +70°C, and military, −55°C to +125°C, temperature ranges.



MECHANICAL DIMENSIONS INCHES (MM)

CONNECTIONS DIAGRAM



FUNCTIONAL SPECIFICATIONS

Typical at 25°C, ±15V supplies, unless otherwise stated.

FILTER CHARACTERISTICS	
Frequency Range[14]	0.001 Hz to 200 kHz
Q Range[1]	0.1 to 1.000
f_0 Accuracy	±5%
f_0 Temperature Coefficient	0.01%/°C
Voltage Gain[1]	0.1 to 1.000
AMPLIFIER CHARACTERISTICS	
Input Offset Voltage	0.5 mV typ., 6 mV max.
Input Bias Current	40 nA typ., 500 nA max.
Input Offset Current	5 nA typ., 200 nA max.
Input Impedance	5 Megohms
Input Com. Mode Voltage Range	±12V min.
Input Voltage Noise, wideband	10 nV/√Hz
Output Voltage Range	±10V min.
Output Current	±5 mA min.
Open Loop Voltage Gain	300.000
Common Mode Rejection Ratio	100 dB
Power Supply Rejection	10 μV/V
Unity Gain Bandwidth	3 MHz
Slew Rate	1 V/μsec.
POWER SUPPLY REQUIREMENTS	
Voltage, rated performance	±15V dc
Voltage Range, operating	±5V to ±18V
Quiescent Current	11.5 mA max.
PHYSICAL/ENVIRONMENTAL	
Operating Temperature Range	
FLT-U2	0°C to +70°C
FLT-U2M	−55°C to +125°C
Storage Temperature Range	−55°C to +125°C
Case	Ceramic 16-pin DIP (double-spaced)

FOOTNOTE:

[1] $f_0Q \leq 5 \times 10^5$ optimally

TECHNICAL NOTES

1. The FLT-U2 has simultaneous lowpass, bandpass, and highpass output functions. The chosen output for a particular function will be at unity gain based on Tables II and III. This means that the other two unused outputs will be at other gain levels. The gain of the lowpass output is always 10 dB higher than the gain of the bandpass output and 20 dB higher than the gain of the highpass output.
2. When tuning the filter and checking it over its frequency range, the outputs should be checked with a scope to make sure there is no waveform clipping present, as this will affect the operation of the filter. In particular the lowpass output should be checked since its gain is the highest.
3. Check f_1, the center frequency for bandpass and the cutoff frequency for lowpass or highpass, at the bandpass output. Here the peaking frequency can easily be determined for high Q filters and the 0° or 180° phase frequency can easily be determined for low Q filters (depending on whether inverting or noninverting).
4. Tuning resistors should be 1% metal film resistors with 100 ppm/°C temperature stability or better for best performance. Likewise external tuning capacitors should be NPO ceramic or other stable capacitor types.

THEORY OF OPERATION

The FLT-U2 block diagram is shown in Figure 1. This is a second order state-variable filter using three operational amplifiers. Lowpass, bandpass, and highpass transfer functions are simultaneously produced at its three output terminals. These three transfer functions are characterized by the following second order equations:

$$H(s) = \frac{K_1}{S^2 + \frac{\omega_0}{Q}S + \omega_0^2} \qquad \text{LOWPASS}$$

$$H(s) = \frac{K_2S}{S^2 + \frac{\omega_0}{Q}S + \omega_0^2} \qquad \text{BANDPASS}$$

$$H(s) = \frac{K_3S^2}{S^2 + \frac{\omega_0}{Q}S + \omega_0^2} \qquad \text{HIGHPASS}$$

where K_1, K_2, and K_3 are arbitrary gain constants.

A second order system is characterized by the location of its poles in the s-plane as shown in Figure 2. The natural radian frequency of this system is ω_0. In Hertz this is $f_0 = \frac{\omega_0}{2\pi}$

The resonant radian frequency of the circuit is different from the natural radian frequency and is:

$$\omega_1 = \omega_0 \sin \theta = \sqrt{\omega_0^2 - \sigma_1^2}$$

The damping factor d determines the amount of peaking in the filter frequency response and is defined as:

$$d = \cos \theta$$

The point at which the peaking becomes zero is called critical damping and is $d = \sqrt{2/2}$.

Q is found from d and is a measure of the sharpness of the resonance of the peaking:

$$Q = \frac{1}{2d}$$

$$\text{Also, } Q = \frac{f_0}{-3\text{ dB Bandwidth}} = \frac{\omega_0}{2\sigma_1}$$

For high Q filters the natural frequency and resonant frequency are approximately equal.

$$\omega_1 \simeq \omega_0 \text{ or } f_1 \simeq f_0$$

This is true since $\omega_1 = \omega_0 \sin \theta$ and $\sin \theta \simeq 1$ as the poles move close to the $j\omega$ axis in the s-plane.

For high Qs ($Q > 1$) we therefore have for the second order filter:

$$\begin{aligned} f_0 &\simeq \text{Bandpass center frequency} \\ &\simeq \text{Lowpass corner frequency} \\ &\simeq \text{Highpass corner frequency} \end{aligned}$$

In the simplified tuning procedure which follows, the tuning is accomplished by independently setting the natural frequency and Q of the filter. This is done most simply by assuming unity gain for the output of the desired filter function. Unity gain means a gain of one (±) at dc for lowpass, at center frequency for bandpass, and at high frequency ($f > > f_0$) for highpass. Unity gain does not apply to all outputs simultaneously but only to the chosen output based on the component values given in the tables. Figure 3 shows the relative gains of the three simultaneous outputs assuming the bandpass gain is set to unity. Note that lowpass gain is always 10 dB higher than bandpass gain and highpass gain is always 10 dB lower than bandpass gain.

SIMPLIFIED TUNING PROCEDURE

1. Select the desired transfer function (lowpass, bandpass, or highpass) and inverted or noninverted output. From this determine the filter configuration (inverting or noninverting) using Table I.

TABLE I FILTER CONFIGURATION

	LP	BP	HP
INVERTING INPUT	INV	NON-INV	INV
NONINVERTING INPUT	NON-INV	INV	NON-INV

2. Starting with the desired natural frequency and Q (determined from the filter transfer function or s-plane diagram), compute f_0Q. For $f_0Q > 10^4$ the actual realized Q will exceed the calculated value. At $f_0Q = 10^4$ the increase is about 1% and at $f_0Q = 10^5$ it is about 20%.

3. **Inverting Configuration.** Using the value of Q from Step 2 find R_1 and R_3 from Table II. R_2 is open, or infinite.

TABLE II INVERTING CONFIGURATION

	R_1	R_2	R_3
LOWPASS	100K	OPEN	$\frac{100K}{3.80\,Q-1}$
BANDPASS	Q X 31.6K	OPEN	$\frac{100K}{3.48\,Q}$
HIGHPASS	10K	OPEN	$\frac{100K}{6.64\,Q-1}$

4. **Noninverting Configuration.** Using the value of Q from Step 2 find R_2 and R_3 from Table III. R_1 is open, or infinite.

TABLE III NONINVERTING CONFIGURATION

	R_1	R_2	R_3
LOWPASS	OPEN	$\frac{316K}{Q}$	$\frac{100K}{3.16\,Q-1}$
BANDPASS	OPEN	100k	$\frac{100K}{3.48\,Q-1}$
HIGHPASS	OPEN	$\frac{31.6K}{Q}$	$\frac{100K}{0.316\,Q-1}$

5. Using the value of f_0 from Step 2, set the natural frequency of the filter by finding R_4 and R_5 from the equation:

$$R_4 = R_5 = \frac{5.03 \text{ X } 10^7}{f_0}$$

where R_4 and R_5 are in ohms and f_0 is in Hertz. The natural frequency varies as $\sqrt{R_4R_5}$ and therefore one value may be increased and the other decreased and the natural frequency will be constant if the geometric mean is constant. To maintain constant bandwidth at the bandpass output while varying center frequency, fix R_4 and vary R_5.

6. For $f_0 < 50$ Hz the internal 1000 pF capacitors should be shunted with external capacitors across pins 5 & 7 and 13 & 14. If equal value capacitors are used, R_4 and R_5 are then computed from:

$$R_4 = R_5 = \frac{5.03 \text{ x } 10^{10}}{f_0C} \text{ (C in pF)}$$

For unequal value capacitors this becomes:

$$R_4 = R_5 = \frac{5.03 \text{ x } 10^{10}}{f_0\sqrt{C_1C_2}} \text{ (}C_1C_2\text{ in pF)}$$

In both cases the capacitance is the sum of the external values and the internal 1000 pF values.

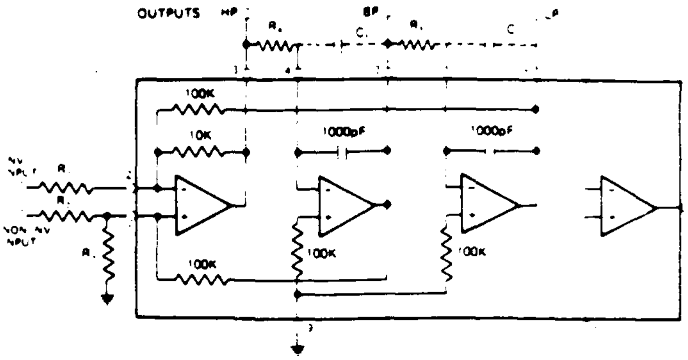


Figure 1.
FLT-U2 Block Diagram

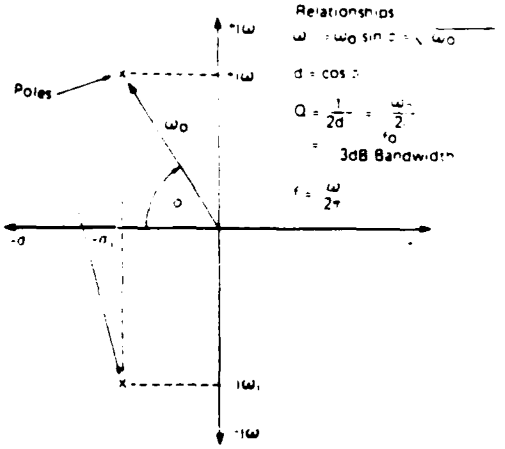


Figure 2.
S-Plane Diagram

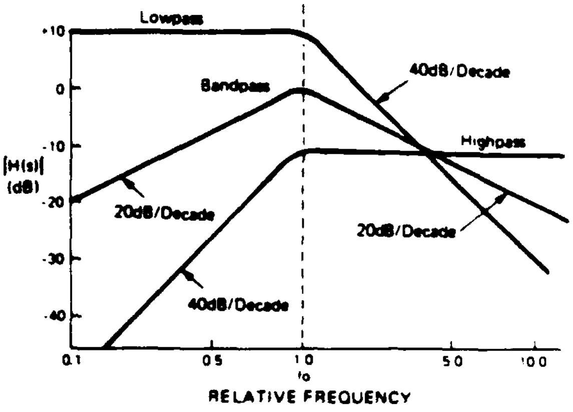


Figure 3.
Relative Gains of Simultaneous Outputs, Q=1

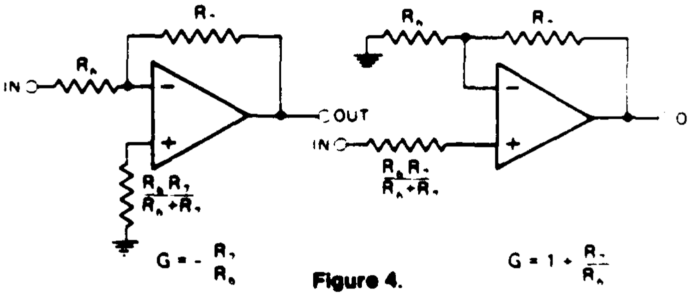


Figure 4.
Uncommitted Op Amp Gain Configurations

SIMPLIFIED TUNING PROCEDURE, (Cont'd)

7. This procedure is based on unity gain output for the desired function. For additional gain, the fourth uncommitted operational amplifier should be used as an inverting or noninverting gain stage following the selected output. See Figure 4. A third pole on the real axis of the s-plane may also be added to the transfer function by adding a capacitor to the gain stage as shown in Figure 5.

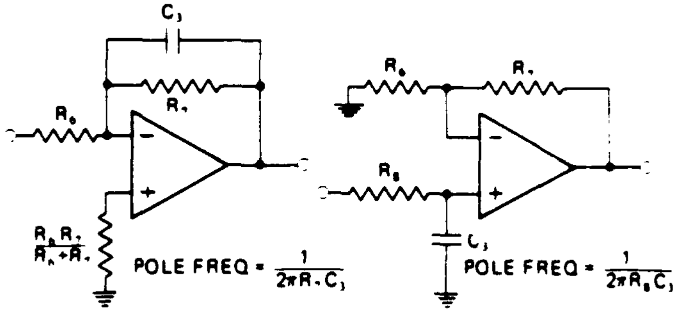


Figure 5.
Using the Uncommitted Op Amp to Add a Real Axis Pole

FILTER DESIGN EXAMPLES

Bandpass Filter With 1 kHz Center Frequency Q = 10 and Inverted Output

1. From Table I the noninverting configuration is chosen to realize an inverted bandpass output $f_0Q = 10^4$ which means the realized Q will be about 1% higher than calculated.
2. From Table III, using Q = 10, we find:

$$R_1 = \text{open}$$
$$R_2 = 100\text{K ohms}$$
$$R_3 = \frac{100\text{K}}{3.48\text{ Q-1}} = \frac{100\text{K}}{33.8} = 2.96\text{K ohms}$$

3. Using f_0 of 1 kHz, R_4 and R_5 are found from the equation:

$$R_4 = R_5 = \frac{5.03 \times 10^7}{1000} = 50.3\text{K ohms}$$

4. This completes the filter design which is shown in Figure 6. To choose the nearest 1% standard value resistors either 49.9K or 51.1K ohms could be used; likewise one value of 49.9K and one of 51.1K could be used giving the geometric mean of $\sqrt{R_4R_5} = \sqrt{49.9\text{K} \times 51.1\text{K}} = 50.5\text{K}$ which is even closer. But due to the filter ±5% frequency tolerance it may be better to hold R_4 constant while varying R_5 to tune it exactly.

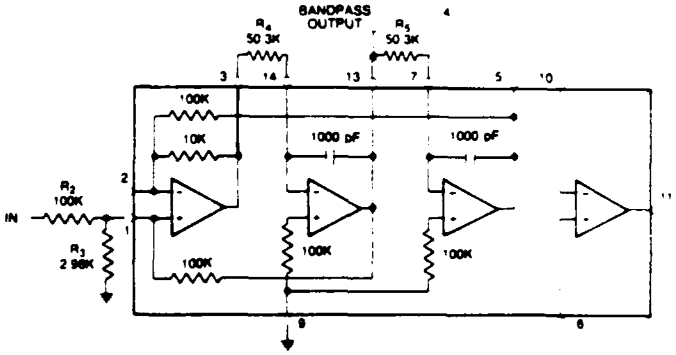


Figure 6.
Bandpass Filter Example

Three-Pole Noninverting Butterworth Low Pass Filter With dc Gain of 10 and Cutoff Frequency of 5 kHz.

The s-plane diagram of the 3-pole Butterworth filter is shown in Figure 7. We will use a second order filter to realize the two complex conjugate poles and the uncommitted operational amplifier to provide the third real axis pole and a dc gain of 10.

1. From Table I, the noninverting filter configuration would normally be used to give a noninverting low pass output. In this case, however, we choose an inverting uncommitted op amp with a gain of 10 and therefore we use the inverting configuration for the filter.. By comparing the second order portion of the Butterworth function $S^2 + \omega_0S + \omega_0^2$ to the standard second order function $S^2 + \omega_0S + \omega_0^2$ we find Q = 1 f_0Q is then 5×10^3 so that Q will not exceed its specified value.
2. From Table II, using Q = 1, we find:

$$R_1 = 100\text{K ohms}$$
$$R_2 = \text{open}$$
$$R_3 = \frac{100\text{K}}{3.80\text{ Q-1}} = 35.7\text{K ohms}$$

3. Using f_0 of 5 kHz, R_4 and R_5 are found from the equation:

$$R_4 = R_5 = \frac{5.03 \times 10^7}{5000} = 10.1\text{K ohms}$$

4. For the uncommitted output amplifier, a gain of −10 is required. This defines $R_7/R_6 = 10$ and we arbitrarily choose $R_6 = 2\text{K}$, $R_7 = 20\text{K}$ ohms.

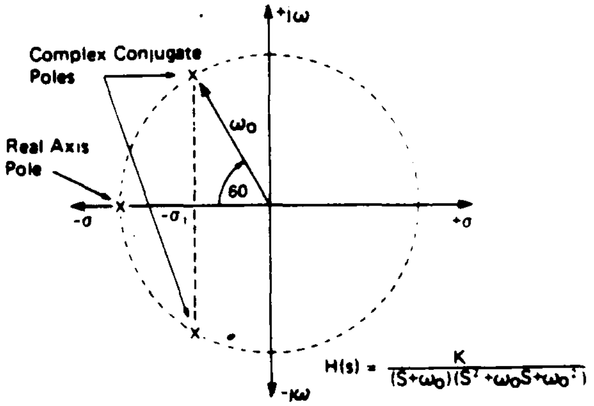


Figure 7.
S-Plane Diagram of 3-Pole Butterworth Lowpass Filter

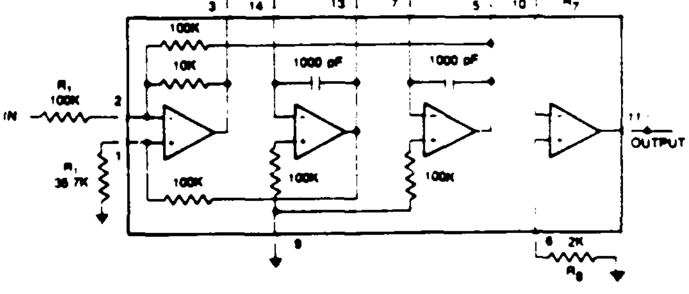


Figure 8.
Three Pole Butterworth Low Pass Filter Example

FILTER DESIGN EXAMPLES, (Cont'd)

5. The final step is to realize the real axis pole of the Butterworth filter. This pole is at 5 kHz and is set by using capacitor C_3 across the feedback resistor R_7:

$$C_3 = \frac{1}{2\pi f R_7} = \frac{1}{6.28 \times 5 \times 10^3 \times 20 \times 10^3} = 1590 \text{ pF}$$

6. This completes the 3-pole Butterworth filter which is shown in Figure 8.

Highpass Filter with Gain of −1, 20 kHz Cutoff Frequency, and Critical Damping

1. From Table I the inverting configuration must be used to realize a highpass gain of −1. An s-plane diagram of this function is shown in Figure 9. Critical damping requires the pole positions to be on a line 45° with respect to the real axis and this results in no frequency peaking. The damping factor d is:

$$d = \cos\theta = \cos 45° = 0.707$$

and $$Q = \frac{1}{2d} = \frac{1}{2(0.707)} = 0.707$$

Because this is a low Q system the natural frequency will not be the same as the highpass cutoff frequency f_1. From Figure 9:

$$f_0 = \frac{f_1}{\cos\theta} = \frac{20 \text{ kHz}}{0.707} = 28.3 \text{ kHz}$$

Then $f_0 Q = 0.707 \times 28.3 \times 10^3 = 2 \times 10^4$ and the Q will exceed its desired value by slightly over 1%.

2. From Table II, using Q = 0.707 we find:

R_1 = 10K ohms
R_2 = open

$$R_3 = \frac{100\text{K}}{6.64\,Q\text{-}1} = \frac{100\text{K}}{3.69} = 27.1\text{K ohms}$$

3. Using f_0 = 28.3 kHz, R_4 and R_5 are found from the equation:

$$R_4 = R_5 = \frac{5.03 \times 10^7}{28.3 \times 10^3} = 1.78\text{K ohms}$$

4. This completes the highpass filter design which is shown in Figure 10. When using this filter, care should be exercised so that clipping does not occur in the filter due to excessive input levels. If clipping occurs, the filter will not operate properly. Clipping will first occur at the lowpass output around f_0 since its gain is 20 dB higher than the highpass output. The signal level should be reduced so that clipping does not occur anywhere in the frequency range used. If higher signal level is required, the highpass output should be amplified by a gain stage using the uncommitted operational amplifier.

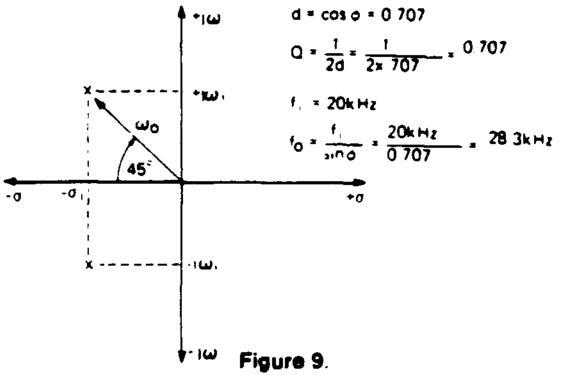

+jω
+jω₁
ω₀
45°
-σ
-σ₁
+σ
-jω₁
-jω
d = cos θ = 0.707
Q = 1/2d = 1/2x.707 = 0.707
f₁ = 20kHz
f₀ = f₁/sin θ = 20kHz/0.707 = 28.3kHz

Figure 9.
S-Plane Diagram of Highpass Filter with Critical Damping

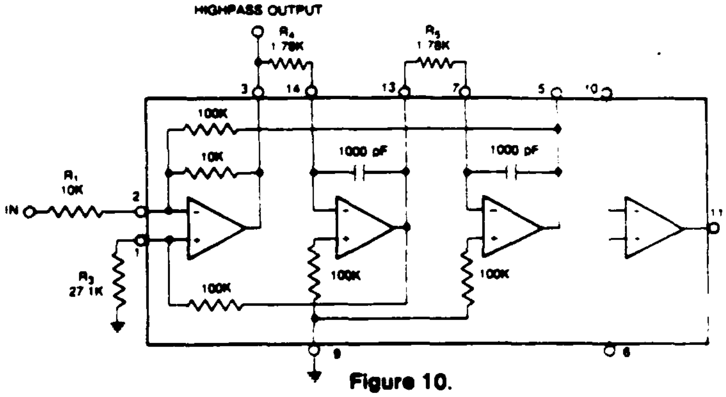


Figure 10.
Highpass Filter Example

ADVANCED FILTERS

All of the common filter types can be realized by using cascaded FLT-U2 stages. This includes multi-pole Butterworth, Bessel, Chebyshev, and Elliptic types. The basic procedure is to implement each pole pair with a single FLT-U2 and cascade enough units to realize all poles. A real axis pole is implemented by an uncommitted operational amplifier stage. Each stage should be separately tuned with an oscillator and scope and then the stages connected together and checked. See Figure 11.

A notch filter can be constructed in several ways. The first way is to use the FLT-U2 as an inverting bandpass filter and sum the output of the filter with the input signal by means of the uncommitted operational amplifier. This produces a net subtraction at the center frequency of the bandpass which produces a null at the output of the amplifier. (See Figure 12.) Likewise lowpass and highpass outputs (which are always in phase) can be subtracted from each other with an external operational amplifier. The highpass output must have some gain added to it, however, so that its gain is equal to that of the lowpass output. A third method is to use two separate FLT-U2's, one as a two-pole lowpass filter and the other as a two-pole highpass filter. Again the outputs are subtracted in an operational amplifier. This method permits independent tuning of the two sections to get the best null response.

Further discussion of filter designs is beyond the scope of this data sheet and the user is referred to the various texts on filter design, some of which are listed below.

Estep, G.J., *The State Variable Active Filter Configuration Handbook*, 2nd Edition, Agoura, Ca., 1974.

Reference Data for Radio Engineers, Howard W. Sams & Co. Inc., 5th Edition.

Christian, E., and Eisenmann, E., *Filter Design Tables and Graphs*, McGraw-Hill Book Co., 1974.

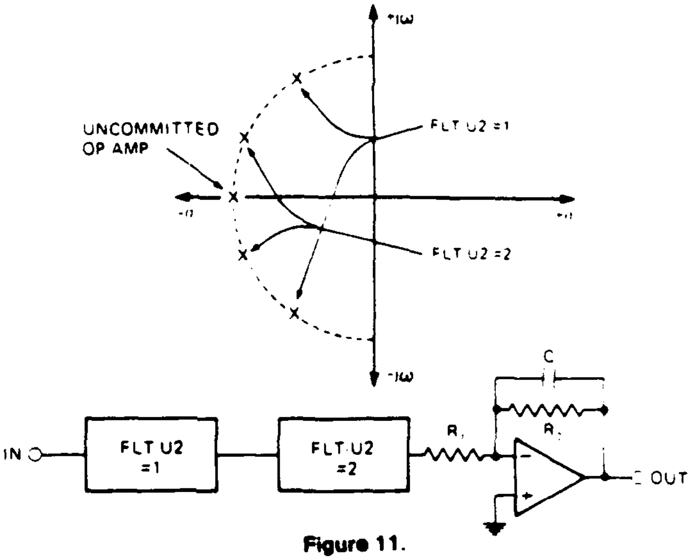


Figure 11.
Realization of a Complex Multipole Filter

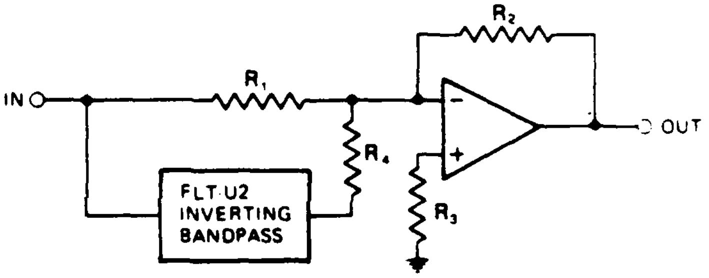
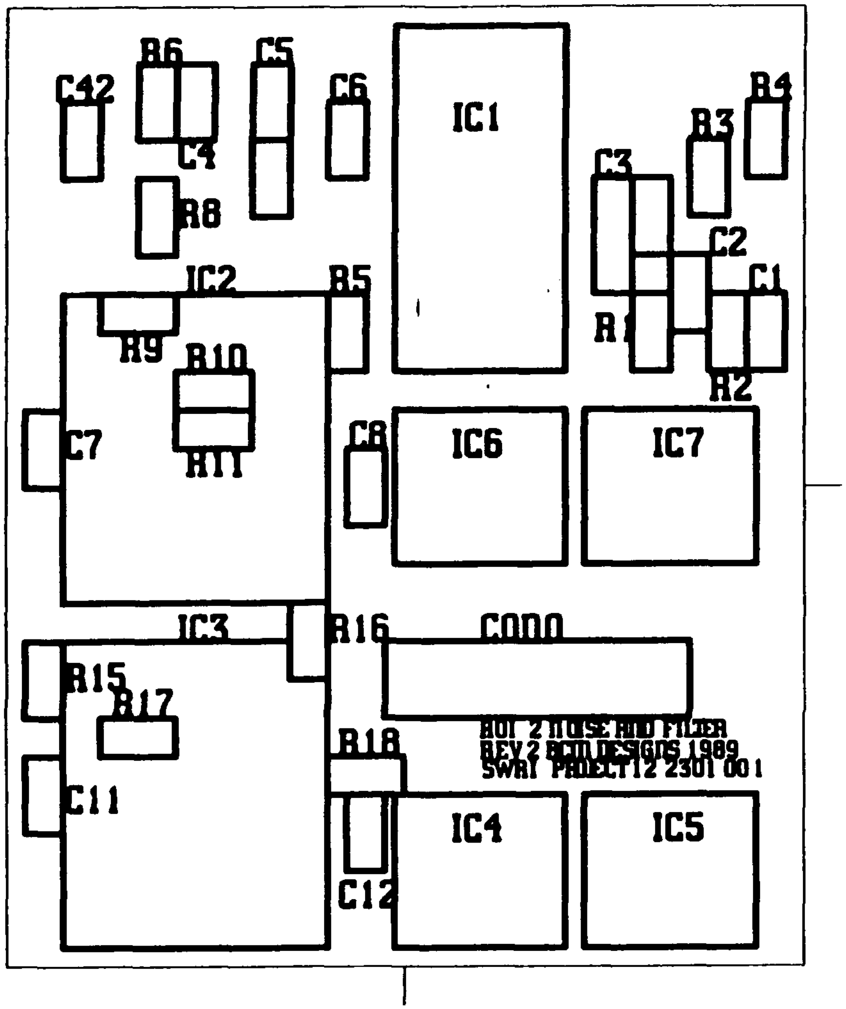


Figure 12.
Realization of Notch Filter

ORDERING INFORMATION

MODEL	OPERATING TEMP. RANGE
FLT-U2	0°C to +70°C
FLT-U2M	−55°C to +125°C

Appendix E - PCB silks



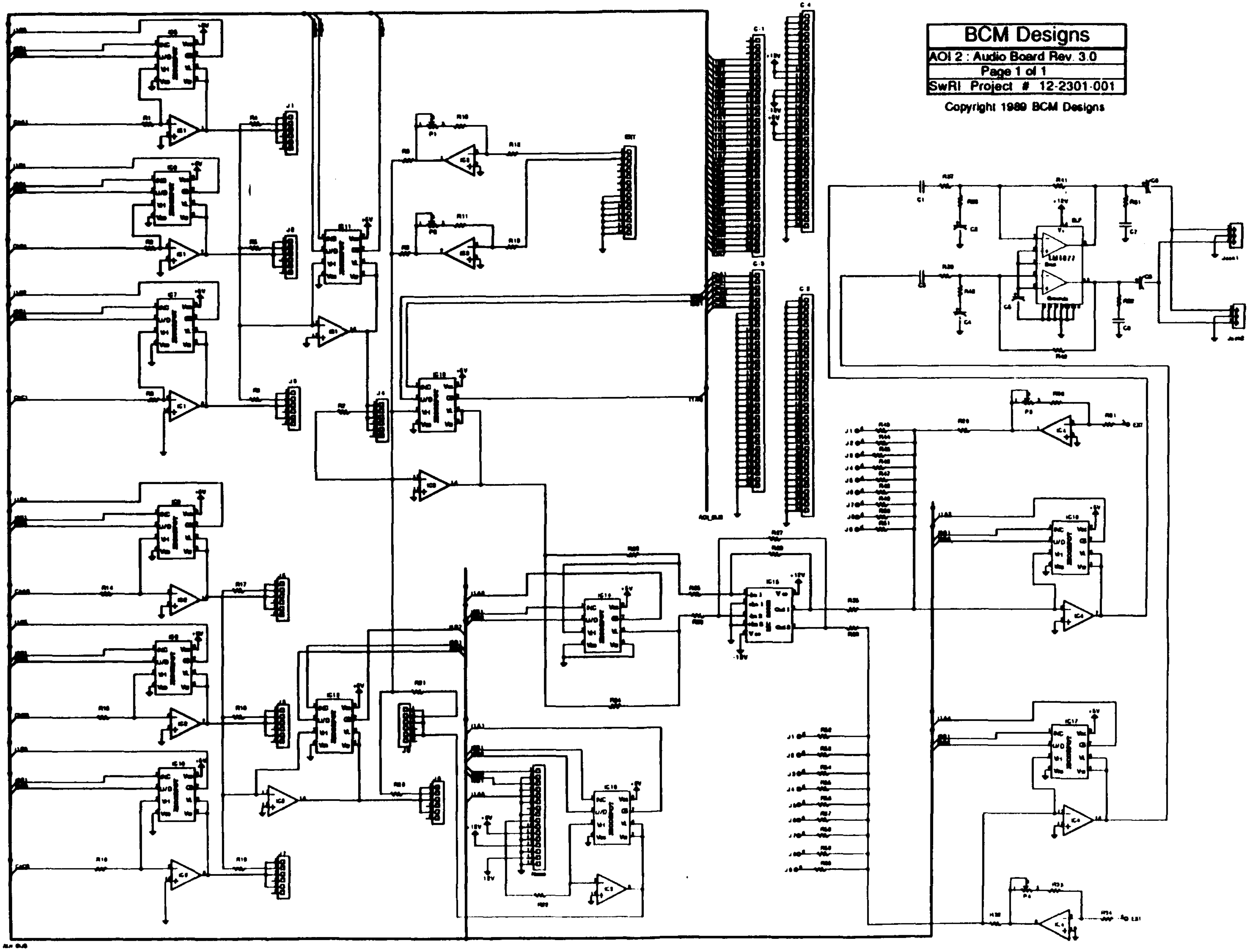
Noise Board SILK

Fri, Sep 15, 1989 10:42 AM

Component Side Scale2:1

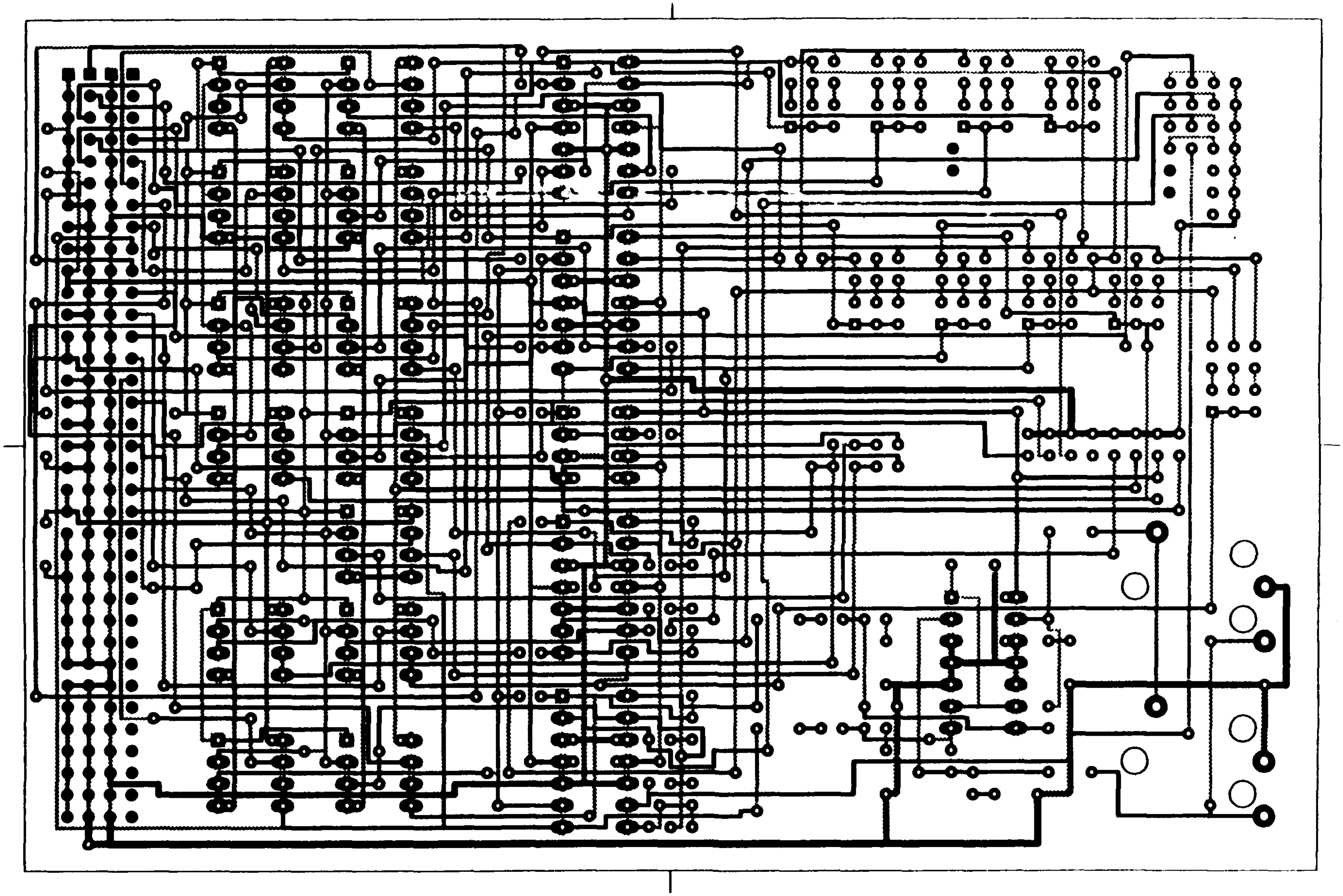
Appendix F - Audio mixer board

Appendix F - Schematic



BCM Designs
AOI 2 : Audio Board Rev. 3.0
Page 1 of 1
SwRI Project # 12-2301-001

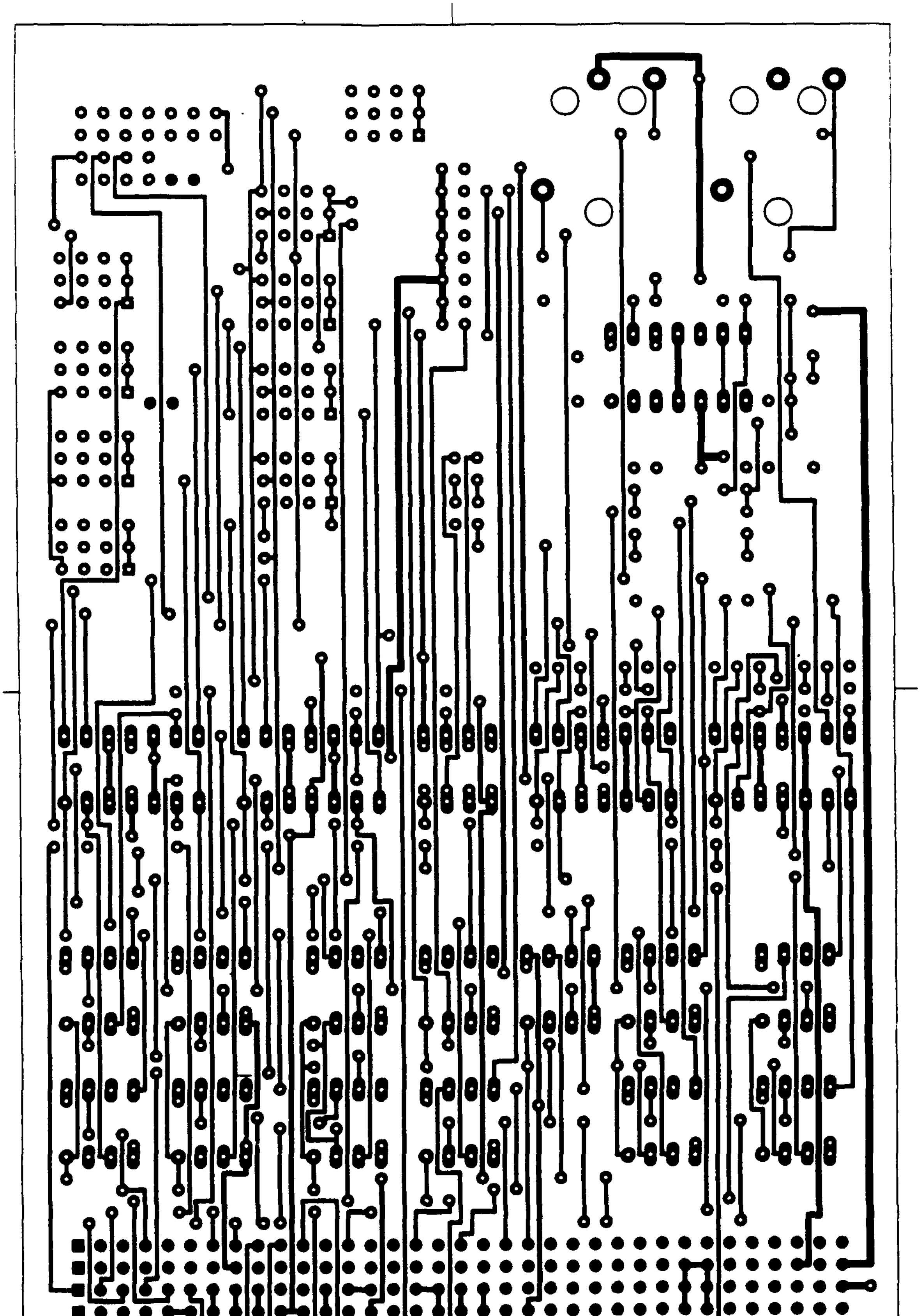
Appendix F - Layout



Audio Final PCB

Wed, Sep 13, 1989 11:23 AM

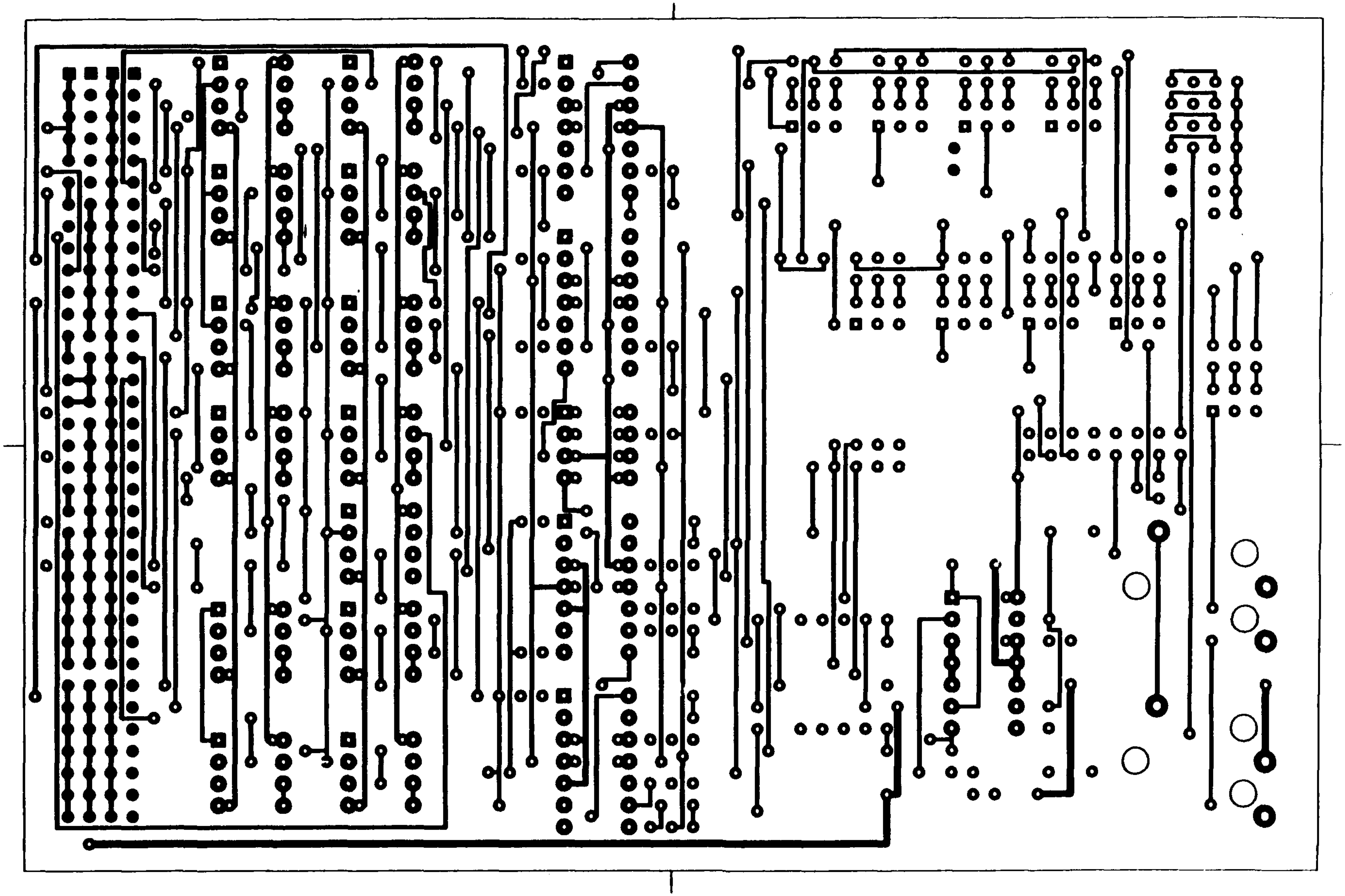
Component Xray Scale150%



Audio Final PCB

Wed, Sep 13, 1989 11:23 AM

Component Side Scale150%

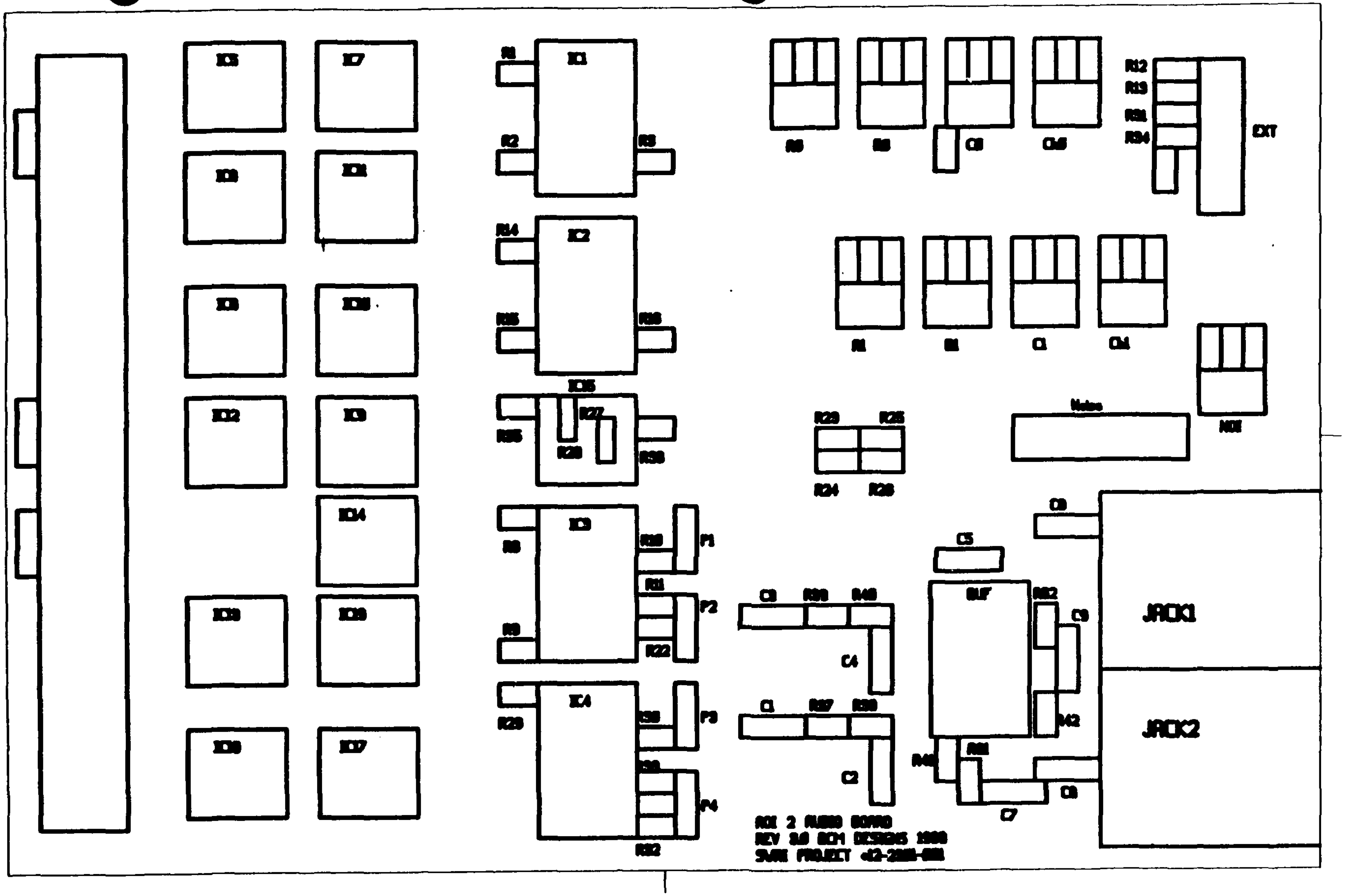


Audio Final PCB

Wed, Sep 13, 1989 11:23 AM

Solder Side Scale150%

Appendix F - PCB silks



Audio SILK 2

Sat, Sep 23, 1989 4:10 PM

Component Xray Scale150%

Appendix F - Data Sheets

op amps

Xicor digital pots

MOTOROLA

MC34080/MC35080 thru MC34085/MC35085

2

HIGH SLEW RATE, WIDE BANDWIDTH, JFET INPUT OPERATIONAL AMPLIFIERS

These devices are a new generation of high speed JFET input monolithic operational amplifiers. Innovative design concepts along with JFET technology provide wide gain bandwidth product and high slew rate. Well matched JFET input devices and advanced trim techniques ensure low input offset errors and bias currents. The all NPN output stage features large output voltage swing, no deadband crossover distortion, high capacitive drive capability, excellent phase and gain margins, low open-loop output impedance, and symmetrical source sink ac frequency response.

This series of devices are available in standard or prime performance (A suffix) grades, fully compensated or decompensated ($A_{VCL} \geq 2$) and are specified over commercial or Military temperature ranges. They are pin compatible with existing Industry standard operational amplifiers, and allow the designer to easily upgrade the performance of existing designs.

- Wide Gain Bandwidth: 8.0 MHz for Fully Compensated Devices
  16 MHz for Decompensated Devices
- High Slew Rate: 25 V/μs for Fully Compensated Devices
  50 V/μs for Decompensated Devices
- High Input Impedance: 10^{12} Ω
- Input Offset Voltage: 0.5 mV Maximum (Single Amplifier)
- Large Output Voltage Swing: − 14.7 V to + 14 V for
  V_{CC}/V_{EE} = ±15 V
- Low Open-Loop Output Impedance: 30 Ω @ 1.0 MHz
- Low THD Distortion: 0.01%
- Excellent Phase/Gain Margins: 55°/7.6 dB for Fully Compensated Devices

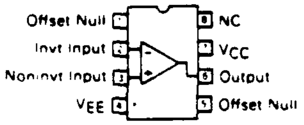
HIGH PERFORMANCE JFET INPUT OPERATIONAL AMPLIFIERS

P SUFFIX
PLASTIC PACKAGE
CASE 626-05

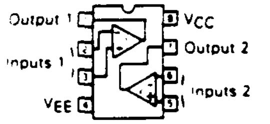
U SUFFIX
CERAMIC PACKAGE
CASE 693-02

D SUFFIX
PLASTIC PACKAGE
CASE 751-02
SO-8

PIN ASSIGNMENTS

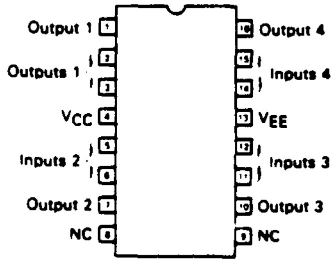


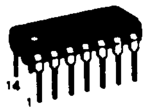
Single, Top View



Dual, Top View

DW SUFFIX
PLASTIC PACKAGE
CASE 751G-01
SO-16L

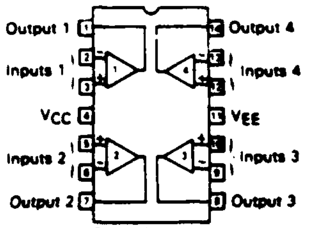




P SUFFIX
PLASTIC PACKAGE
CASE 646-06

L SUFFIX
CERAMIC PACKAGE
CASE 632-08

PIN ASSIGNMENTS



Quad, Top View

ORDERING INFORMATION

Op Amp Function	Fully Compensated	$A_{VCL} \geq 2$ Decompensated	Temperature Range	Package
Single	MC35081U,AU MC34081D,AD MC34081P,AP	MC35080U,AU MC34080D,AD MC34080P,AP	− 55 to + 125°C 0 to + 70°C 0 to + 70°C	Ceramic DIP SO-8 Plastic DIP
Dual	MC34082P,AP	MC34083P,AP	0 to + 70°C	Plastic DIP
Quad	MC35084L,AL MC34084DW MC34084P,AP	MC35085L,AL MC34085DW MC34085P,AP	− 55 to + 125°C 0 to + 70°C 0 to + 70°C	Ceramic DIP SO-16L Plastic DIP

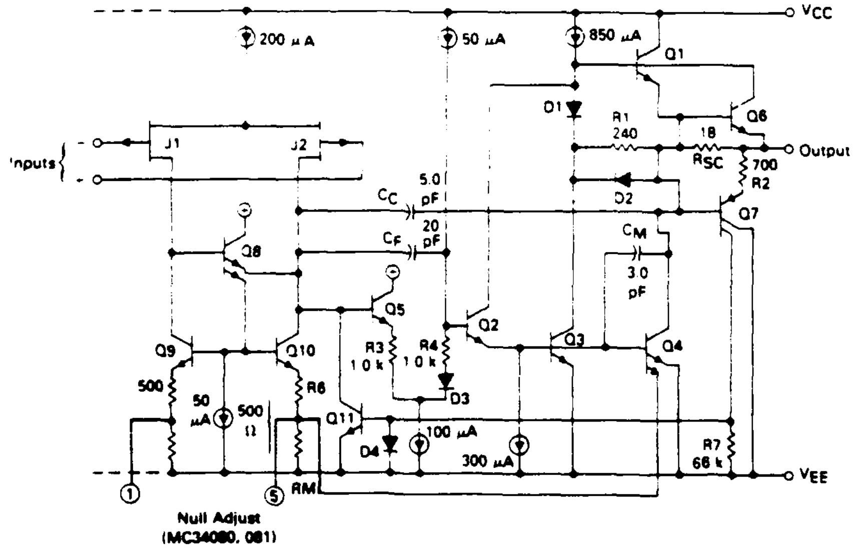
MC34080, MC35080 Series

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage (from V_{CC} to V_{EE})	V_S	+44	Volts
Input Differential Voltage Range	V_{IDR}	Note 1	Volts
Input Voltage Range	V_{IR}	Note 1	Volts
Output Short-Circuit Duration (Note 2)	t_S	Indefinite	Seconds
Operating Ambient Temperature Range	T_A		°C
MC35XXX		−55 to +125	
MC34XXX		0 to +70	
Operating Junction Temperature	T_J		°C
Ceramic Package		+165	
Plastic Package		+125	
Storage Temperature Range	T_{stg}		°C
Ceramic Package		−65 to +165	
Plastic Package		−55 to +125	

NOTES:
1. Either or both input voltages must not exceed the magnitude of V_{CC} or V_{EE}
2. Power dissipation must be considered to ensure maximum junction temperature (T_J) is not exceeded.

EQUIVALENT CIRCUIT SCHEMATIC (EACH AMPLIFIER)



DC ELECTRICAL CHARACTERISTICS (V_{CC} = +15 V, V_{EE} = −15 V, T_A = T_{low} to T_{high} [Note 3], unless otherwise noted)

Characteristic	Symbol	A Suffix			Non-Suffix			Unit
		Min	Typ	Max	Min	Typ	Max	
Input Offset Voltage (Note 4)	V_{IO}							mV
Single								
T_A = +25°C		—	0.3	0.5	—	0.5	1.0	
T_A = 0°C to +70°C (MC34080, MC34081)		—	—	2.5	—	—	3.0	
T_A = −55°C to +125°C (MC35080, MC35081)		—	—	3.5	—	—	4.0	
Dual								
T_A = +25°C		—	0.6	1.0	—	1.0	3.0	
T_A = 0°C to +70°C (MC34082, MC34083)		—	—	3.0	—	—	5.0	
T_A = −55°C to +125°C (MC35082, MC35083)		—	—	4.0	—	—	6.0	
Quad								
T_A = +25°C		—	3.0	6.0	—	6.0	12	
T_A = 0°C to +70°C (MC34084, MC34085)		—	—	8.0	—	—	14	
T_A = −55°C to +125°C (MC35084, MC35085)		—	—	9.0	—	—	15	
Average Temperature Coefficient of Offset Voltage	$\Delta V_{IO}/\Delta T$	—	10	—	—	10	—	µV/°C
Input Bias Current (V_{CM} = 0 Note 5)	I_{IB}							
T_A = +25°C		—	0.06	0.2	—	0.06	0.2	nA
T_A = 0°C to +70°C		—	—	4.0	—	—	4.0	
T_A = −55°C to +125°C		—	—	50	—	—	50	
Input Offset Current (V_{CM} = 0 Note 5)	I_{IO}							
T_A = +25°C		—	0.02	0.1	—	0.02	0.1	nA
T_A = 0°C to +70°C		—	—	2.0	—	—	2.0	
T_A = −55°C to +125°C		—	—	25	—	—	25	
Large Signal Voltage Gain (V_O = ±10 V, R_L = 2.0 k)	A_{VOL}							V/mV
T_A = +25°C		50	80	—	25	80	—	
T_A = T_{low} to T_{high}		25	—	—	15	—	—	
Output Voltage Swing	V_{OH}							V
R_L = 2.0 k, T_A = +25°C		13.2	13.7	—	13.2	13.7	—	
R_L = 10 k, T_A = +25°C		13.4	13.9	—	13.4	13.9	—	
R_L = 10 k, T_A = T_{low} to T_{high}		13.4	—	—	13.4	—	—	
	V_{OL}							
R_L = 2.0 k, T_A = +25°C		—	−14.1	−13.5	—	−14.1	−13.5	
R_L = 10 k, T_A = +25°C		—	−14.7	−14.1	—	−14.7	−14.1	
R_L = 10 k, T_A = T_{low} to T_{high}		—	—	−14.0	—	—	−14.0	
Output Short-Circuit Current (T_A = +25°C) Input Overdrive = 1.0 V, Output to Ground	I_{SC}							mA
Source		20	31	—	20	31	—	
Sink		20	28	—	20	28	—	
Input Common Mode Voltage Range T_A = +25°C	V_{ICR}	(V_{EE} + 4.0) to (V_{CC} − 2.0)			(V_{EE} + 4.0) to (V_{CC} − 2.0)			V
Common Mode Rejection Ratio (R_S ≤ 10 k, T_A = +25°C)	CMRR	75	90	—	70	90	—	dB
Power Supply Rejection Ratio (R_S = 100 Ω, T_A = 25°C)	PSRR	75	86	—	70	86	—	dB
Power Supply Current	I_D							mA
Single								
T_A = +25°C		—	2.5	3.4	—	2.5	3.4	
T_A = T_{low} to T_{high}		—	—	4.2	—	—	4.2	
Dual								
T_A = +25°C		—	4.9	6.0	—	4.9	6.0	
T_A = T_{low} to T_{high}		—	—	7.5	—	—	7.5	
Quad								
T_A = +25°C		—	9.7	11	—	9.7	11	
T_A = T_{low} to T_{high}		—	—	13	—	—	13	

NOTES (CONTINUED)

3. T_{low} = −55°C for MC35080,A	T_{low} = 0°C for MC34080,A	T_{high} = +125°C for MC35080,A	T_{high} = +70°C for MC34080,A
MC35081,A	MC34081,A	MC35081,A	MC34081,A
MC35082,A	MC34082,A	MC35082,A	MC34082,A
MC35083,A	MC34083,A	MC35083,A	MC34083,A
MC35084,A	MC34084,A	MC35084,A	MC34084,A
MC35085,A	MC34085,A	MC35085,A	MC34085,A

4. See application information for typical changes in input offset voltage due to solderability and temperature cycling

5. Limits at T_A = +25°C are guaranteed by high temperature (T_{high}) testing

AC ELECTRICAL CHARACTERISTICS (V_{CC} = +15 V, V_{EE} = −15 V, T_A = +25°C unless otherwise noted)

Characteristic	Symbol	A Suffix			Non-Suffix			Unit
		Min	Typ	Max	Min	Typ	Max	
Slew Rate (V_{in} = −10 V to +10 V, R_L = 2.0 k, C_L = 100 pF)	SR							V/µs
Compensated A_V = −1.0		20	25	—	20	25	—	
A_V = +1.0		—	30	—	—	30	—	
Decompensated A_V = −2.0		40	50	—	40	50	—	
A_V = +1.0		—	50	—	—	50	—	
Settling Time (10 V Step, A_V = −1.0)	t_s							µs
To 0.10% (+1/2 LSB of 9-Bits)		—	0.72	—	—	0.72	—	
To 0.01% (+1/2 LSB of 12-Bits)		—	1.6	—	—	1.6	—	
Gain Bandwidth Product (f = 200 kHz)	GBW							MHz
Compensated		6.0	8.0	—	6.0	8.0	—	
Decompensated		12	16	—	12	16	—	
Power Bandwidth (R_L = 2.0 k, V_O = 20 V_{p-p}, THD = 5.0%)	BWp							kHz
Compensated A_V = −1.0		—	400	—	—	400	—	
Decompensated A_V = −1.0		—	800	—	—	800	—	
Phase Margin (Compensated)	ϕm							Degrees
R_L = 2.0 k		—	55	—	—	55	—	
R_L = 2.0 k, C_L = 100 pF		—	39	—	—	39	—	
Gain Margin (Compensated)	A_m							dB
R_L = 2.0 k		—	7.6	—	—	7.6	—	
R_L = 2.0 k, C_L = 100 pF		—	4.5	—	—	4.5	—	
Equivalent Input Noise Voltage R_S = 100 Ω, f = 1.0 kHz	e_n	—	30	—	—	30	—	nV/√Hz
Equivalent Input Noise Current (f = 1.0 kHz)	i_n	—	0.01	—	—	0.01	—	pA/√Hz
Input Capacitance	C_i	—	5.0	—	—	5.0	—	pF
Input Resistance	r_i	—	10^{12}	—	—	10^{12}	—	Ω
Total Harmonic Distortion A_V = −10, R_L = 2.0 k, 2.0 ≤ V_O ≤ 20 V_{p-p}, f = 10 kHz	THD	—	0.05	—	—	0.05	—	%
Channel Separation (f = 10 kHz)	—	—	120	—	—	120	—	dB
Open-Loop Output Impedance (f = 1.0 MHz)	z_o	—	35	—	—	35	—	Ω

TYPICAL PERFORMANCE CURVES

FIGURE 1 — INPUT COMMON MODE VOLTAGE RANGE versus TEMPERATURE

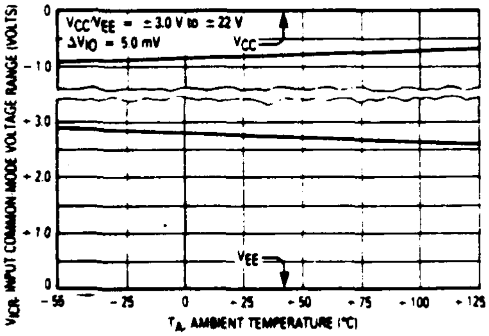


FIGURE 2 — INPUT BIAS CURRENT versus TEMPERATURE

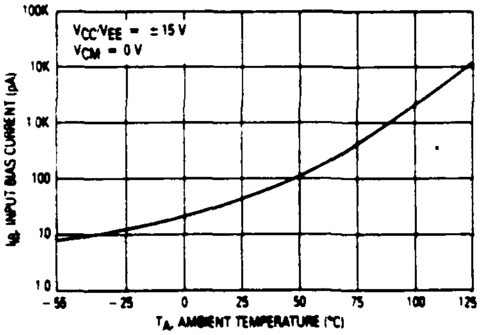


FIGURE 3 — INPUT BIAS CURRENT versus INPUT COMMON-MODE VOLTAGE

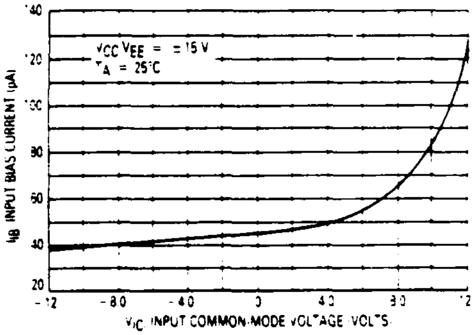


FIGURE 4 — OUTPUT VOLTAGE SWING versus SUPPLY VOLTAGE

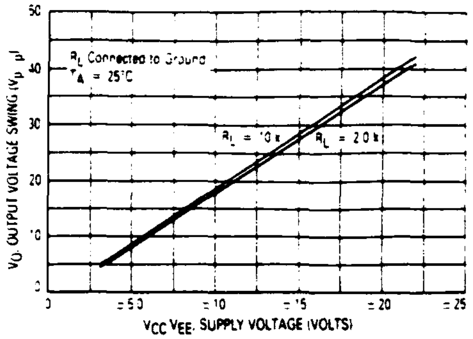


FIGURE 5 — OUTPUT SATURATION versus LOAD CURRENT

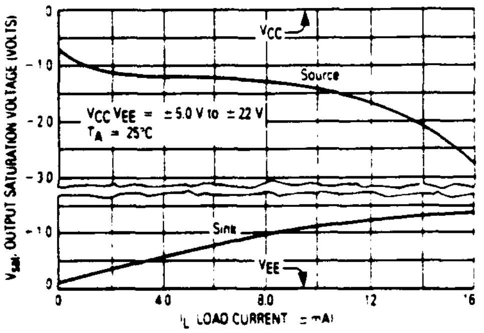


FIGURE 6 — OUTPUT SATURATION versus LOAD RESISTANCE TO GROUND

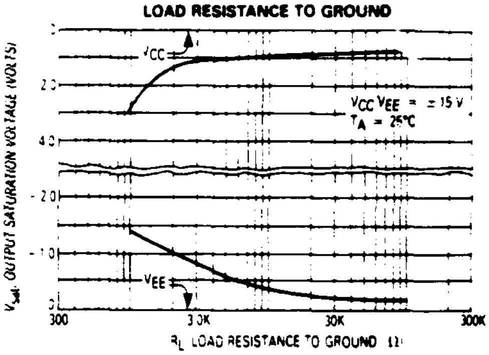


FIGURE 7 — OUTPUT SATURATION versus LOAD RESISTANCE TO VCC

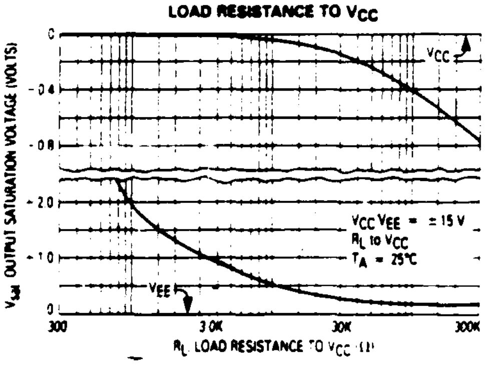


FIGURE 8 — OUTPUT SHORT CIRCUIT CURRENT versus TEMPERATURE

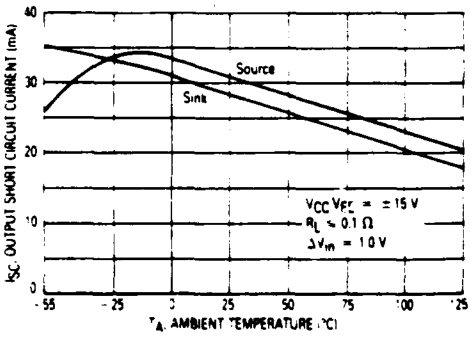


FIGURE 9 — OUTPUT IMPEDANCE versus FREQUENCY

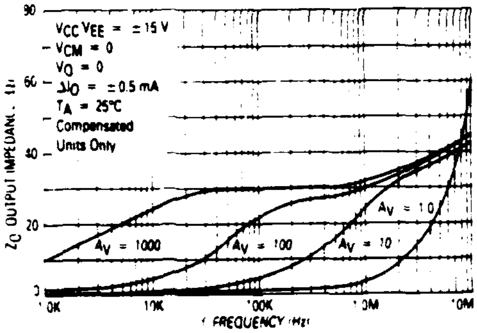


FIGURE 10 — OUTPUT IMPEDANCE versus FREQUENCY

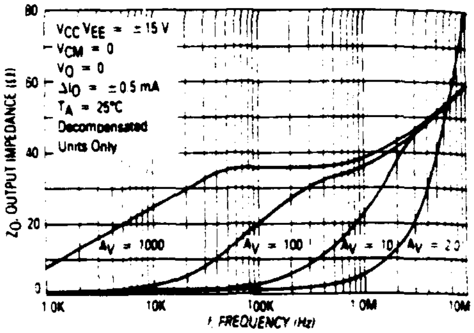


FIGURE 11 — OUTPUT VOLTAGE SWING versus FREQUENCY

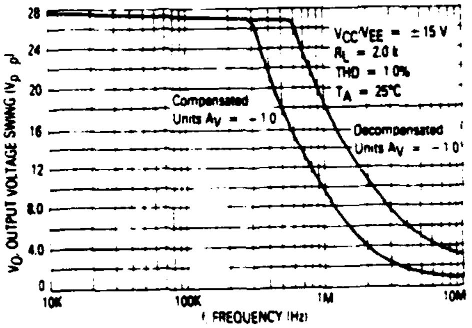


FIGURE 12 — OUTPUT DISTORTION versus FREQUENCY

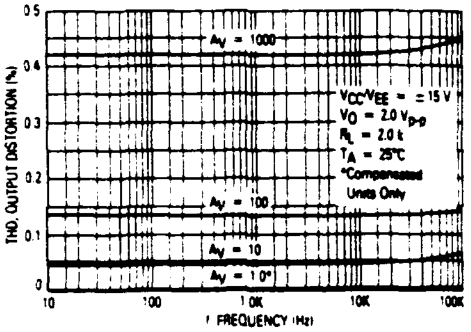


FIGURE 13 — OPEN-LOOP VOLTAGE GAIN versus TEMPERATURE

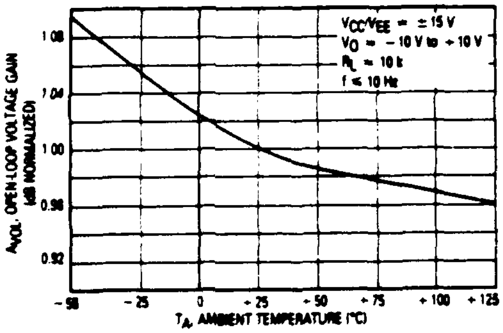


FIGURE 14 — OPEN-LOOP VOLTAGE GAIN AND PHASE versus FREQUENCY

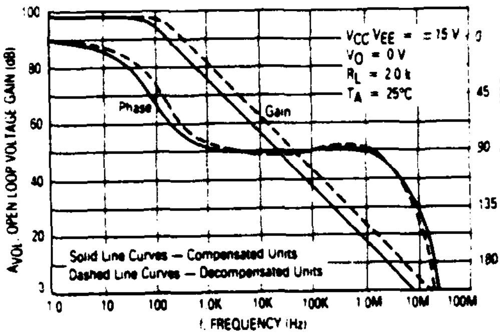


FIGURE 15 — OPEN-LOOP VOLTAGE GAIN AND PHASE versus FREQUENCY

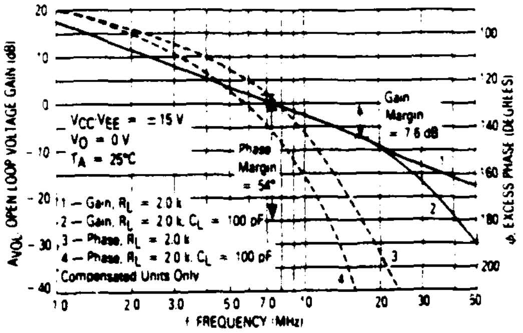


FIGURE 16 — OPEN-LOOP VOLTAGE GAIN AND PHASE versus FREQUENCY

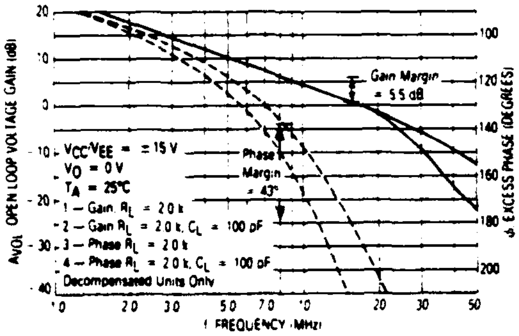


FIGURE 17 — NORMALIZED GAIN BANDWIDTH PRODUCT versus TEMPERATURE

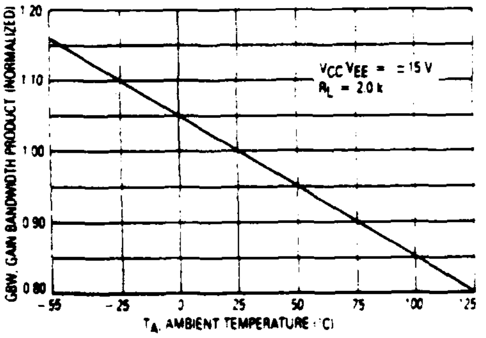


FIGURE 18 — PERCENT OVERSHOOT versus LOAD CAPACITANCE

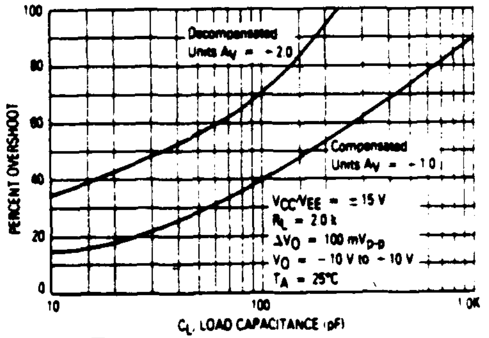


FIGURE 19 — PHASE MARGIN versus LOAD CAPACITANCE

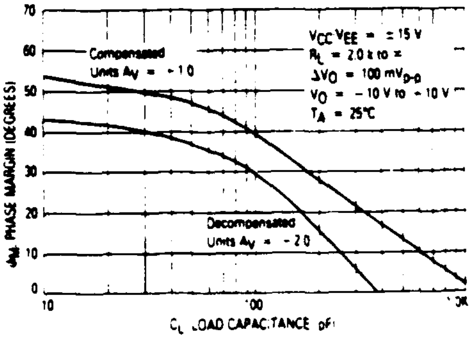


FIGURE 20 — GAIN MARGIN versus LOAD CAPACITANCE

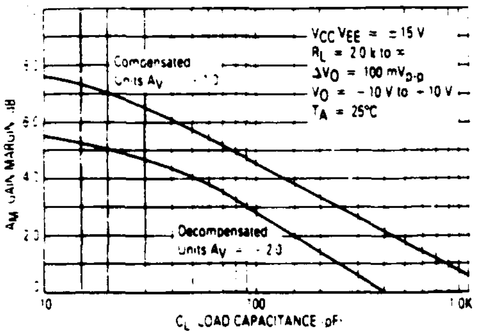


FIGURE 21 — PHASE MARGIN versus TEMPERATURE

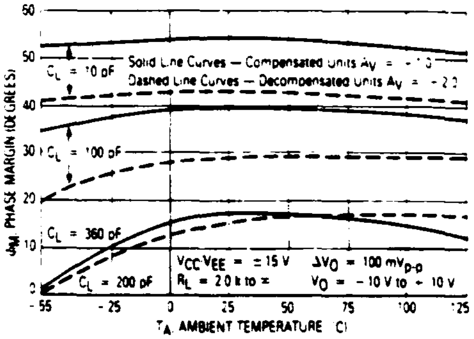


FIGURE 22 — GAIN MARGIN versus TEMPERATURE

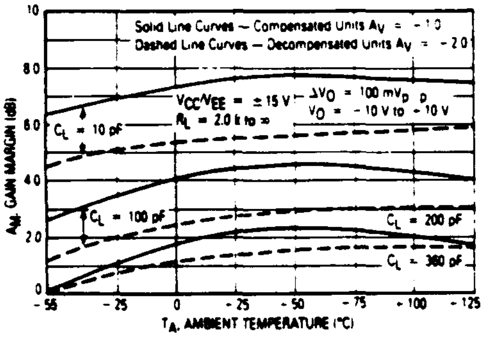
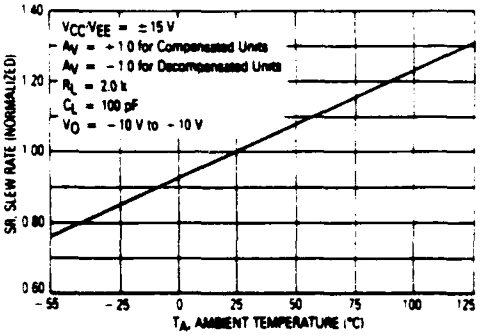


FIGURE 23 — NORMALIZED SLEW RATE versus TEMPERATURE



MC34084 TRANSIENT RESPONSE

A_V = +1.0, R_L = 2.0 k, V_{CC}/V_{EE} = ±15 V, T_A = 25°C

FIGURE 24 — SMALL-SIGNAL

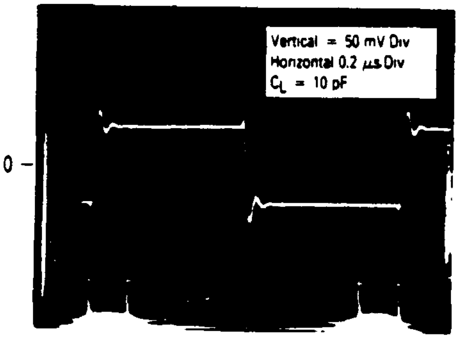
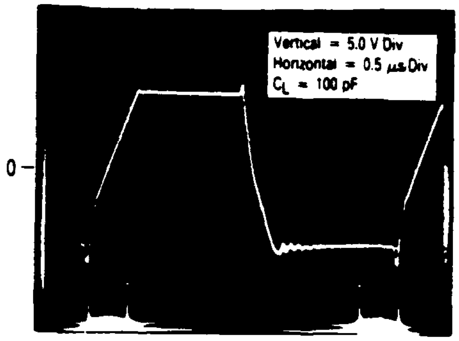


FIGURE 25 — LARGE-SIGNAL



MC34085 TRANSIENT RESPONSE

A_V = +2.0, R_L = 2.0 k, V_{CC}/V_{EE} = ±15 V, T_A = 25°C

FIGURE 26 — SMALL-SIGNAL

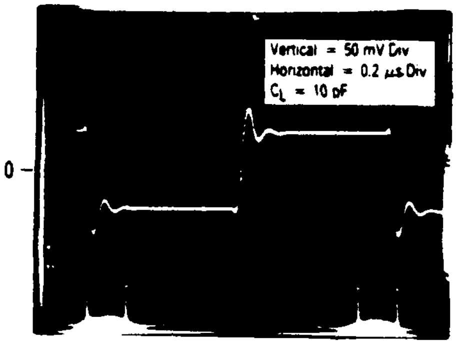


FIGURE 27 — LARGE-SIGNAL

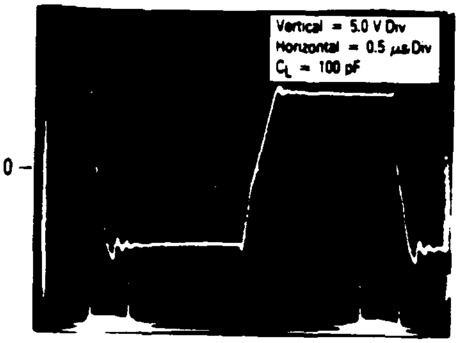


FIGURE 28 — COMMON-MODE REJECTION RATIO versus FREQUENCY

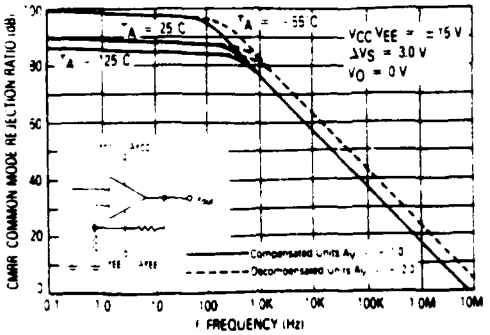


FIGURE 29 — POWER SUPPLY REJECTION RATIO versus FREQUENCY

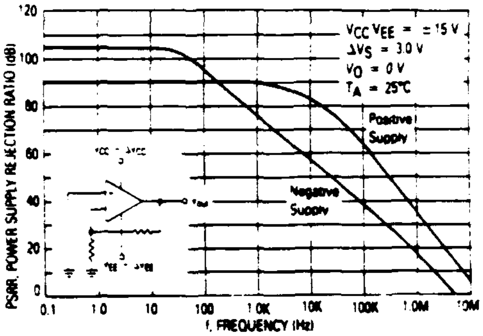


FIGURE 30 — POWER SUPPLY REJECTION RATIO versus TEMPERATURE

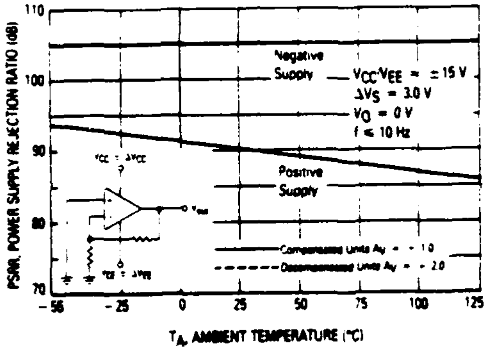


FIGURE 31 — NORMALIZED SUPPLY CURRENT versus SUPPLY VOLTAGE

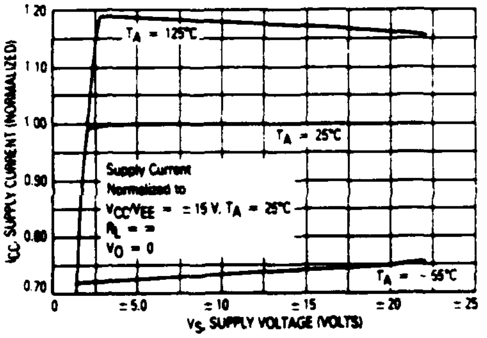


FIGURE 32 — CHANNEL SEPARATION versus FREQUENCY

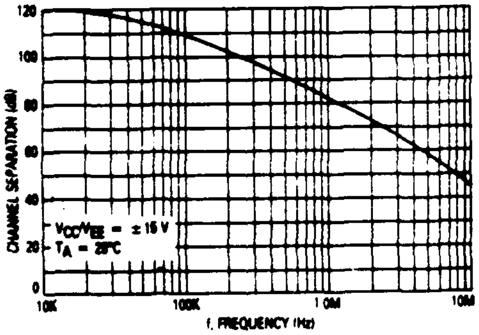
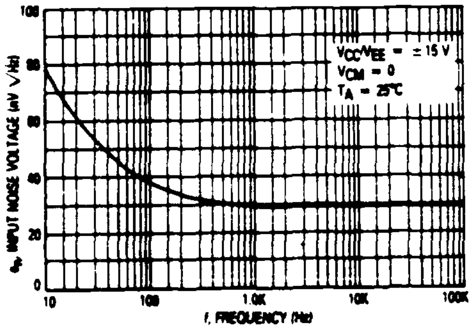


FIGURE 33 — SPECTRAL NOISE DENSITY



APPLICATIONS INFORMATION

The bandwidth and slew rate of the MC34080 series is nearly double that of currently available general purpose JFET op-amps. This improvement in ac performance is due to the P-channel JFET differential input stage driving a compensated miller integration amplifier in conjunction with an all NPN output stage.

The all NPN output stage offers unique advantages over the more conventional NPN PNP transistor Class AB output stage. With a 10 k load resistance, the op-amp can typically swing within 1.0 V of the positive rail (V_{CC}), and within 0.3 volts of the negative rail (V_{EE}), providing a 28.7 Vp − p swing from ±15 volt supplies. This large output swing becomes most noticeable at lower supply voltages. If the load resistance is referenced to V_{CC} instead of ground, the maximum possible output swing can be achieved for a given supply voltage. For light load currents, the load resistance will pull the output to V_{CC} during the positive swing and the NPN output transistor will pull the output very near V_{EE} during the negative swing. The load resistance value should be much less than that of the feedback resistance to maximize pull-up capability.

The all NPN transistor output stage is also inherently fast, contributing to the operational amplifier's high gain-bandwidth product and fast settling time. The associated high frequency output impedance is 50 ohms (typical) at 8.0 MHz. This allows driving capacitive loads from 0 to 300 pF without oscillations over the military temperature range, and over the full range of output swing. The 55° phase margin and 7.6 dB gain margin as well as the general gain and phase characteristics are virtually independent of the sink source output swing conditions. The high frequency characteristics of the MC34080 series is especially useful for active filter applications.

The common mode input range is from 2.0 volts below the positive rail (V_{CC}) to 4.0 volts above the negative rail (V_{EE}). The amplifier remains active if the inputs are biased at the positive rail. This may be useful for some applications in that single supply operation is possible with a single negative supply. However, a degradation of offset voltage and voltage gain may result.

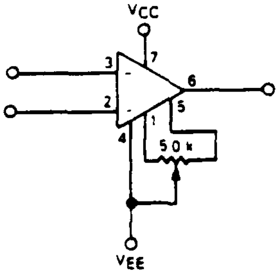
Phase reversal does not occur if either the inverting or noninverting input (or both) exceeds the positive common mode limit. If either input (or both) exceeds the negative common mode limit, the output will be in the high state. The input stage also allows a differential up to ±44 volts, provided the maximum input voltage range is not exceeded. The supply voltage operating range is from ±5.0 V to ±22 V.

For optimum frequency performance and stability careful component placement and printed circuit board layout should be exercised. For example, long unshielded input or output leads may result in unwanted input-output coupling. In order to reduce the input capacitance, resistors connected to the input pins should be physically close to these pins. This not only minimizes the input pole for optimum frequency response, but also minimizes extraneous "pickup" at this node.

Supply decoupling with adequate capacitance close to the supply pin is also important, particularly over temperature, since many types of decoupling capacitors exhibit large impedance changes over temperature.

Primarily due to the JFET inputs of the op amp, the input offset voltage may change due to temperature cycling and board soldering. After 20 temperature cycles (−55°C to 165°C), the typical standard deviation for input offset voltage is 559 µV and 473 µV in the plastic and ceramic packages respectively. With respect to board soldering (260°C, 10 seconds) the typical standard deviation for input offset voltage is 525 µV and 227 µV in the plastic and ceramic package respectively. Socketed plastic or ceramic packaged devices should be used over a minimal temperature range for optimum input offset voltage performance.

FIGURE 34 — OFFSET NULLING CIRCUIT



National Semiconductor Corporation

LM1877 Dual Power Audio Amplifier

General Description

The LM1877 is a monolithic dual power amplifier designed to deliver 2W/channel continuous into 8Ω loads. The LM1877 is designed to operate with a low number of external components, and still provide flexibility for use in stereo phonographs, tape recorders and AM-FM stereo receivers, etc. Each power amplifier is biased from a common internal regulator to provide high power supply rejection, and output Q point centering. The LM1877 is internally compensated for all gains greater than 10.

Features

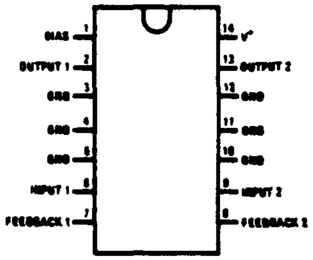
- 2W/channel
- −65 dB ripple rejection, output referred
- −65 dB channel separation, output referred
- Wide supply range, 6V–24V
- Very low cross-over distortion
- Low audio band noise
- AC short circuit protected
- Internal thermal shutdown

Applications

- Multi-channel audio systems
- Stereo phonographs
- Tape recorders and players
- AM-FM radio receivers
- Servo amplifiers
- Intercom systems
- Automotive products

Connection Diagram

Dual-In-Line Package

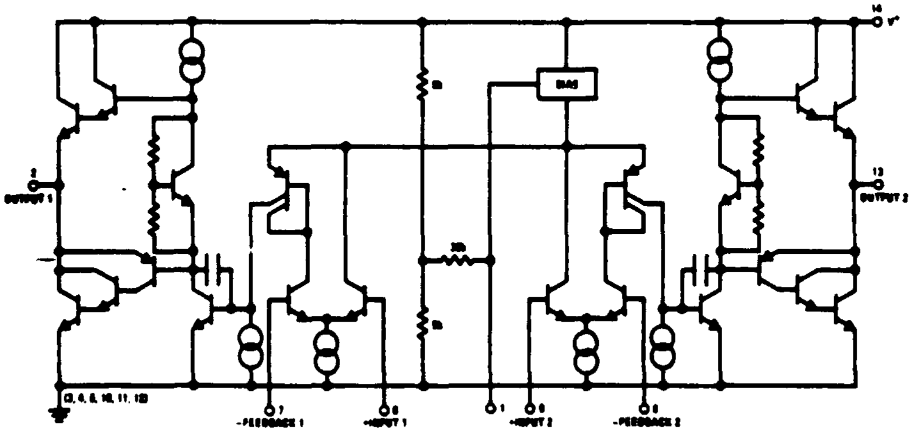


TL/H/7913-1

Top View

Order Number LM1877N-9
See NS Package Number N14A

Equivalent Schematic Diagram



TL/H/7913-2

Absolute Maximum Ratings

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

Supply Voltage	26V
Input Voltage	±0.7V
Operating Temperature	0°C to +70°C
Storage Temperature	−65°C to +150°C
Junction Temperature	150°C
Lead Temperature (Soldering, 10 sec.)	260°C

Electrical Characteristics

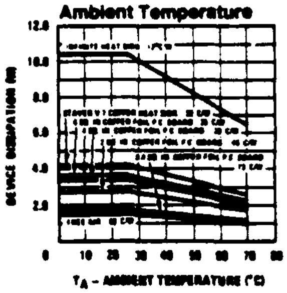
V_S = 20V, T_A = 25°C, (See Note 1) R_L = 8Ω, A_V = 50 (34 dB) unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Total Supply Current	P_O = 0W		25	50	mA
Output Power LM1877	THD = 10% V_S = 20V, R_L = 8Ω	2.0			W/Ch
Total Harmonic Distortion LM1877	f = 1 kHz, V_S = 14V				
	P_O = 50 mW/Channel		0.075		%
	P_O = 500 mW/Channel		0.045		%
	P_O = 1 W/Channel		0.055		%
Output Swing	R_L = 8Ω		V_S −6		Vp-p
Channel Separation	C_F = 50 μF, C_{IN} = 0.1 μF, f = 1 kHz, Output Referred				
	V_S = 20V, V_O = 4 Vrms	−50	−70		dB
	V_S = 7V, V_O = 0.5 Vrms		−60		dB
PSRR Power Supply Rejection Ratio	C_F = 50 μF, C_{IN} = 0.1 μF, f = 120 Hz, Output Referred				
	V_S = 20V, V_{RIPPLE} = 1 Vrms	−50	−65		dB
	V_S = 7V, V_{RIPPLE} = 0.5 Vrms		−40		dB
Noise	Equivalent Input Noise				
	R_S = 0, C_{IN} = 0.1 μF, BW = 20 Hz−20 kHz, Output Noise Wideband		2.5		μV
	R_S = 0, C_N = 0.1 μF, A_V 200		0.80		mV
Open Loop Gain	R_S = 0, f = 100 kHz, R_L = 8Ω		70		dB
Input Offset Voltage			15		mV
Input Bias Current			50		nA
Input Impedance	Open Loop		4		MΩ
DC Output Level	V_S = 20V	9	10	11	V
Slew Rate			2.0		V/μs
Power Bandwidth			65		kHz
Current Limit			1.0		A

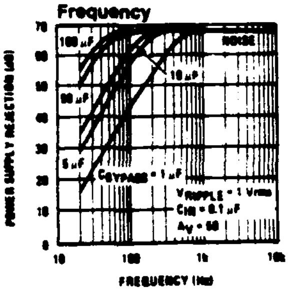
Note 1: For operation at ambient temperature greater than 25°C, the LM1877 must be derated based on a maximum 150°C junction temperature using a thermal resistance which depends upon device mounting techniques.

Typical Performance Characteristics

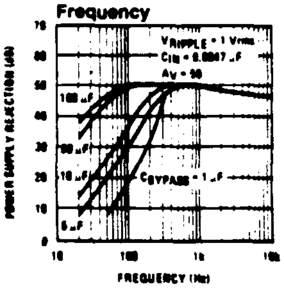
Device Dissipation vs Ambient Temperature



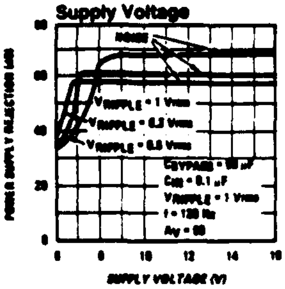
Power Supply Rejection Ratio (Referred to the Output) vs Frequency



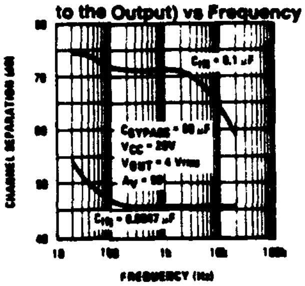
Power Supply Rejection Ratio (Referred to the Output) vs Frequency



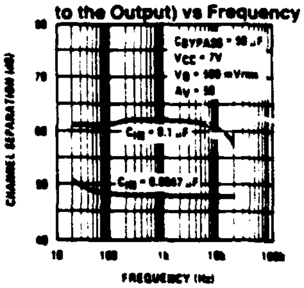
Power Supply Rejection Ratio (Referred to the Output) vs Supply Voltage



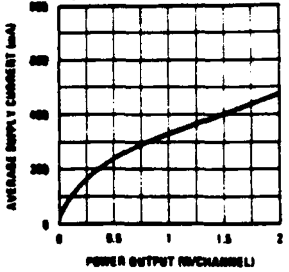
Channel Separation (Referred to the Output) vs Frequency



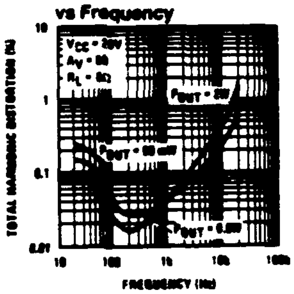
Channel Separation (Referred to the Output) vs Frequency



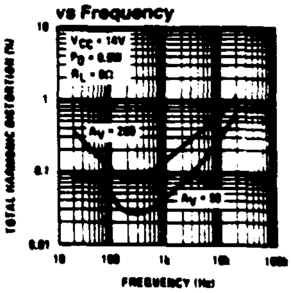
Average Supply Current vs P_{OUT}



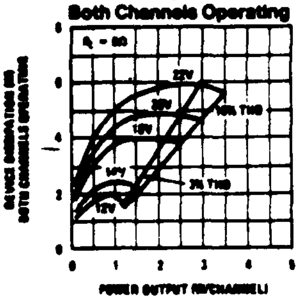
Total Harmonic Distortion vs Frequency



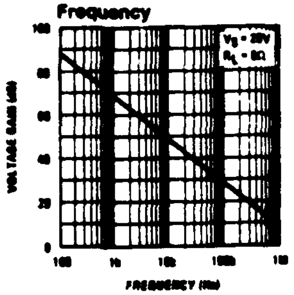
Total Harmonic Distortion vs Frequency



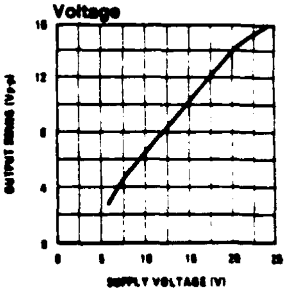
Power Dissipation (W) Both Channels Operating



Open Loop Gain vs Frequency



Output Swing vs Supply Voltage



TL/H/7913-3

PRELIMINARY INFORMATION

Commercial X9MME
Industrial X9MMEI

E²POT™ Digitally Controlled Potentiometer

FEATURES

- **Solid State Reliability**
- **Single Chip MOS Implementation**
- **Three Wire TTL Control**
- **Operates From Standard 5V Supply**
- **Wide Analog Voltage Range ±5V Min.**
- **99 Resistive Elements**
  - **—Temperature Compensated**
  - **—±20% End to End Resistance Range**
- **100 Wiper Tap Points**
  - **—Wiper Position Digitally Controlled**
  - **—Wiper Position Stored in Nonvolatile Memory Then Automatically Recalled on Power-Up**
- **100 Year Wiper Position Retention**
- **8 Pin Mini-DIP Package**

DESCRIPTION

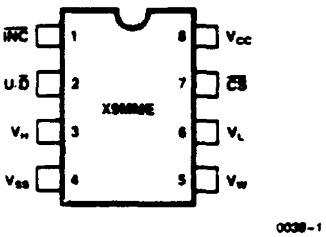
The Xicor X9MME is a solid state nonvolatile potentiometer, packaged in an 8 pin mini-DIP and is ideal for digitally controlled resistance trimming.

The X9MME is a resistor array composed of 99 resistive elements. Between each element and at either end are tap points accessible to the wiper element. The position of the wiper element on the array is controlled by the $\overline{CS}$, U/$\overline{D}$, and $\overline{INC}$ inputs. The position of the wiper can be stored in nonvolatile memory and is recalled upon a subsequent power-up.

The resolution of the X9MME is equal to the maximum resistance value divided by 99. As an example; for the X9503 (50 KΩ) each tap point represents 505Ω.

Xicor E² products are designed and tested for applications requiring extended endurance. Refer to Xicor reliability reports for further endurance information.

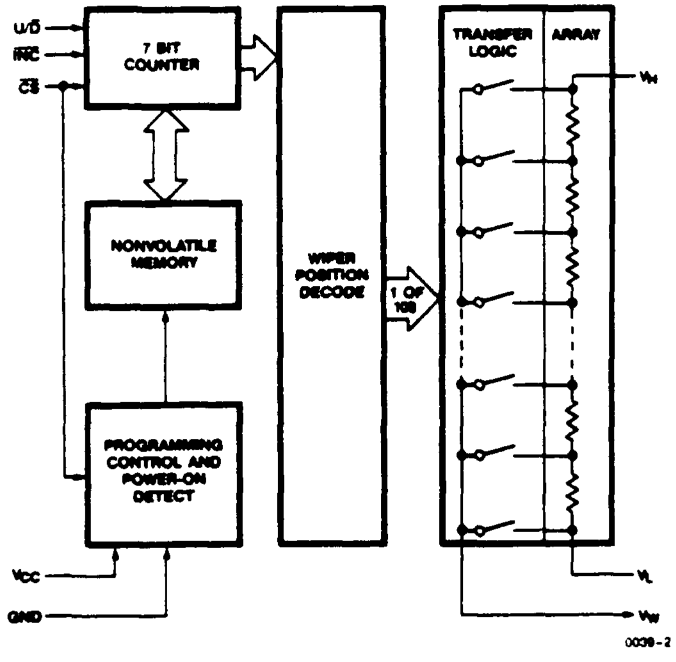
PIN CONFIGURATION



PIN NAMES

V_H	High Terminal of Pot
V_W	Wiper Terminal of Pot
V_L	Low Terminal of Pot
V_{SS}	Ground
V_{CC}	System Power
U/$\overline{D}$	Up/Down Control
$\overline{INC}$	Wiper Movement Control
$\overline{CS}$	Chip Select for Wiper Movement/Storage

FUNCTIONAL DIAGRAM



ANALOG CHARACTERISTICS

Electrical Characteristics

End to End Resistance Tolerance	±20%
Power Rating at 25°C	10 mW
Wiper Current	±1 mA Max.
Typical Wiper Resistance	40Ω at 1 mA
Typical Noise	< −120 dB/√Hz Ref: 1V

Resolution

Resistance	1%

Linearity

Absolute Linearity(1)	±1.0 MI(2)
Relative Linearity(3)	±0.2 MI(2)

Temperature Coefficient

−40°C to +85°C	±300 ppm/°C Typical

Wiper Adjustability

Unlimited Wiper Adjustment
(Volatile Mode While Chip is Selected)

Nonvolatile Storage of Wiper Position	10,000 Cycles Typical

Environmental Characteristics

Temperature Range

Operating	X9MME	0°C to +70°C
	X9MMEI	−40°C to +85°C
Storage		−65°C to +150°C

Physical Characteristics

Marking Includes:
Manufacturer's Trademark
Resistance Value or Code
Date Code

ABSOLUTE MAXIMUM RATINGS*

Temperature Under Bias	−65°C to +135°C
Storage Temperature	−65°C to +150°C
Voltage on $\overline{CS}$, $\overline{INC}$, U/$\overline{D}$ and V$_{CC}$ Referenced to Ground	−1.0V to +7.0V
Voltage on V$_H$ and V$_L$ Referenced to Ground	−8.0V to +8.0V
Lead Temperature (Soldering, 10 Seconds)	+300°C
Wiper Current	±1 mA

*COMMENT

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

D.C. OPERATING CHARACTERISTICS

X9MME T$_A$ = 0°C to +70°C, V$_{CC}$ = +5V ±10%, unless otherwise specified.
X9MMEI T$_A$ = −40°C to +85°C, V$_{CC}$ = +5V ±10%, unless otherwise specified.

Symbol	Parameter	Limits			Units	Test Conditions
		Min.	Typ.(4)	Max.		
I$_{CC}$	Supply Current		25	35	mA	
I$_{LI}$	Input Leakage Current			±10	μA	V$_{IN}$ = 0V to 5.5V, $\overline{INC}$, U/$\overline{D}$, $\overline{CS}$
V$_{IH}$	Input High Voltage	2.0		V$_{CC}$ + 1.0	V	
V$_{IL}$	Input Low Voltage	−1.0		0.8	V	
R$_W$	Wiper Resistance		40	100	Ω	±1 mA
V$_{VH}$	V$_H$ Voltage	−5.0		+5.0	V	
V$_{VL}$	V$_L$ Voltage	−5.0		+5.0	V	
C$_{IN}$(5)	$\overline{CS}$, $\overline{INC}$, U/$\overline{D}$, Input Capacitance			10	pF	

Notes: (1) Absolute Linearity is utilized to determine actual wiper voltage versus expected voltage as determined by wiper position when used as a potentiometer.

Absolute Linearity = (V$_{W(n)}$(actual) − V$_{W(n)}$(expected)) = ±1 MI Max.

(2) 1 MI = R$_{TOT}$/99 or $\frac{V_H - V_L}{99}$ = Minimum Increment.

(3) Relative Linearity is utilized to determine the actual change in voltage between successive tap position when used as a potentiometer. It is a measure of the error in step size.

Relative Linearity = V$_{W(n+1)}$ − [V$_{W(n)}$ + MI] = ±0.2 MI Max.

Typical values of Linearity are shown in Figure 3.

(4) Typical values are for T$_A$ = 25°C and nominal supply voltage.

(5) This parameter is periodically sampled and not 100% tested.

A.C. CONDITIONS OF TEST

Input Pulse Levels	0V to 3.0V
Input Rise and Fall Times	10 ns
Input	1.5V

MODE SELECTION

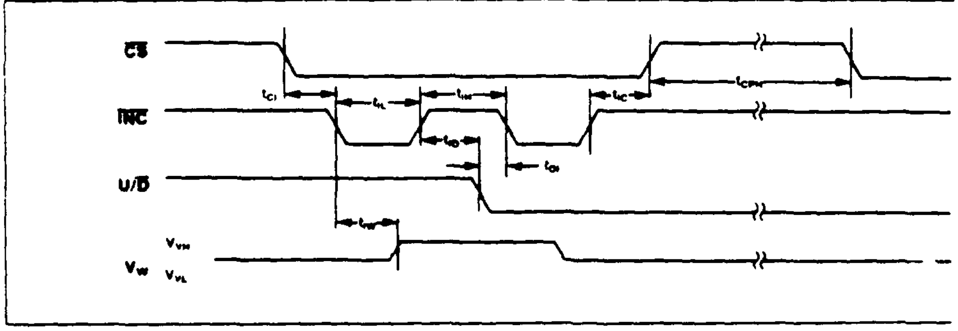
$\overline{CS}$	$\overline{INC}$	U/$\overline{D}$	M
L	↓	H	Wiper Up
L	↓	L	Wiper Do
↑	H	X	Store Wip

A.C. CHARACTERISTICS

X9MME T_A = 0°C to +70°C, V_{CC} = +5V ±10%, unless otherwise specified.
X9MMEI T_A = −40°C to +85°C, V_{CC} = +5V ±10%, unless otherwise specified.

Symbol	Parameter	Limits		
		Min.	Typ.(6)	Max.
t_{CI}	$\overline{CS}$ to $\overline{INC}$ Setup	100		
t_{ID}	$\overline{INC}$ High to U/$\overline{D}$ Change	100		
t_{DI}	U/$\overline{D}$ to $\overline{INC}$ Setup	2.9		
t_{IL}	$\overline{INC}$ Low Period	1		
t_{IH}	$\overline{INC}$ High Period	3		
t_{IC}	$\overline{INC}$ Inactive to $\overline{CS}$ Inactive	1		
t_{CPH}	$\overline{CS}$ Deselect Time	20		
t_{IW}	$\overline{INC}$ to V_W Change		100	500

A.C. Timing



Note: (6) Typical values are for T_A = 25°C and nominal supply voltage.

PIN DESCRIPTIONS

V_H

The high terminal of the X9MME is capable of handling an input voltage from −5V to +5V.

V_L

The low terminal input is limited from −5V to +5V.

V_W

The wiper terminal series resistance is typically less than 40Ω. The value of the wiper is controlled by the use of U/$\overline{\text{D}}$ and $\overline{\text{INC}}$.

Up/Down (U/$\overline{\text{D}}$)

The U/$\overline{\text{D}}$ input controls the direction of the wiper movement and the value of the nonvolatile counter.

Increment ($\overline{\text{INC}}$)

The $\overline{\text{INC}}$ input is negative-edge triggered. Toggling $\overline{\text{INC}}$ will move the wiper and either increment or decrement the counter in the direction indicated by the logic level on the U/$\overline{\text{D}}$ input.

Chip Select ($\overline{\text{CS}}$)

The device is selected when the $\overline{\text{CS}}$ input is LOW. The current counter value is stored in nonvolatile memory when $\overline{\text{CS}}$ is returned HIGH with $\overline{\text{INC}}$ HIGH.

DEVICE OPERATION

The $\overline{\text{INC}}$, U/$\overline{\text{D}}$ and $\overline{\text{CS}}$ inputs control the movement of the wiper along the resistor array. HIGH to LOW transitions on $\overline{\text{INC}}$, with $\overline{\text{CS}}$ LOW, increment (U/$\overline{\text{D}}$ = HIGH) or decrement (U/$\overline{\text{D}}$ = LOW) an internal counter. The output of the counter is decoded to position the wiper. When $\overline{\text{CS}}$ is brought HIGH the counter value is automatically stored in the nonvolatile memory. Upon power-up the nonvolatile memory contents are restored to the counter.

With the wiper at position 99, additional increments (U/$\overline{\text{D}}$ = HIGH) will not move the wiper. With the wiper at position 0, additional decrements (U/$\overline{\text{D}}$ = LOW) will not move the wiper.

The state of U/$\overline{\text{D}}$ may be changed while $\overline{\text{CS}}$ remains LOW, allowing a gross then fine adjustment during system calibration.

If V_{CC} is removed while $\overline{\text{CS}}$ is LOW the contents of the nonvolatile memory may be lost.

The end to end resistance of the array will fluctuate once V_{CC} is removed.

APPLICATIONS

The combination of a digital interface and nonvolatile memory in a silicon based trimmer pot provides many application opportunities that could not be addressed by either mechanical potentiometers or digital to analog circuits. The X9MME addresses and solves many issues that are of concern to designers of a wide range of equipment.

Consider the possibilities:

Automated assembly line calibration versus mechanical tweaking of potentiometers.

Protection against drift due to vibration or contamination.

Eliminate precise alignment of PWB mounted potentiometers with case access holes.

Eliminate unsightly access holes on otherwise aesthetically pleasing enclosures.

Product enhancements such as keyboard adjustment of volume or brightness control.

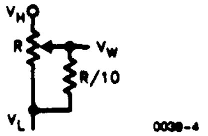
Front panel microprocessor controlled calibration of test instruments.

Remote location calibration via radio, modem or LAN link.

Calibration of hard to reach instruments in aircraft or other confined spaces.

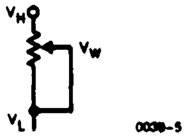
APPLICATION CIRCUITS

Application Circuit #1



Approximating audio trim with external resistor.

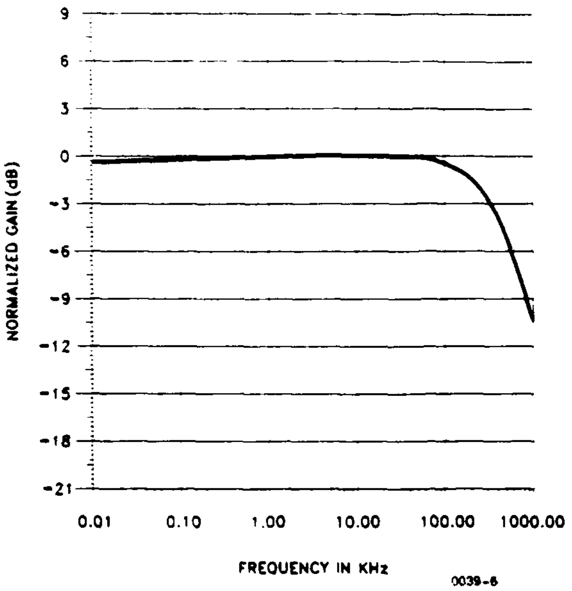
Application Circuit #2



Utilizing the X9MME as a variable resistor.

Note: Maximum Wiper Current = 1 mA.

Figure 1: Typical Frequency Response for X9103



TEST CONDITIONS

V_{CC} = 5.0V

Temp. = Room

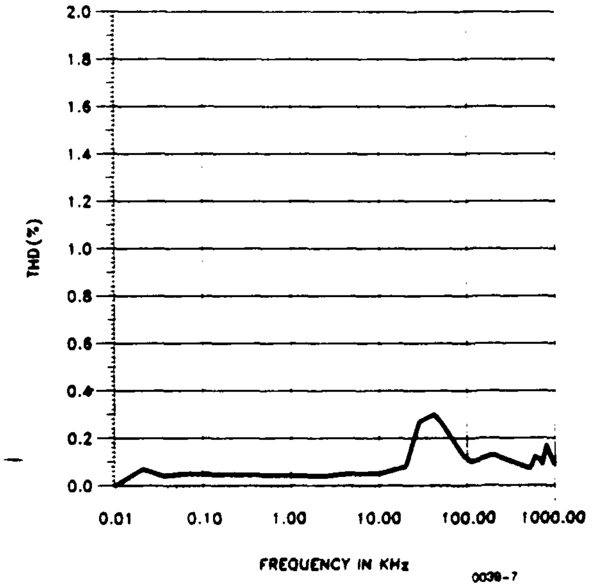
Wiper @ Tap 50

V_H = 0.5V RMS

Normalized (0 dB @ 1 KHz)

Test Circuit #1

Figure 2: Typical Total Harmonic Distortion for X9103



TEST CONDITIONS

V_{CC} = 5.0V

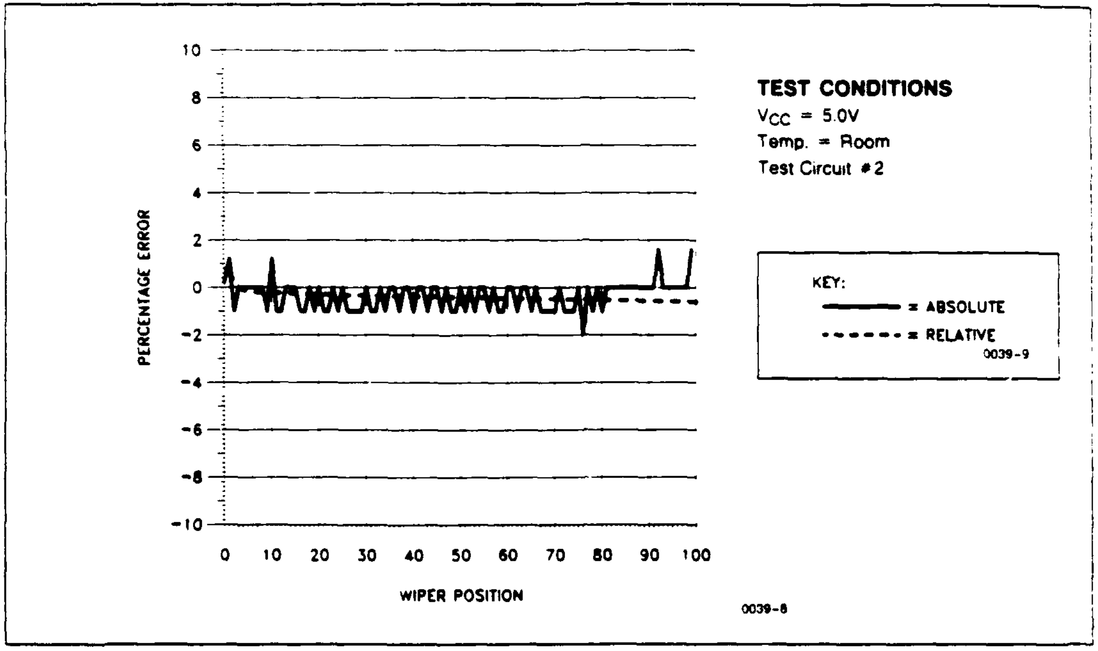
Temp. = Room

Wiper @ Tap 50

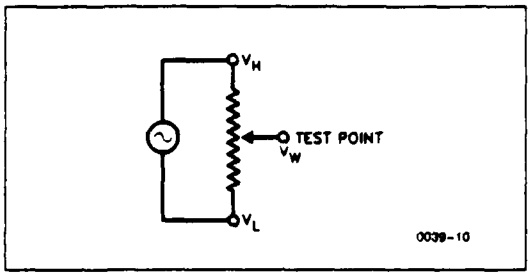
V_H = 2V RMS

Test Circuit #1

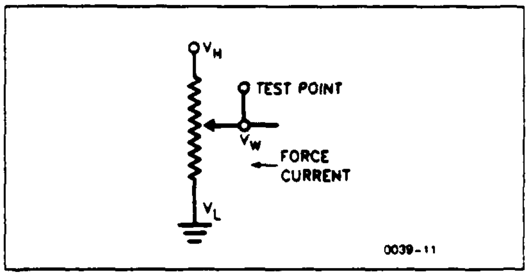
Figure 3: Typical Linearity for X9103



Test Circuit #1



Test Circuit #2



Standard Parts

Minimum Resistance	Wiper Increments	Maximum Resistance	Part Number
40Ω	101Ω	10 KΩ	X9103
40Ω	505Ω	50 KΩ	X9503
40Ω	1010Ω	100 KΩ	X9104

Appendix G - Parts list

AOI 2 Rev 3.0 Component List

The following is a list of components as of 10/9/89 for the 8930 Digital Board, Audio Board and Noise Board.

Audio Board:

R1 = 10 KΩ	R2 = 10 KΩ	R3 = 10 KΩ
R4 = 20 KΩ	R5 = 20 KΩ	R6 = 20 KΩ
R7 = 20 KΩ	R8 = 20 KΩ	R9 = 20 KΩ
R10 = 1 KΩ	R11 = 1 KΩ	R12 = 10 KΩ
R13 = 10 KΩ	R14 = 10 KΩ	R15 = 10 KΩ
R16 = 10 KΩ	R17 = 20 KΩ	R18 = 20 KΩ
R19 = 20 KΩ	R20 = 20 KΩ	R21 = 20 KΩ
R22 = 10 KΩ	R23 = 24.3 KΩ	R24 = 24.3 KΩ
R25 = 10 KΩ	R26 = 10 KΩ	R27 = 58.3 KΩ
R28 = 58.3 KΩ	R29 = 100 KΩ	R30 = 1 KΩ
R31 = 10 KΩ	R32 = 100 KΩ	R33 = 1KΩ
R34 = 10 KΩ	R35 = 1 KΩ	R36 = 1 KΩ
R37 = 10 KΩ	R38 = 10 KΩ	R39 = 1 KΩ
R40 = 1KΩ	R41 = 10 KΩ	R42 = 10 KΩ

R43 through R 62 = 100 KΩ

IC1 through IC4 = MC34084

IC5 through IC14 = Xicor 50 KΩ digitally controlled pot

IC15 = MC34082
IC16 = Xicor 50 KΩ digitally controlled Pot
IC17 = Xicor 50 KΩ digitally controlled Pot
IC19 = Xicor 50 KΩ digitally controlled Pot

P1 through P4 = 50 KΩ Pot

Noise Board:

R1 = 100 K	R2 = 16 K	R3 = 6.2 K
R4 = 510 K	R5 = 1 K	R6 = 2.7 K
R8 = 10 K	R9 = 27.1 K	R10 = 251.5 K
R11 = 251.5 K	R12 = 10 K	R13 = 10 K
R14 = 10 K	R15 = 10 K	R16 = 27.1 K
R17 = 25 K	R18 = 25 K	

C1 = 1 uF	C2 = 0.1 uF	C3 = 10 uF
C4 = 0.05 uF	C5 = 0.1 uF	C6 = 0.1 uF
C7 = 0.1 uF	C8 = 0.1 uF	C9 = 0.1 uF
C11 = 0.1 uF	C12 = 0.1 uF	

Pot1 = 50 K	Pot2 = 100 K

Component Count

8930 Digital Board:

Component	Number Needed
74xx688	2
65C22	1
AY8930	2
0.1 uF	5
NMI Connector	1
AOI Connector	1
Address select Connector	1

Power / Ground Plane Board:

Component	Number Needed
NMI Connector	1
AOI Connector	1
Ferrite Beads	3
10 μF Cap	3
1 μF Cap	3
.01 μF Cap	3
Power Connector	1

Audio Board:

Component	Number Needed
1 KΩ	7
10 KΩ	17
20 KΩ	11
24.3 KΩ	2
58.3 KΩ	2
100 kΩ	22
0.1 uF	17
MC34084	4
MC34082	1
50 KΩ Xicor Pot	13
50 KΩ Pot	4
LM1877	2
Audio Jacks	2
External Audio Connector	1
Noise Connector	1
AOI Connector	1

Noise Board:

Component	Number Needed
1 KΩ	1
2.7 KΩ	1

6.2 KΩ	1
10 KΩ	5
16 KΩ	1
25 KΩ	2
27.1 KΩ	2
100 KΩ	1
251.5 KΩ	2
510 KΩ	1
0.05 uF	1
0.1 uF	8
1 uF	1
10 uF	1
50 KΩ Pot	2
FLT-U2	2
LM389	1
Xicor 50K Pot	4
Noise Connector	1

Total List:

Component	Number Needed
1 KΩ	8
2.7 KΩ	1
6.2 KΩ	1
10 KΩ	22
16 KΩ	1
20 KΩ	11
24.3 KΩ	2
25 KΩ	2
27.1 KΩ	2
58.3KΩ	2
100 KΩ	23
251.5 KΩ	2
510 KΩ	1
0.05 uF	1
0.01 µF	3
0.1 uF	30
1 uF	4
10 uF	4
50 kΩ Pot	6
MC34084	4
MC34082	1
50 KΩ Xicor Pot	17
LM1877	2
FLT-U2	2
LM389	1
74xx688	2
65C22	1
AY8930	2
MNI Connector	2
AOI Connector	3
Address select connector	1

Audio Jacks	2
External Audio connector	1
Noise Connector	2
Power Connector	1

Appendix H - Source code listing

```
SEND SETUP
SEND ARRAYS
SEND CASE
SEND DOERMAKE
SEND CHIPCTRL
SEND SELECTOR
SEND VIEW8930
SEND ENVELOPE
SEND CONVERT
SEND TIME6522
SEND XICORS
SEND CONFIGUR
SEND TRANSIT
SEND ACIACOM
SEND SHOFRAME
SEND AOI
SEND SYSINITS
SEND DEMO
```

```
HEX
304 1C !
50 1E !
360 22 !
FORGET TASK  ( Reset )
380 DP !

DECIMAL

: OCTAL 8 BASE ! ;
: BINARY 2 BASE ! ;

VARIABLE WAIT.COUNT 5000 WAIT.COUNT !
: WAIT WAIT.COUNT @ 0 DO I DROP LOOP ;

: NIP SWAP DROP ;
: TUCK SWAP OVER ;
: -ROT ROT ROT ;
: INCR 1 SWAP +! ;
: DECR -1 SWAP +! ;
: 0SET 0 SWAP ! ;
: C0SET 0 SWAP C! ;
-1 CONSTANT TRUE
0 CONSTANT FALSE
: +- OVER OVER - -ROT + ;  ( n1 n2 - { n1-n2 } { n1 + n2 } )

: GET.OUT ?TERMINAL IF R> DROP EXIT THEN ;

: TASK ;

DECIMAL
```

```
DECIMAL

: CARRAY CREATE 3 + ALLOT DOES> + ;
: ARRAY CREATE 2+ 2* ALLOT DOES> SWAP 2*  + ;
: 2ARRAY  ( ROW COL -- ) CREATE 2 ALLOT 1+ SWAP  1+ SWAP DUP , * 2*
 ALLOT DOES> ROT OVER @ * ROT + 2* + 2+ ;
: DARRAY CREATE 1+ 4 * 2+ ALLOT DOES> SWAP 4 * + ;


: TABLE CREATE 2 ALLOT DOES> SWAP 2* + @ ;
( INITIALIZE BY COMMAING   )

: CTABLE CREATE 2 ALLOT DOES> + C@ ;
( INITIALIZE BY C-COMMAING  )

( NUMBERS TO BIT WEIGHTS )

TABLE 2^ 1 , 2 , 4 , 8 , 16 , 32 , 64 , 128 , 256 , 512 ,
         1024 , 2048 , 4096 , 8192 , 16384 , 32768 ,
( MODIFIED FROM W. BADEN )
( FORTH DIMENSIONS VOLUME 8, #5, P 31 )

DECIMAL

: CASE DUP ;
: OF [COMPILE] IF COMPILE DROP ;
IMMEDIATE
: ENDOF COMPILE EXIT [COMPILE] THEN ;
IMMEDIATE
: ENDCASE DROP ;
: =OR 2 PICK = OR ;
( N N1 N2 - F : T IF N >= N1 & <= N2 )
: BETWEEN 1+ OVER - >R - R> U< ;
: ASCII BL WORD COUNT 1- ABORT" ?? " C@ STATE @ IF [COMPILE] LITERAL THEN ;
IMMEDIATE
```

```
DECIMAL

: NOP ;

: DOER CREATE  2 ALLOT ['] NOP CFA ,  DOES>  @ >R  ;

( THE ALLOT STEPS OVER THE POINTER PUT IN THE  )
( FIRST WORD OF THE CHILD'S PARAMETER FIELD BY )
( DOES>                                        )
( DOER NAMEX  makes NAMEX vectored to NOP      )


: MAKE [COMPILE] ' @ 2+ [COMPILE] ' CFA SWAP ! ;

( MAKE NAMEX NEW-WORD revectors NAMEX          )
( MAKE NAMEX NOP revectors NAMEX to NOP        )
```

```
HEX

8000  CONSTANT 68HC11.PORT.A

3 2^  CONSTANT 68HC11.PORT.A.PULSE.BIT

DFF0  CONSTANT  AUDIO.6522.BASE

DECIMAL

AUDIO.6522.BASE  0  +  CONSTANT AUDIO.6522.PORT.B
AUDIO.6522.BASE 15  +  CONSTANT AUDIO.6522.PORT.A
AUDIO.6522.BASE  2  +  CONSTANT AUDIO.6522.PORT.B.DDR
AUDIO.6522.BASE  3  +  CONSTANT AUDIO.6522.PORT.A.DDR
AUDIO.6522.BASE  4  +  CONSTANT AUDIO.6522.T1C-L
AUDIO.6522.BASE  5  +  CONSTANT AUDIO.6522.T1C-H
AUDIO.6522.BASE  6  +  CONSTANT AUDIO.6522.T1L-L
AUDIO.6522.BASE  7  +  CONSTANT AUDIO.6522.T1L-H
AUDIO.6522.BASE  8  +  CONSTANT AUDIO.6522.T2C-L
AUDIO.6522.BASE  9  +  CONSTANT AUDIO.6522.T2C-H
AUDIO.6522.BASE 10  +  CONSTANT AUDIO.6522.SR
AUDIO.6522.BASE 11  +  CONSTANT AUDIO.6522.ACR
AUDIO.6522.BASE 12  +  CONSTANT AUDIO.6522.PCR
AUDIO.6522.BASE 13  +  CONSTANT AUDIO.6522.IFR
AUDIO.6522.BASE 14  +  CONSTANT AUDIO.6522.IER

DECIMAL

0  2^  CONSTANT SOUND.CHIP.SELECT.BIT   ( chip enable )
1  2^  CONSTANT BUS.CONTROL.BIT   ( BC1 for AY-8930s )
2  2^  CONSTANT BUS.DIRECTION.BIT ( BDIR for AY-8930s )

( bits 3 and 4 are used in XICORS )

5  2^  CONSTANT HOW.RESET.BOTH.AY-8930S.BIT

: P.ON      ( n port - )
     DUP C@   ( n port contents )
     ROT      ( port contents n )
     OR       ( port new.contents )
     SWAP C! ;

: P.OFF     ( n port - )
     DUP C@   ( n port contents )
     ROT      ( port contents n )
     OVER     ( port contents n contents )
     AND      ( port contents bits.in.n.currently.on )
     XOR      ( port new.contents )
     SWAP C! ;

HEX

: PULSE
     68HC11.PORT.A.PULSE.BIT   68HC11.PORT.A OVER OVER P.ON P.OFF ;

: INIT.AUDIO.6522.PORT.B
     FF  AUDIO.6522.PORT.B.DDR  C!
     20  AUDIO.6522.PORT.B C! ;       ( put reset bit high )

: HARDWARE.RESET.BOTH.AY-8930S
     HDW.RESET.BOTH.AY-8930S.BIT   AUDIO.6522.PORT.B   P.OFF
     HDW.RESET.BOTH.AY-8930S.BIT   AUDIO.6522.PORT.B   P.ON ;

: TO.SOUND
     FF AUDIO.6522.PORT.A.DDR C! ;

: FROM.SOUND
      0 AUDIO.6522.PORT.A.DDR C! ;

: SETUP.SOUND.CHIPS
    INIT.AUDIO.6522.PORT.B
    TO.SOUND
    HARDWARE.RESET.BOTH.AY-8930S  ;

: DNULL
    [ BUS.CONTROL.BIT BUS.DIRECTION.BIT OR ] LITERAL
    AUDIO.6522.PORT.B   P.OFF ;

: DLATCH
    [ BUS.CONTROL.BIT BUS.DIRECTION.BIT OR ] LITERAL
    AUDIO.6522.PORT.B   P.ON ;

: DREAD
    BUS.CONTROL.BIT  AUDIO.6522.PORT.B   P.ON ;

: DWRITE
    BUS.DIRECTION.BIT  AUDIO.6522.PORT.B  P.ON ;

VARIABLE SELECTED.SOUND.CHIP

: AY-8930.0
         SOUND.CHIP.SELECT.BIT AUDIO.6522.PORT.B  P.OFF
         0 SELECTED.SOUND.CHIP ! ;

: AY-8930.1
         SOUND.CHIP.SELECT.BIT  AUDIO.6522.PORT.B  P.ON
         1 SELECTED.SOUND.CHIP ! ;
```

```
: SET.SOUND.CHIP          ( 0 = AY-8930.0 | 1 = AY-8930.1 - )
       0= IF AY-8930.0 ELSE AY-8930.1 THEN ;


: RSET DNULL DLATCH AUDIO.6522.PORT.A C! DNULL ;  ( for AY-8930 addresses )

( Default AUDIO.6522.PORT.A bus direction is out to the peripherals )

: <GET> FROM.SOUND DNULL DREAD AUDIO.6522.PORT.A C@ DNULL   TO.SOUND ;

: <SEND> DNULL AUDIO.6522.PORT.A C! DWRITE DNULL ;

BINARY

10100000  CONSTANT BANK.A.CODE
10110000  CONSTANT BANK.B.CODE
00001111  CONSTANT CLEAR.NIBBLE.MASK

OCTAL

: BANK.A               ( - { bank A is current } )
   15 RSET
   <GET>
   CLEAR.NIBBLE.MASK   ( special case because bank code shares )
   AND                 ( a register with Channel A shape/cycle )
   BANK.A.CODE
   OR
   <SEND>
;



: BANK.B              ( { bank B is current } )
   15 RSET
   <GET>
   CLEAR.NIBBLE.MASK
   AND                ( special case like BANK.A )
   BANK.B.CODE
   OR
   <SEND>
;


DECIMAL
```

```
: SEND       ( n register #bytes bank - )
    0=  IF BANK.A  ELSE  BANK.B THEN
    1 = IF
          RSET
          <SEND>          ( transmits low order byte )
       ELSE
          DUP             ( n register register )
          RSET            ( n register )
          OVER            ( n register n )
          <SEND>
          1+              ( n register1+ )
          RSET
          ><              ( swap bytes )
          <SEND>          ( transmits low order byte --- former hi byte )
       THEN
;


: GET        ( register #bytes bank - )
    0=  IF BANK.A  ELSE  BANK.B THEN
    1 = IF
          RSET
          <GET>          ( get low order byte )
       ELSE
          DUP            ( register register )
          RSET
          <GET>          ( register low-byte )
          SWAP           ( low-byte register )
          1+
          RSET
          <GET>          ( low-byte high-byte
          256 * +        ( combined-low&high-bytes )
       THEN
;

DECIMAL
```

```
( SELECT BY REGISTER NAMES  )

DECIMAL

0 CONSTANT A
1 CONSTANT B
2 CONSTANT C

: TONE.PERIOD      (  channel - register #bytes bank  )
      2*           (  register  )
      2            (  register #bytes )
      A            (  register #bytes bank )
;

OCTAL

16 CONSTANT PORT.A.REGISTER

: AY-8930.PORT.A    ( channel - register #bytes bank )
      PORT.A.REGISTER
      1
      A
;

: AY-8930.PORT.B    ( channel - register #bytes bank )
      PORT.A.REGISTER 1+
      1
      A
;

6 CONSTANT NOISE.PERIOD.REGISTER

: NOISE.PERIOD      ( channel - register #bytes bank )
      NOISE.PERIOD.REGISTER        ( AY-8930 address )
      1
      A
;

7 CONSTANT NOT.ENABLE.REGISTER

: NOT.ENABLE        ( channel - register #bytes bank )
      NOT.ENABLE.REGISTER          ( AY-8930 address )
      1
      A
;

: AMPLITUDE         ( channel - register #bytes bank )
      10 +          ( base AY-8930 address )
      1
```

```
     A
;


: OR.ON        ( bits  )
    2 PICK
    2 PICK
    2 PICK
    GET       ( bits current.value ( register current ) )
    4 ROLL
    OR        ( add the new bits )
    3 ROLL
    3 ROLL
    3 ROLL
    SEND      ( store in register )
;

: XOR.OFF       ( bits )
    2 PICK
    2 PICK
    2 PICK
    GET       ( bits value )
    4 ROLL
    OVER      ( value bits value )
    AND       ( value "on" bits to turn off )
    XOR       ( turn them off )
    3 ROLL
    3 ROLL
    3 ROLL
    SEND      ( store in register )
 ;


: ENVELOPE.PERIOD   ( channel - register #bytes bank )
     CASE   0 = OF 13  2  A  ENDOF
     CASE   1 = OF  0  2  B  ENDOF
     CASE   2 = OF  2  2  B  ENDOF
     ABORT" CHANNEL RANGE ERROR"
     ENDCASE
;

: SHAPE/CYCLE        ( channel - register #bytes bank )
     CASE   0 = OF  15  1  A ENDOF
     CASE   1 = OF   4  1  B ENDOF
     CASE   2 = OF   5  1  B ENDOF
     ABORT" CHANNEL RANGE ERROR"
     ENDCASE
;
```

```
6 CONSTANT CH.A.DUTY.CYCLE.REGISTER

: DUTY.CYCLE    ( - register #bytes bank )
    CH.A.DUTY.CYCLE.REGISTER
    +
    1
    B
;

11 CONSTANT NOISE.AND.MASK.REGISTER

: NOISE.AND.MASK     ( - register #bytes bank )
      NOISE.AND.MASK.REGISTER
      1
      B
;

12 CONSTANT NOISE.OR.MASK.REGISTER

: NOISE.OR.MASK      ( - register #bytes bank )
      NOISE.OR.MASK.REGISTER
      1
      B
;

DECIMAL

1 CONSTANT HOLD.BIT
2 CONSTANT ALT.BIT
4 CONSTANT ATTACK.BIT
8 CONSTANT CONTINUE.BIT


( WARNING - THE NOT.ENABLE REGISTER IS NEGATIVE LOGIC )

: TONE.ENABLE   ( ch )
      2^ NOT.ENABLE XOR.OFF ;

: TONE.DISABLE   ( ch )
      2^ NOT.ENABLE OR.ON ;

: NOISE.ENABLE    ( ch )
     2^ 8 * NOT.ENABLE XOR.OFF ;

: NOISE.DISABLE   ( ch )
     2^ 8 * NOT.ENABLE OR.ON ;

: ZERO.AMPLITUDE   ( ch )
```

```
     0 SWAP AMPLITUDE SEND
;

: INIT.SOUND.CHIP
    255 NOT.ENABLE SEND  ( enables AY-8930 ports as outputs )
    0 A AMPLITUDE SEND
    0 B AMPLITUDE SEND
    0 C AMPLITUDE SEND
    0 A TONE.PERIOD SEND
    0 B TONE.PERIOD SEND
    0 C TONE.PERIOD SEND
    0 NOISE.PERIOD SEND
    0 A ENVELOPE.PERIOD SEND
    0 B ENVELOPE.PERIOD SEND
    0 C ENVELOPE.PERIOD SEND
    255 A SHAPE/CYCLE XOR.OFF
    255 B SHAPE/CYCLE XOR.OFF
    255 C SHAPE/CYCLE XOR.OFF
    4 A DUTY.CYCLE SEND   ( 50% DUTY CYCLE )
    4 B DUTY.CYCLE SEND   ( 50% DUTY CYCLE )
    4 C DUTY.CYCLE SEND   ( 50% DUTY CYCLE )
    255 NOISE.AND.MASK SEND
    0 NOISE.OR.MASK SEND
;
```

```
DECIMAL

: 16BITS ( d - )
   <# # # # # 32 HOLD # # # # 32 HOLD 32 HOLD
      # # # # 32 HOLD # # # # 32 HOLD 32 HOLD #> TYPE ;

: BITS.  ( n - n )
   BASE @ BINARY OVER 0 16BITS BASE ! ;

: HEX.
   BASE @ HEX OVER  8 U.R 2 SPACES BASE ! ;

: DEC.
   BASE @ DECIMAL OVER  8 U.R  2 SPACES BASE ! ;

: ALL. DEC. HEX. BITS. DROP ;   ( n - )

: SAY.CHANNEL  ( n - n )
    DUP
       CR ." CHANNEL   "
       CASE 0 = OF ." A  " ENDOF
       CASE 1 = OF ." B  " ENDOF
       CASE 2 = OF ." C  " ENDOF
       ENDCASE  ;

: SEE   ( ch )
    SAY.CHANNEL
    CR ."  Register          Decimal      Hex         Binary "
    CR ."  TONE PERIOD      "
    DUP  TONE.PERIOD GET ALL.
    CR ."  AMPLITUDE        "
    DUP AMPLITUDE GET ALL.
    CR ."  ENVELOPE PERIOD  "
    DUP ENVELOPE.PERIOD GET ALL.
    CR ."  SHAPE/CYCLE      "
    DUP  SHAPE/CYCLE GET ALL.
    CR ."  DUTY  CYCLE      "
         DUTY.CYCLE GET ALL.
    CR ."  ENABLE REGISTER  "
         NOT.ENABLE GET 255 XOR ALL.
    CR ."  NOISE PERIOD     "
    NOISE.PERIOD GET ALL.
    CR ."  NOISE AND MASK   "
    NOISE.AND.MASK GET ALL.
    CR ."  NOISE OR MASK    "
    NOISE.OR.MASK GET ALL.
    4 SPACES ;
```

```
( words to work the envelope registers )

: ENVELOPE.ON 32 SWAP AMPLITUDE SEND ;   ( ch - )
: ENVELOPE.OFF 0 SWAP AMPLITUDE SEND ;   ( ch - )

: ATTACK.ONCE     ( ch - )
           [ ATTACK.BIT ALT.BIT CONTINUE.BIT HOLD.BIT OR OR OR ]
           LITERAL SWAP SHAPE/CYCLE OR.ON ;

: DECAY.ONCE      ( ch - )
           [ ATTACK.BIT CONTINUE.BIT ALT.BIT HOLD.BIT OR OR OR ]
           LITERAL SWAP SHAPE/CYCLE XOR.OFF ;

: ATTACK.AND.HOLD    ( ch - )
           ALT.BIT OVER SHAPE/CYCLE XOR.OFF
           [ ATTACK.BIT CONTINUE.BIT HOLD.BIT OR OR ]
           LITERAL SWAP SHAPE/CYCLE OR.ON ;

: TRIANGLES    ( ch - )
           HOLD.BIT OVER SHAPE/CYCLE XOR.OFF
           [ ATTACK.BIT ALT.BIT CONTINUE.BIT OR OR ]
           LITERAL SWAP SHAPE/CYCLE OR.ON ;

: ATTACKS      ( ch - )
           [ ALT.BIT HOLD.BIT OR ] LITERAL OVER SHAPE/CYCLE XOR.OFF
           [ ATTACK.BIT CONTINUE.BIT OR ] LITERAL SWAP SHAPE/CYCLE OR.ON ;

: DECAYS       ( ch - )
           [ ATTACK.BIT ALT.BIT HOLD.BIT OR OR ]
           LITERAL OVER SHAPE/CYCLE XOR.OFF
           CONTINUE.BIT SWAP SHAPE/CYCLE OR.ON ;
```

```
DECIMAL

250000. 2CONSTANT CLOCK     ( AY-8930 clock freqeuncy divided by 8 )
( this is a empirical setting, the preliminary manual divides by )

: HZ CLOCK ROT UM/MOD NIP ;
(  2000 HZ leaves the TONE.PERIOD )
(  HZ can be used for tone or noise periods )
(  noise period is the input clock rate to the polynomial shift register )

: EHZ CLOCK 32 UM/MOD NIP SWAP / ;
( envelope period is the time for the 32 envelope steps )
( USES T1 AND T2 ON AUDIO.6522 )

: TASK ;

HEX

: INIT.TIMER.HOW
     E0 AUDIO.6522.ACR C!   ( T1 SQUARE WAVE, T2 COUNT DOWN )
     40 AUDIO.6522.PORT.B.DDR P.OFF  ( MAKE PB6 AN INPUT )
;

DECIMAL

50000  CONSTANT T1.50.MSEC
10000  CONSTANT T1.10.MSEC
 5000  CONSTANT T1.5.MSEC
 1000  CONSTANT T1.1.MSEC
  100  CONSTANT T1..1.MSEC
   10  CONSTANT T1..01.MSEC

VARIABLE  T1.INITIAL.COUNT
T1.1.MSEC T1.INITIAL.COUNT !

: ><!   ( n addr -  stored lowbyte in addr then hibyte in addr +1 )
      SWAP >< SWAP !
;

: @>< @ >< ;

: INIT.T1.COUNT
     T1.INITIAL.COUNT  @ AUDIO.6522.T1L-L ><!
     T1.INITIAL.COUNT  @ AUDIO.6522.T1C-L ><!
;

VARIABLE T2.INITIAL.COUNT
HEX FFFE T2.INITIAL.COUNT ! DECIMAL


: INIT.T2.COUNT
     T2.INITIAL.COUNT  @ AUDIO.6522.T2C-L ><!
;

: INIT.TIMER
      INIT.TIMER.HOW
      INIT.T1.COUNT
      INIT.T2.COUNT
;

: START.TIMER
```

```
    INIT.T2.COUNT
;

: READ.TIMER  ( - n )
     T2.INITIAL.COUNT @ AUDIO.6522.T2C-L @>< -
;


( WORDS FOR SCAN TIMING )

VARIABLE  T2.INTER.SCAN.COUNT
1000      T2.INTER.SCAN.COUNT !   ( default 1 second delay )

5 2^ CONSTANT  AUDIO.6522.T2.INTERRUPT.FLAG

: START.INTER.SCAN.INTERVAL
   T2.INTER.SCAN.COUNT @
   AUDIO.6522.T2C-L ><!
;

: ELAPSED.INTER.SCAN.INTERVAL
   T2.INTER.SCAN.COUNT @
   AUDIO.6522.T2C-L @>< -
;

: ?INTER.SCAN.INTERVAL.DONE
   AUDIO.6522.T2.INTERRUPT.FLAG
   AUDIO.6522.IFR C@ AND 0= NOT
;

: INIT.SYNCH
   INIT.TIMER.HOW
   INIT.T1.COUNT
   START.INTER.SCAN.INTERVAL
;

: SYNCH
   BEGIN
      ?INTER.SCAN.INTERVAL.DONE
   UNTIL
   START.INTER.SCAN.INTERVAL
;
```

```
( Audio Board Control for Xicor Digital Potentiometers )

: TASK ;

( Audio.6522.Port B )

3 2^ CONSTANT UP/DOWN.BIT     ( 1 IS UP )
4 2^ CONSTANT STEP.BIT

VARIABLE STEP.DIRECTION
       0 STEP.DIRECTION !

: STEP.UP
    UP/DOWN.BIT AUDIO.6522.PORT.B P.ON
    TRUE STEP.DIRECTION !
;

: STEP.DOWN
    UP/DOWN.BIT AUDIO.6522.PORT.B P.OFF
    FALSE STEP.DIRECTION !
;

( The order of the following constants will determine  )
( an array column, the bit weight and the output port   )
( AY-8930.1 Port A )

 0 CONSTANT PRE.MIXER.LEVEL
 1 CONSTANT NOISE.LEVEL
 2 CONSTANT BALANCE
 3 CONSTANT LEFT.AUDIO.LEVEL
 4 CONSTANT RIGHT.AUDIO.LEVEL
 5 CONSTANT NOT.USED
 6 CONSTANT HIGH.PASS.FILTER.1
 7 CONSTANT HIGH.PASS.FILTER.2

 6 CONSTANT FILTER
( ALIAS FOR HIGH.PASS.FILTER.1 )

( AY-8930.1 Port B )

 8 CONSTANT AY-8930.0.CHANNEL.A.LEVEL
 9 CONSTANT AY-8930.0.CHANNEL.B.LEVEL
10 CONSTANT AY-8930.0.CHANNEL.C.LEVEL
11 CONSTANT AY-8930.0.LEVEL
12 CONSTANT AY-8930.1.CHANNEL.A.LEVEL
13 CONSTANT AY-8930.1.CHANNEL.B.LEVEL
14 CONSTANT AY-8930.1.CHANNEL.C.LEVEL
```

```
15 CONSTANT AY-8930.1.LEVEL


0 CONSTANT XICOR.INITIAL.SETTING
1 CONSTANT XICOR.CURRENT.SETTING

1 15 2ARRAY XICOR.ARRAY

VARIABLE <STEP.TEMP>
       0 <STEP.TEMP> !

: STEP.TEMP  <STEP.TEMP> @ ; ( WORKS LIKE A CONSTANT )

: <SELECT>        ( uses STEP.TEMP )
     AY-8930.1
     STEP.TEMP 8 <
     IF STEP.TEMP 2^ AY-8930.PORT.A  ( get the bit weight directly )
     ELSE STEP.TEMP 8 - 2^ AY-8930.PORT.B ( shift down by 8 )
     THEN        ( - bit.weight port )
;

:  SELECT <SELECT> XOR.OFF ;   ( bit.weight port - )

: UNSELECT <SELECT> OR.ON ; ( bit.weight port - )

: STEP.SETTING  ( Xicor must be selected )
    STEP.BIT  AUDIO.6522.PORT.B P.OFF ( XICOR STEP IS ACTIVE LOW )
    XICOR.CURRENT.SETTING STEP.TEMP XICOR.ARRAY  ( addr )
    STEP.DIRECTION @ IF INCR ELSE DECR THEN
    STEP.BIT AUDIO.6522.PORT.B P.ON ( PUT IT BACK HIGH )
;

: GOTO.SETTING     ( target.setting Xicor.constant 0-15 )
    <STEP.TEMP> !  ( the key to this routine )
    DUP 0 100 BETWEEN ( target )
    IF
      DUP SELECT   ( target target - target target )
      XICOR.CURRENT.SETTING STEP.TEMP XICOR.ARRAY @
      -  ( target target current - target delta )
      0< IF STEP.DOWN ELSE STEP.UP THEN ( sels AY-8930.0)
      ( - target )
      BEGIN    ( target )
        DUP XICOR.CURRENT.SETTING STEP.TEMP XICOR.ARRAY @
        = NOT
      WHILE  STEP.SETTING
      REPEAT
    ELSE  ABORT" BAD ARGUMENT TO GOTO.SETTING"
    THEN ( target )
    DROP
    UNSELECT
;

: ALL.XICORS.TO.ZERO
    AY-8930.1
    255 AY-8930.PORT.A SEND
    255 AY-8930.PORT.B SEND
    ( ABOVE THREE LINES ARE REQUIRED BECAUSE SELECT IS )
    ( NEGATIVE LOGIC  -- A LOW SELECTS THE XICOR )
    STEP.DOWN
    16 0 DO
           I <STEP.TEMP> !
           SELECT
           100 0 DO STEP.SETTING LOOP
           0 XICOR.CURRENT.SETTING I XICOR.ARRAY !
           UNSELECT
         LOOP
;

: ALL.XICORS.TO.INITIAL.SETTING
     ALL.XICORS.TO.ZERO
     16 0 DO
            I <STEP.TEMP> !
            SELECT
            XICOR.INITIAL.SETTING I XICOR.ARRAY @
            I GOTO.SETTING
            UNSELECT
          LOOP
;

: SHOW.XICORS
CR
." XICOR        SETTINGS       INITIAL    CURRENT "
CR
." PRE.MIXER.LEVEL            "
XICOR.INITIAL.SETTING PRE.MIXER.LEVEL XICOR.ARRAY @ 10 U.R
XICOR.CURRENT.SETTING PRE.MIXER.LEVEL XICOR.ARRAY @ 10 U.R
CR
." NOISE.LEVEL                "
XICOR.INITIAL.SETTING  NOISE.LEVEL XICOR.ARRAY @ 10 U.R
XICOR.CURRENT.SETTING  NOISE.LEVEL XICOR.ARRAY @ 10 U.R
CR
." BALANCE                    "
XICOR.INITIAL.SETTING  BALANCE XICOR.ARRAY @ 10 U.R
XICOR.CURRENT.SETTING  BALANCE XICOR.ARRAY @ 10 U.R
CR
." LEFT.AUDIO.LEVEL           "
XICOR.INITIAL.SETTING  LEFT.AUDIO.LEVEL XICOR.ARRAY @ 10 U.R
XICOR.CURRENT.SETTING  LEFT.AUDIO.LEVEL XICOR.ARRAY @ 10 U.R
```

```
CR
." RIGHT.AUDIO.LEVEL        "
XICOR.INITIAL.SETTING RIGHT.AUDIO.LEVEL XICOR.ARRAY @ 10 U.R
XICOR.CURRENT.SETTING RIGHT.AUDIO.LEVEL XICOR.ARRAY @ 10 U.R
CR
." NOT.USED                 "
XICOR.INITIAL.SETTING  NOT.USED XICOR.ARRAY @ 10 U.R
XICOR.CURRENT.SETTING  NOT.USED XICOR.ARRAY @ 10 U.R
CR
." HIGH.PASS.FILTER.1       "
XICOR.INITIAL.SETTING HIGH.PASS.FILTER.1 XICOR.ARRAY @ 10
U.R
XICOR.CURRENT.SETTING HIGH.PASS.FILTER.1 XICOR.ARRAY @ 10
U.R
CR
." HIGH.PASS.FILTER.2       "
XICOR.INITIAL.SETTING  HIGH.PASS.FILTER.2 XICOR.ARRAY @ 10
U.R
XICOR.CURRENT.SETTING  HIGH.PASS.FILTER.2 XICOR.ARRAY @ 10
U.R
CR
." AY-8930.0.CHANNEL.A.LEVEL "
XICOR.INITIAL.SETTING  AY-8930.0.CHANNEL.A.LEVEL XICOR.ARRAY
@ 10 U.R
XICOR.CURRENT.SETTING  AY-8930.0.CHANNEL.A.LEVEL XICOR.ARRAY
@ 10 U.R
CR
." AY-8930.0.CHANNEL.B.LEVEL "
XICOR.INITIAL.SETTING  AY-8930.0.CHANNEL.B.LEVEL XICOR.ARRAY
@ 10 U.R
XICOR.CURRENT.SETTING  AY-8930.0.CHANNEL.B.LEVEL XICOR.ARRAY
@ 10 U.R
CR
." AY-8930.0.CHANNEL.C.LEVEL "
XICOR.INITIAL.SETTING  AY-8930.0.CHANNEL.C.LEVEL XICOR.ARRAY
@ 10 U.R
XICOR.CURRENT.SETTING  AY-8930.0.CHANNEL.C.LEVEL XICOR.ARRAY
@ 10 U.R
CR
." AY-8930.0.LEVEL          "
XICOR.INITIAL.SETTING  AY-8930.0.LEVEL XICOR.ARRAY @ 10 U.R
XICOR.CURRENT.SETTING  AY-8930.0.LEVEL XICOR.ARRAY @ 10 U.R
CR
." AY-8930.1.CHANNEL.A.LEVEL "
XICOR.INITIAL.SETTING  AY-8930.1.CHANNEL.A.LEVEL XICOR.ARRAY
@ 10 U.R
XICOR.CURRENT.SETTING  AY-8930.1.CHANNEL.A.LEVEL XICOR.ARRAY
@ 10 U.R
CR
```

```
." AY-8930.1.CHANNEL.B.LEVEL "
XICOR.INITIAL.SETTING  AY-8930.1.CHANNEL.B.LEVEL XICOR.ARRAY
@ 10 U.R
XICOR.CURRENT.SETTING  AY-8930.1.CHANNEL.B.LEVEL XICOR.ARRAY
@ 10 U.R
CR
." AY-8930.1.CHANNEL.C.LEVEL "
XICOR.INITIAL.SETTING  AY-8930.1.CHANNEL.C.LEVEL XICOR.ARRAY
@ 10 U.R
XICOR.CURRENT.SETTING  AY-8930.1.CHANNEL.C.LEVEL XICOR.ARRAY
@ 10 U.R
CR
." AY-8930.1.LEVEL           "
XICOR.INITIAL.SETTING  AY-8930.1.LEVEL XICOR.ARRAY @ 10 U.R
XICOR.CURRENT.SETTING  AY-8930.1.LEVEL XICOR.ARRAY @ 10 U.R
;

VARIABLE  DEFAULT.XICOR.SETTING
75        DEFAULT.XICOR.SETTING !

: DEFAULT.XICOR.INITIAL.SETTINGS
    16 0 DO
          DEFAULT.XICOR.SETTING @
          XICOR.INITIAL.SETTING I XICOR.ARRAY !
        LOOP
;

: SETUP.XICORS.TO.DEFAULT
    DEFAULT.XICOR.INITIAL.SETTINGS
    ALL.XICORS.TO.INITIAL.SETTING
;
```

```
( SOUND CHIPS MUST BE SET UP )

SETUP.SOUND.CHIPS
AY-8930.0
INIT.SOUND.CHIP
AY-8930.1
INIT.SOUND.CHIP
SETUP.XICORS.TO.DEFAULT
```

```
( TRANSITION BETWEEN NEW HARDWARE AND AOI SOFTWARE FOR TESTS )

( THIS IS WHERE THE DATA FROM THE FLIGHT INFORMATION PACAKAGE GOES )
( ROUTINE TO CAPTURE THE DATA SHOULD BE INSTALLED IN ONE OF THE HOOKS )
( IN AOI WITH A PAUSE TO WAIT FOR NEW DATA )

7 ARRAY INPUT.DATA

( THESE MAKE THE ARRAY ELEMENTS FUNCTION LIKE VARIABLES )

: AIRSPEED 0 INPUT.DATA ;
: ANGLE.OF.ATTACK  1 INPUT.DATA ;
: VERTICAL.VELOCITY 2 INPUT.DATA ;
: HEADING.ERROR 3 INPUT.DATA ;
: ROLL.ANGLE 4 INPUT.DATA ;
: PITCH.ANGLE 5 INPUT.DATA ;
: ALTITUDE 6 INPUT.DATA ;
: CHECKSUM 7 INPUT.DATA ;

( COMPATIBILITY DEFINITIONS )

: CENTER.BALANCE  50 BALANCE GOTO.SETTING ;

: GOTO.BALANCE BALANCE GOTO.SETTING ;   ( n - )
```

```
( DRIVER FOR NMI-5002 DUAL ACIA BOARD )

: TASK ;

HEX

A000 CONSTANT 6552.BASE
6552.BASE 4 + CONSTANT 6552.INT.REG.2
6552.BASE 5 + CONSTANT 6552.CTRL.2     ( WRITE )
6552.BASE 5 + CONSTANT 6552.STATUS.2   ( READ )
6552.BASE 7 + CONSTANT 6552.DATA.2


: INIT.ACIA.2
  0C  6552.CTRL.2 C!   ( 9600 baud )
  E1  6552.CTRL.2 C!   ( 8 bits odd parity not enabled RTS lo )
  7F  6552.INT.REG.2   ( turn off interrupts )
;

: RECEIVE.2  ( - n )
   BEGIN
     6552.INT.REG.2 C@ 1 AND
   UNTIL
     6552.DATA.2 C@
;

: XMIT.2    ( n - )
   6552.DATA.2 C!
   BEGIN
     6552.STATUS.2 C@
     40 AND
   UNTIL
;

DECIMAL

: RECEIVE.TEST
     BEGIN
        RECEIVE.2
        EMIT
     ?TERMINAL
     UNTIL
;

: XMIT.TEST
      BEGIN
        ?TERMINAL
        IF KEY
           XMIT.2
        THEN
      AGAIN
;

HEX
01 CONSTANT SOH
02 CONSTANT STX
03 CONSTANT ETX

DECIMAL

VARIABLE CALCULATED.CHECKSUM
VARIABLE TIMEOUT.COUNT

: GET.DATA.FRAME
    0 CALCULATED.CHECKSUM !
    STX XMIT.2
    0 TIMEOUT.COUNT !
    BEGIN
      TIMEOUT.COUNT @
      1+ DUP TIMEOUT.COUNT !
      50 =
      IF
        STX XMIT.2
        0 TIMEOUT.COUNT !
      THEN
      RECEIVE.2
      SOH =
    UNTIL
    14 0 DO
         RECEIVE.2
         DUP
           0 INPUT.DATA I + C!
           CALCULATED.CHECKSUM +!
         LOOP
     RECEIVE.2 0 INPUT.DATA 14 + C!
     RECEIVE.2 0 INPUT.DATA 15 + C!
     BEGIN
       RECEIVE.2
       ETX =
     UNTIL
;

: SHOW
  CR
  8 0 DO
    I INPUT.DATA @
    8 U.R
    LOOP
```

```
CALCULATED.CHECKSUM @
8 U.R
CR
;
```

```
( DISPLAYS INPUT.DATA ARRAY )
: TASK ;

: SHOW.DATA.FRAME
      CR
      8 0 DO I INPUT.DATA @ 8 U.R LOOP
      CALCULATED.CHECKSUM @ 8 U.R
      CR
;
```

```
: START.AOI ;

( PROTOTYPE AOI SOFTWARE )

DOER N.DO.AOA
DOER N.DO.AIRSPEED
DOER N.DO.ROLL
DOER N.DO.HEADING.ERROR
DOER N.DO.ALTITUDE
DOER N.DO.VERTICAL.VELOCITY
DOER HOOK1
DOER HOOK2
DOER HOOK3
DOER HOOK4
DOER N.STARTUP

: AOI  N.STARTUP
    BEGIN
       HOOK1
       HOOK2
       HOOK3
       HOOK4
       N.DO.AIRSPEED
       N.DO.VERTICAL.VELOCITY
       N.DO.ROLL
       N.DO.HEADING.ERROR
       N.DO.ALTITUDE
       N.DO.AOA
   0 UNTIL ;

: CENTER CENTER.BALANCE ;

( 0 IS LEFT 70 DEGREE ROLL )
( 2048 IS LEVEL )
( 4096 IS RIGHT 70 DEGREE ROLL )
( 29.25 COUNTS/DEGREE )

VARIABLE LEFT.ROLL.LIMIT.VALUE
1463 LEFT.ROLL.LIMIT.VALUE !
( 20 DEGREES LEFT LIMIT )

VARIABLE LEFT.ROLL.THRESHOLD
1901 LEFT.ROLL.THRESHOLD !
( SET TO 5 DEGREES LEFT )

VARIABLE ROLL.ZERO
2048 ROLL.ZERO !

VARIABLE RIGHT.ROLL.THRESHOLD
```

```
2194 RIGHT.ROLL.THRESHOLD !
( SET TO 5 DEGREES RIGHT )

VARIABLE RIGHT.ROLL.LIMIT.VALUE
2633 RIGHT.ROLL.LIMIT.VALUE !
( 20 DEGREES RIGHT LIMIT )

: DO.ROLL
       ROLL.ANGLE  @
         LEFT.ROLL.LIMIT.VALUE @ RIGHT.ROLL.LIMIT.VALUE @
       BETWEEN
       IF
          ROLL.ANGLE @
            LEFT.ROLL.THRESHOLD @ RIGHT.ROLL.THRESHOLD @
          BETWEEN
          IF  50 GOTO.BALANCE
          ELSE  ROLL.ANGLE @ ROLL.ZERO @  <
              IF LEFT.ROLL.THRESHOLD @ ROLL.ANGLE   @ -
                50
           LEFT.ROLL.THRESHOLD @ LEFT.ROLL.LIMIT.VALUE @ -
                */ 50 SWAP - GOTO.BALANCE
              ELSE
                ROLL.ANGLE @ RIGHT.ROLL.THRESHOLD @ -
                50
         RIGHT.ROLL.LIMIT.VALUE @ RIGHT.ROLL.THRESHOLD @  -
                */ 50 + GOTO.BALANCE
              THEN
          THEN
       ELSE ROLL.ANGLE @
         LEFT.ROLL.LIMIT.VALUE @ >
         IF 100 GOTO.BALANCE
         ELSE 0 GOTO.BALANCE
         THEN
       THEN
;

( 0 IS LEFT 180 DEGREE ERROR )
( 180 IS NO HEADING.ERROR )
( 359 IS RIGHT 179 DEGREE ERROR )

VARIABLE LEFT.HEADING.ERROR.LIMIT.VALUE
160 LEFT.HEADING.ERROR.LIMIT.VALUE !

VARIABLE LEFT.HEADING.ERROR.THRESHOLD
175 LEFT.HEADING.ERROR.THRESHOLD !

VARIABLE HEADING.ERROR.ZERO
180 HEADING.ERROR.ZERO !
```

```
VARIABLE RIGHT.HEADING.ERROR.THRESHOLD
185 RIGHT.HEADING.ERROR.THRESHOLD !

VARIABLE RIGHT.HEADING.ERROR.LIMIT.VALUE
200 RIGHT.HEADING.ERROR.LIMIT.VALUE !


: DO.HEADING.ERROR
       HEADING.ERROR  @
        RIGHT.HEADING.ERROR.LIMIT.VALUE @ LEFT.HEADING.ERROR.LIMIT.VALUE @
       BETWEEN
       IF
          HEADING.ERROR  @
           RIGHT.HEADING.ERROR.THRESHOLD @ LEFT.HEADING.ERROR.THRESHOLD @
          BETWEEN
          IF  50 GOTO.BALANCE
          ELSE  HEADING.ERROR  @ HEADING.ERROR.ZERO @  <
              IF RIGHT.HEADING.ERROR.THRESHOLD @ HEADING.ERROR   @ -
                50
           RIGHT.HEADING.ERROR.THRESHOLD @ RIGHT.HEADING.ERROR.LIMIT.VALUE @ -
                */ 50 SWAP - GOTO.BALANCE
              ELSE
                HEADING.ERROR  @ LEFT.HEADING.ERROR.THRESHOLD @ -
                50
         LEFT.HEADING.ERROR.LIMIT.VALUE @ LEFT.HEADING.ERROR.THRESHOLD @  -
                */ 50 + GOTO.BALANCE
              THEN
          THEN
       ELSE HEADING.ERROR  @
         LEFT.HEADING.ERROR.LIMIT.VALUE @ >
         IF 100 GOTO.BALANCE
         ELSE 0 GOTO.BALANCE
         THEN
       THEN
;



VARIABLE AIRSPEED.CHANNEL
C AIRSPEED.CHANNEL !

VARIABLE AIRSPEED.CHIP
0 AIRSPEED.CHIP !

VARIABLE AIRSPEED.AMPLITUDE
20 AIRSPEED.AMPLITUDE !

(    0 IS 0 KNOTS OF AIRSPEED )
( 4096 IS 330 KNOTS OF AIRSPEED )
( 12.41 COUNTS/KNOT )

VARIABLE LOW.AIRSPEED.LIMIT.VALUE
1116 LOW.AIRSPEED.LIMIT.VALUE !
( SET TO 90 KNOTS - STALL FOR THE QUEENAIRE )

VARIABLE HI.AIRSPEED.LIMIT.VALUE
2234  HI.AIRSPEED.LIMIT.VALUE !
( SET TO 180 KNOTS - TYPICAL FOR THE QUEENAIRE )

VARIABLE AIRSPEED.MAX.FREQ   ( Hz )
1000  AIRSPEED.MAX.FREQ !

VARIABLE AIRSPEED.MIN.FREQ   ( Hz )
40  AIRSPEED.MIN.FREQ !

VARIABLE AIRSPEED.AMPLITUDE.FLAG
FALSE AIRSPEED.AMPLITUDE.FLAG !

: SETUP.AIRSPEED
        AIRSPEED.CHIP @ SET.SOUND.CHIP
        AIRSPEED.CHANNEL @ TONE.ENABLE
;

: DO.AIRSPEED
        AIRSPEED.CHIP @ SET.SOUND.CHIP
        AIRSPEED  @  LOW.AIRSPEED.LIMIT.VALUE @ <
        IF FALSE AIRSPEED.AMPLITUDE.FLAG !  ( used by DO.VERTICAL.VELOCITY )
        ELSE
           AIRSPEED @ HI.AIRSPEED.LIMIT.VALUE @ >
           IF
             AIRSPEED.MAX.FREQ @
           ELSE
             AIRSPEED @ LOW.AIRSPEED.LIMIT.VALUE  @ - ( delta )
             AIRSPEED.MAX.FREQ @ AIRSPEED.MIN.FREQ @ -    ( freq. range )
             HI.AIRSPEED.LIMIT.VALUE @ LOW.AIRSPEED.LIMIT.VALUE  @ - ( used )
             */  AIRSPEED.MIN.FREQ @ +
           THEN
           HZ AIRSPEED.CHANNEL @ TONE.PERIOD SEND
           TRUE AIRSPEED.AMPLITUDE.FLAG !
        THEN
;

VARIABLE ATTACK.DECAY.FLAG
1 CONSTANT ADF.DECAY
2 CONSTANT ADF.ATTACK

( 0 IS DIVE 2048 FEET/MIN )
( 2048 IS LEVEL FLIGHT )
( 4096 IS CLIMB 2048 FEET/MIN )
( 1 FOOT/MINUTE/COUNT )

VARIABLE VERTICAL.VELOCITY.CHANNEL
C VERTICAL.VELOCITY.CHANNEL !

VARIABLE VERTICAL.VELOCITY.CHIP
0 VERTICAL.VELOCITY.CHIP !

VARIABLE CLIMB.2048.FT/MIN.VALUE
4096 CLIMB.2048.FT/MIN.VALUE !

VARIABLE CLIMB.THRESHOLD
2348 CLIMB.THRESHOLD !

VARIABLE VERTICAL.VELOCITY.ZERO
2048 VERTICAL.VELOCITY.ZERO !

VARIABLE DIVE.THRESHOLD
1748 DIVE.THRESHOLD !

VARIABLE DIVE.2048.FT/MIN.VALUE
0 DIVE.2048.FT/MIN.VALUE !

VARIABLE VERT.VEL.MIN.ENVELOPE.PERIOD
12000 VERT.VEL.MIN.ENVELOPE.PERIOD !

: SETUP.VERTICAL.VELOCITY
      0 ATTACK.DECAY.FLAG !
;

: DO.VERTICAL.VELOCITY
       VERTICAL.VELOCITY.CHIP @  SET.SOUND.CHIP
       VERTICAL.VELOCITY  @
       DIVE.2048.FT/MIN.VALUE @ CLIMB.2048.FT/MIN.VALUE @
       BETWEEN
       IF
           VERTICAL.VELOCITY  @
            DIVE.THRESHOLD @ CLIMB.THRESHOLD @
           BETWEEN
           IF
             0 ATTACK.DECAY.FLAG !
             AIRSPEED.AMPLITUDE.FLAG @
                 IF AIRSPEED.AMPLITUDE @
                    AIRSPEED.CHANNEL @  AMPLITUDE SEND
                 THEN
           ELSE
           VERTICAL.VELOCITY  @ VERTICAL.VELOCITY.ZERO @ >
```

```
              IF        ( climb )
                  CLIMB.2048.FT/MIN.VALUE @ VERTICAL.VELOCITY  @ -
                  32767 VERT.VEL.MIN.ENVELOPE.PERIOD @ 2/ - ( span )
                  CLIMB.2048.FT/MIN.VALUE @ CLIMB.THRESHOLD @ -
                  */ 2*
                  ATTACK.DECAY.FLAG @ ADF.DECAY = NOT
                  IF
                    VERTICAL.VELOCITY.CHANNEL @ DECAYS
                  THEN ADF.DECAY ATTACK.DECAY.FLAG !
              ELSE   ( dive )
                  VERTICAL.VELOCITY  @  DIVE.2048.FT/MIN.VALUE @ -
                  32767 VERT.VEL.MIN.ENVELOPE.PERIOD @ 2/ - ( span )
                  DIVE.THRESHOLD @ DIVE.2048.FT/MIN.VALUE @ -
                  */ 2*
                  ATTACK.DECAY.FLAG @ ADF.ATTACK = NOT
                  IF
                    VERTICAL.VELOCITY.CHANNEL @ ATTACKS
                  THEN ADF.ATTACK ATTACK.DECAY.FLAG !
              THEN
                  VERT.VEL.MIN.ENVELOPE.PERIOD @ +
                  VERTICAL.VELOCITY.CHANNEL @ ENVELOPE.PERIOD SEND
                  VERTICAL.VELOCITY.CHANNEL @ ENVELOPE.ON
            THEN
       ELSE
          VERTICAL.VELOCITY  @
          DIVE.2048.FT/MIN.VALUE @  <
             IF
                ATTACK.DECAY.FLAG @ ADF.ATTACK = NOT
                IF
                  VERTICAL.VELOCITY.CHANNEL @ ATTACKS
                THEN ADF.ATTACK ATTACK.DECAY.FLAG !
             ELSE
                ATTACK.DECAY.FLAG @ ADF.DECAY = NOT
                IF
                  VERTICAL.VELOCITY.CHANNEL @ DECAYS
                THEN ADF.DECAY ATTACK.DECAY.FLAG !
             THEN
             VERT.VEL.MIN.ENVELOPE.PERIOD @
             VERTICAL.VELOCITY.CHANNEL @ ENVELOPE.PERIOD SEND
             VERTICAL.VELOCITY.CHANNEL @ ENVELOPE.ON
       THEN
;

VARIABLE ANGLE.OF.ATTACK.TOLERANCE
200 ANGLE.OF.ATTACK.TOLERANCE !

VARIABLE ANGLE.OF.ATTACK.CHANNEL
8 ANGLE.OF.ATTACK.CHANNEL !
```

```
VARIABLE ANGLE.OF.ATTACK.CHIP
0 ANGLE.OF.ATTACK.CHIP !

VARIABLE ANGLE.OF.ATTACK.AMPLITUDE    ( 0 through 31 )
15 ANGLE.OF.ATTACK.AMPLITUDE !

VARIABLE ANGLE.OF.ATTACK.NEUTRAL.VALUE
4893 ANGLE.OF.ATTACK.NEUTRAL.VALUE !

VARIABLE ANGLE.OF.ATTACK.STALL.VALUE
6800 ANGLE.OF.ATTACK.STALL.VALUE !

: SETUP.AOA
     ANGLE.OF.ATTACK.CHIP @ SET.SOUND.CHIP
     ANGLE.OF.ATTACK.CHANNEL @ NOISE.ENABLE
;

: DO.AOA
       ANGLE.OF.ATTACK.CHIP  @  SET.SOUND.CHIP
       ANGLE.OF.ATTACK  @
       ANGLE.OF.ATTACK.NEUTRAL.VALUE @ ANGLE.OF.ATTACK.TOLERANCE @ +  <
       IF ANGLE.OF.ATTACK.CHANNEL @  ZERO.AMPLITUDE
       ELSE
         ANGLE.OF.ATTACK  @ ANGLE.OF.ATTACK.STALL.VALUE @ >
         IF
           1
         ELSE
           ANGLE.OF.ATTACK  @ ANGLE.OF.ATTACK.NEUTRAL.VALUE @
           ANGLE.OF.ATTACK.TOLERANCE @ + - ( delta )
           255
           ANGLE.OF.ATTACK.STALL.VALUE @
           ANGLE.OF.ATTACK.NEUTRAL.VALUE @ ANGLE.OF.ATTACK.TOLERANCE @ + -
           */ 255 SWAP -
         THEN
         NOISE.PERIOD SEND
         ANGLE.OF.ATTACK.AMPLITUDE @ ANGLE.OF.ATTACK.CHANNEL @ AMPLITUDE SEND
       THEN ;


VARIABLE ALTITUDE.CHANNEL
A ALTITUDE.CHANNEL !

VARIABLE ALTITUDE.CHIP
0 ALTITUDE.CHIP !

( 0 IS 2048 FEET BELOW FIP SETTING )
( 2048 IS AT FIP SETTING )
( 4096 IS 2048 FEET ABOVE FIP SETTING )
```

```
VARIABLE LOW.ALTITUDE.LIMIT.VALUE
1548 LOW.ALTITUDE.LIMIT.VALUE !

VARIABLE HIGH.ALTITUDE.LIMIT.VALUE
2548 HIGH.ALTITUDE.LIMIT.VALUE !

VARIABLE ALTITUDE.ENVELOPE.PERIOD
2000 ALTITUDE.ENVELOPE.PERIOD !

VARIABLE LOW.ALTITUDE.FREQ  ( Hz )
1500  HZ LOW.ALTITUDE.FREQ !

VARIABLE HIGH.ALTITUDE.FREQ  ( Hz )
3500  HZ HIGH.ALTITUDE.FREQ !

: SETUP.ALTITUDE
        ALTITUDE.CHANNEL @ TRIANGLES
        ALTITUDE.ENVELOPE.PERIOD @ ALTITUDE.CHANNEL @ ENVELOPE.PERIOD SEND
        ALTITUDE.CHANNEL @ TONE.ENABLE
;

: DO.ALTITUDE
      ALTITUDE.CHIP @  SET.SOUND.CHIP
      ALTITUDE  @
       LOW.ALTITUDE.LIMIT.VALUE @ HIGH.ALTITUDE.LIMIT.VALUE @
      BETWEEN
      IF ALTITUDE.CHANNEL @ ZERO.AMPLITUDE
      ELSE
        ALTITUDE  @ LOW.ALTITUDE.LIMIT.VALUE @ <
        IF LOW.ALTITUDE.FREQ @  ALTITUDE.CHANNEL @ TONE.PERIOD SEND
        ELSE HIGH.ALTITUDE.FREQ @ ALTITUDE.CHANNEL @ TONE.PERIOD SEND
        THEN
        ALTITUDE.CHANNEL @ ENVELOPE.ON
      THEN
;
```

```
( INITIALIZATIONS FOR THE AOI STARTUP ROUTINE )

DOER N.MORE.AOI.STARTUP.1
DOER N.MORE.AOI.STARTUP.2
DOER N.MORE.AOI.STARTUP.3



: SETUP
    SETUP.SOUND.CHIPS
    AY-8930.0
    INIT.SOUND.CHIP
    255 AY-8930.PORT.A SEND
    255 AY-8930.PORT.B SEND
    AY-8930.1
    INIT.SOUND.CHIP
    255 AY-8930.PORT.A SEND
    ALL.XICORS.TO.INITIAL.SETTING
    CENTER
    SETUP.AIRSPEED
    SETUP.VERTICAL.VELOCITY
    SETUP.AOA
    SETUP.ALTITUDE
    N.MORE.AOI.STARTUP.1
    N.MORE.AOI.STARTUP.2
    N.MORE.AOI.STARTUP.3
    INIT.ACIA.2
    INIT.SYNCH   ( INITIALIZE T1-T2 AUDIO.6522.TIMER FOR SYNCH )
    100 NOISE.LEVEL GOTO.SETTING
;
```

```
(  Hooks  vectors and dumps median to screen )
(  A to D connection removed )
(  SHOW.MEDIANS removed )

MAKE N.STARTUP SETUP  ( IN FILE SYSINITS )

MAKE HOOK1 SYNCH
( WAITS UNTIL 1 SECOND HAS ELAPSED SINCE TIMER STARTED )
( THE TIME FOR THE AOI PROCESSING IS WITHIN THE SECOND )

MAKE HOOK2 GET.DATA.FRAME
( PUTS A FRAME OF DATA INTO INPUT.DATA ARRAY )
( INPUT.DATA IS DEFINED IN FILE TRANSITS )
( GET.DATA.FRAME IS DEFINED IN ACIACOM )

MAKE HOOK3 SHOW.DATA.FRAME
( SHOWS THE DATA FRAME FROM INPUT.ARRAY )

MAKE HOOK4 GET.OUT
( GET.OUT BREAKS YOU OUT OF THE PROGRAM IF ANY KEY IS HIT )
( GET.OUT DEFINED IN FILE SETUP )

( MAKE N.DO.AOA                  DO.AOA )
( AOA VANE IS NOT YET ON AIRPLANE )

MAKE N.DO.AIRSPEED             DO.AIRSPEED

( BOTH DO.ROLL AND DO.HEADING.ERROR USE THE HEADPHONE BALANCE )
( THEREFORE ONLY ONE CAN BE USED AT A TIME )
MAKE N.DO.ROLL                 DO.ROLL
( MAKE N.DO.HEADING.ERROR      DO.HEADING.ERROR )

MAKE N.DO.ALTITUDE             DO.ALTITUDE

MAKE N.DO.VERTICAL.VELOCITY    DO.VERTICAL.VELOCITY
```

This is the command line used with Norton's LP program
to produce the HP Laserjet listings:

```
lp filespec /h60/w132/set:\aoi2\lasercod/l25/r25
```

The file lasercod contains the setup for the HP Laserjet
and consists of:

```
\027E\027(s16.66H
\027E\027(s16.66H
```