IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZDMH-A-D
PRODUCT NAME:	DMC11 HIGH SPEED JUMP AND FREE RUNNING TESTS
DATE:	JANUARY 1977
MAINTAINER:	DIAGNOSTICS
AUTHOR:	FAY BASHAW

1. ABSTRACT

The function of the DMC11 diagnostics is to verify that the option operates according to specifications. The diagnostics verfiy that there are no malfunctions and the all operations of the DMC11 are correct in its environment.

Parameters must be set up to alert the diagnostics to the DMC11 configuration. These parameters are contained in the STATUS TABLE and are generated in two ways: 1) Manual Input - the operator answers questions. 2) Autosizing - the program determines the parameters automatically.

DZDMH tests the DMC11-AL micro-processor (M8200-YB) with high speed crom, or the KMC11 micro-processor (M8204). It performs jump tests on the micro-processor, verifies the control ROM of the M8200-YB, and tests the CRAM and other unique functions of the M8204. If a DMC11-AL (M8200-YB) and line unit (M8202-YA or M8202-YD) are present, free-running tests are performed. These tests are skipped if a KMC (M8204) or no line-unit is present. The best test is with a line-unit installed. DZDMH can be used as a Heat Test Diagnostic by Manufacturing.

Currently there are four off line diagnostics that are to be run in sequence to insure that if an error should occur it will be detected at an early stage.

NOTE: Additional diagnostics may be added in the future.

The four diagnostics are:

1. DZDMC [REV] Basic W/R and Micro-processor tests
2. DZDME [REV] DDCMP Line unit tests
3. DZDMF [REV] BITSTUFF Line unit tests
4. DZDMG [REV] Low speed jump and Free-running tests (Heat test tape) NOTE: DZDMG IS RUN ONLY ON A DMC11-AR (M8200-YA).
   DZDMH [REV] High speed jump and Free-running tests (Heat test tape) NOTE: DZDMH IS RUN ONLY ON A DMC11-AL (M8200-YB).

2. REQUIREMENTS

2.1 EQUIPMENT

Any PDP11 family CPU (except an LSI-11) with minimum 8k memory
ASR 33 (or equilivalent)
DMC11-AL (M8200-YB) or an KMC11-A (M8204) with a DMC11-MA or a DMC11-MD

2.2 STORAGE

Program will use all 8K of memory except where ABL and BOOTSTRAP LOADER reside. Locations 1500 thru 1640; contain the "STATUS TABLE" information which is generated at start of diagnostics by manual input (questions) or automatically (auto-sizing). This area is an overlay area and should not be altered by the operator.

3. LOADING PROCEEDURE

3.1 METHOD

All programs are in absolute format and are loaded using the ABSOLUTE LOADER. NOTE; if the diagnostics are on a media such as DISK ,MAGTAPE,DECTAPE, or CASSETTE; follow instructions for the monitor which has been provided on that specific media.

ABSOLUTE LOADER starting address *500

MEMORY	* SIZE
4k	17
8k	37
12k	57
16k	77
20k	117
24k	137
28k	157

3.1.1 Place address of ABS loader into switch register.
(also place 'HALT' SW up)

3.1.2 Depress 'LOAD ADDRESS' key on console and release.

3.1.3 Depress 'START KEY' on console and release (program should now be loading into CPU)

4. STARTING PROCEEDURE

a. Set switch register to 000200
b. Depress 'LOAD ADDRESS' key and release
c. Set SWR to zero for 'AUTO SIZING' or SWR bit0=1 for manual input (questions) or SWR bit7=1 to use existing parameters set up by a previous start or a previously run DMC11 diagnostic.
d. Depress 'START KEY' and release. The program will type Maindec Name and program name (if this was the first start up of the program) and also the following:

MAP OF DMC11 STATUS

PC	CSR	STAT1	STAT2	STAT3
001500	160010	145310	177777	000000
001510	160020	145320	177777	000000

The program will type 'R' and proceed to run the diagnostic. The above is only an example. This would indicate the status table starting at add. 1500 in the program. In this example the table contains the information and status of two DMC11'S. THE STATUS TABLE MUST BE VERIFIED BY THE USER IF AUTO SIZING IS DONE. For information of status table see section 8.4 for help.

If the diagnostic was started with SW00=1 indicating manual parameter input then the following shows an example of the questions asked and some example answers:

```
HOW MANY DMC11'S TO BE TESTED?1

01
CSR ADDRESS?160010
VECTOR ADDRESS?310
BR PRIORITY LEVEL?  (4,5,6,7)?5
DOES MICRO-PROCESSOR HAVE CRAM?  (Y OR N)N
WHICH LINE UNIT?  IF NONE TYPE "N",  IF  M8201  TYPE  "1",  IF
M8202 TYPE "2"?1
IS THE LOOP BACK CONNECTOR ON?Y
SWITCH PAC#1 (DDCMP LINE#)?377
SWITCH PAC#2 (BM873 BOOT ADD)?377
```

Following the questions the status map is printed out as described above, the information in the map reflects the answers to the questions. If the diagnostic was started with SW00=0 and SW07=0 (AUTO-SIZING) then no questions are asked and only the status-map is printed out. If AUTO-SIZING is used the status information must be verified to be correct (match the hardware). if it does not match the hardware the diagnostic must be restarted with SW00=1 and the questions answered.

4.1 CONTROL SWITCH SETTINGS

SW 15 Set: Halt on error
SW 14 Set: Loop on current test
SW 13 Set: Inhibit error print out
SW 12 Set: Inhibit type out/abell on error.
SW 11 Set: Inhibit iterations. (quick pass)
SW 10 Set: Escape to next test on error
SW 09 Set: Loop with current data
SW 08 Set: Catch error and loop on it
SW 07 Set: Use previous status table.
SW 06 Set: Halt in ROMCLK routine before clocking micro-processor
SW 05 Set: Reserved
SW 04 Set: Reserved
SW 03 Set: Reselect DMC11's desired active
SW 02 Set: Lock on selected test
SW 01 Set: Restart program at selected test
SW 00 Set: Build new status table from questions. (If SW07=0 and SW00=0 a new status table is built by auto-sizing)

Switch 06 and 08-15 are dynamic and can be changed as needed while the diagnostic is running. Switches 00-03 and switch 07 are static, and are used only on starting or restarting the diagnostic.

4.1.2 SWITCH REGISTER OPTIONS (at start up)

SW 01 RESTART PROGRAM AT SELECTED TEST. It is stronaly suggested that at least one pass has been made before trying to select a test, the reason being is that the program has to clear areas and set up parameters. When this switch is used the diagnostic will ask TEST NO.? Answer by typing the number of the test desired and carrige return to begin execution at the selected test.

SW 02 LOCK ON SELECTED TEST. This switch when used with SW01 will cause the program to constantly loop on the selected test. Hitting any key on the console will let it advance to the next test and loop until a key is hit again. If SW02=0 when SW01 is used. The program will begin at the selected test and continue normal operations.

SW 03 RESELECT DMC11'S DESIRED ACTIVE. Please note that a message is typed out for setting the switch register equal to DMC11's active. this means if the system has four DMC11s; bits 00,01,02,03 will be set in loc 'DMACTV' from the switch register. Using this switch(SW00) alters that location;therefore if four DMC11s are in the system ***DO NOT*** set switchs greater than SW 03 in the up position. this would be a fatal error. do not select more active DMC11s than there is information on in the status table.

METHOD: A: Load address 200
B: Start with SW 00=1
C: Program will type message
D: Set a switch for each DMC desired active. EXAMPLE: If you have 4 DMC's but only want to run the first and the last set SWR bits 0 and 3 = 1. PRESS CONTINUE
E: Number (IF VALID) will be in data lights (excluding 11/05)
F: Set with any other switch settings desired. PRESS CONTINUE.

4.1.3 DYNAMIC SWITCHES

ERROR SWITCHES

1. SW 12 Delete print out/bell on error.
2. SW 13 Delete error printout.
3. SW 15 Halt on the error.
4. SW 08 Goto beginning of the test(on error).
5. SW 10 Goto next test(on error).

SCOPE SWITCHES

1. SW06 Halt in ROMCLK routine before clocking micro-processor instruction. This allows the operator to scope a micro-processor instruction in the static state before it is clocked. Hit continue to resume running.
2. SW09 (if enabled by 'SCOPI') on an error; If an '*' is printed in front of the test no. (ex. *TEST NO. 10) SW09 is incorporated in that test and therefore SW09 is usually the best switch for the scope loop (SW14=0, SW10=0, SW09=1, SW08=0). If SW09 is not enabeled; and there is a HARD error (constant); SW08 is best. (SW14=1,0, SW10=0, SW09=0, SW08=1). for intermittemt errors; SW14=1 will loop on test reguardless of error or not error. (SW14=1, SW10=0, SW09=0, SW08=1,0)
3. SW11 Inhibit interations.
4. SW14 Loop on current test.

4.2 STARTING ADDRESS

Starting address is at 000200 there are no other starting addresses for the DMC11 diagnostics. (See Section 4.0)

NOTE: If address 000042 is non-zero the program assumes it is under ACT11 or XXDP control and will act accordingly after all available DMC11's are tested the program will return to 'XXDP' or 'ACT-11'.

5. OPERATING PROCEDURE

When program is initially started messages as described in section 4.0 will be printed, and program will begin running the diagnostic

5.2 PROGRAM AND/OR OPERATOR ACTION

The typical approach should be

1. Halt on error (via SW 15=1) when ever an error occurs.
2. Clear SW 15.
3. Set SW 14: (loop on this test)
4. Set SW 13: (inhibit error print out)

The TEST NUMBER and PC will be typed out and possibily an error message (this depends on the test) to give the operator an idea as to the source of the problem. If it is necessary to know more information concerning the error report; LOOK IN THE LISTING for that TEST NUMBER which was typed out and then NOTE THE PC of the ERROR REPORT this way the EXACT FUNCTION of the test CAN BE DETERMINED.

6. ERRORS

As described previously there will always be a TEST NUMBER and PC typed out at the time of an error (providing SW 13=0 and SW 12=0), in most cases additional information will be supplied in the the error message to give the operator an indication of the error.

6.2 ERROR RECOVERY

If for some reason the DMC11 should 'HANG THE BUS' (gain control of bus so that console manual functions are inhibited) an init or power down/up is necessary for operator to regain control of cpu. If this should happen; look in location 'TSTNO' (address 1226)for the number of the test that was running at the time of the catastrophic error. In this way the operator will have an idea as to what the DMC11 was doing at the time of the error.

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

See section 4. (PLEASE)
Status table should be verified reguardless of how program was started. Also it is important to use this listing along with the information printed on the TTY to completly isolate problems.

7.2 OPERATING RESTRICTIONS

The first time a DMC11 diagnostic is loaded into core and run the STATUS TABLE must be set up. This is done by manual input (SW00=1) or by autosizing (SW00=0 and SW07=0). Thereafter however the status table need not be setup by subsequent restarts or even loading the next DMC diagnostic because the STATUS TABLE is overlayed. The current parameters in the STATUS TABLE are used when SW07=1 on start up.

7.3 HARDWARE CONFIGURATION RESTRICTIONS

DMC11(M8200)- Jumper W1 must be in, and switch 7 of E76 must be in the OFF position.

KMC(M8204)- Jumper W1 must be in.

LINE UNIT(M8201)- Jumpers W1, W2, and W4 must be IN. Jumpers W3, and W5 must be OUT. SW8 of E26 must be in the ON POSITION.

LINE UNIT (M8202)- Jumper W1 must be in. SW8 of E26 must be in the OFF position.

8. MISCELLANEOUS

8.1 EXECUTION TIME

All DMC11 device diagnostics will give an 'END PASS' message (providing no errors and sw12=0) within 4 mins. This is assuming SW11=1 (DELETE ITERATIONS) is set to give the fastest possible execution. The actual execution time depends greatly on the PDP11 CPU configuration and the amount of memory in the system.

8.2 PASS COMPLETE

NOTE: EVERY time the program is started; the tests will run as if SW11 (delete iterations) was up (=1). This is to 'VERIFY NO HARD ERRORS' as soon as possible. Therefore the first pass -EACH TIME PROGRAM IS STARTED- will be a 'QUICK PASS' until all DMC11's in system are tested. When the diagnostic has completed a pass the following is an example of the print out to be expected.

```
END PASS DZDMH CSR: 175000 VEC: 0300 PASSES: 000001
ERRORS: 000000
```

NOTE: The pass count and error counts are cummulitive for each DMC11 that is running, and are set to zero only when the diagnostic is started. Therefore after an overnight run for example, the total passes and errors for each DMC11 since the diagnostic was started are reflected in PASSES: and ERRORS:.

8.4 KEY LOCATIONS

RETURN (1214) Contains the address where program will return when iteration count is reached or if loop on test is asserted.

NEXT (1216) Contains the address of the next test to be peformed.

TSTNO (1226) Contains the number of the test now being peformed.

RUN (1316) The bit in 'RUN' always points to the DMC11 currently being tested. EXAMPLE: (RUN) 1302/0000000001000000 Means that DMC11 no.06 is the DMC11 now running.

DMCR00-DMCR17
DMST00-DMST17
(1500)-(1640)

These locations contain the information needed to test up to 16 (decimal) DMC11s sequentialy, they contain the CSR,VECTOR and STATUS concerning the configuration of each DMC11.

DMACTV (1306) Each bit set in this location indicates that the associated DMC11 will be tested in turn. EXAMPLE: (DMACTV) 1276/0000000000011111 means that DMC11 no. 00,01,02,03,04 will be tested. EXAMPLE: (DMACTV) 1276/0000000000010001 Means that DMC11 no. 00,04 will be tested.

DMCSR (1404) Contains the CSR of the current DMC11 under test.

8.4A 'STATUS TABLE' (1500-1640)

The table is filled by AUTO SIZING or by the manual parameter input (questions) as described previously. Also if desired by user; the locations may be altered by hand (toggled in) to suit the specific configuration.

The example status map shown below contains information for two DMC11'S. the table can contain up to 16 DMC11'S. Following the map is a description of the bits for each map entry

MAP OF DMC11 STATUS

PC	CSR	STAT1	STAT2	STAT3
001500	160010	145310	177777	000000
001510	160020	016320	000000	000000

Each map entry contains 4 words which contain the status information for 1 DMC11. The PC shows where in core memory the first of the 4 words is. In the example above the first DMC'S status is in locations, 1500, 1502, 1504, and 1506. The second DMC status is located at 1510, 1512, 1514, and 1516. The information contained in each 4 word entry is defined as follows:

CSR: Contains DMC11 CSR address

STAT1: BITS 00-08 IS DMC11 VECTOR ADDRESS
BIT15=1 MICRO-PROCESSOR HAS CRAM
BIT15=0 MICRO-PROCESSOR HAS CROM
BIT14=1 TURNAROUND CONNECTOR IS ON
BIT14=0 NO TURNAROUND CONNECTOR
BIT13=0 LINE UNIT IS AN M8201
BIT13=1 LINE UNIT IS AN M8202
BIT12=1 NO LINE UNIT
BITS 09-11 IS DMC11 BR PRIORITY LEVEL

STAT2: LOW BYTE IS SWITCH PAC#1 (DDCMP LINE NUMBER)
HIGH BYTE IS SWITCH PAC#2 (BM873 BOOT ADD)

STAT3: BIT0=1 PERFORM FREE RUNNING TESTS ON KMC
(must be set manually, SEE TEST 50)

8.5 METHOD OF AUTO SIZING

8.5.1 FINDING THE CONTROL STATUS REGISTER.

The auto-sizing routine finds a DMC11 as follows: It starts at address 160000 and tests all address in increments of 10 up to and including address 167760. If the address does not time out, the following is done, the first CROM address is written to a 125252 then it is read back. If it contains a -1 or 125252 or 63220 a DMC11 or KMC11 has been found, if not, the address is updated by 10 and the search continues. A -1 indicates a DMC11 with no CROM, a 125252 indicates a KMC11 with CRAM and a 63220 indicates a DMC11 with the DDCMP CROM. Further tests are performed at this point to determine which line unit, if any, is installed, if a loop-back connector is installed and various switch settings on the line unit. THIS IS WHY THE STATUS TABLE MUST BE VERIFIED BY THE USER AND IF ANY OF THE INFORMATION DOES NOT AGREE WITH THE HARDWARE THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS MUST BE ANSWERED. All DMC11's in the system will be found by the auto-sizer. If it does not find a DMC11 the diagnostic must be restarted and the questions answered.

8.5.2 FINDING THE VECTOR AND BR LEVEL

The vector area (address 300-776) is filled with the instruction IOT and ',+2' (next address). The processor status is started at 7 and the DMC is programmed to interrupt. The PS is lowered by 1 until the DMC interrupts, a delay is made and if no interupt occures at PS level 3 (because of a bad DMC11) the program assumes vector address 300 at BR level 5 and the problem should be fixed in the diagnostic. Once the problem is fixed; the program should be re-setup again to get correct vector. If an interupt occured; the address to which the DMC11 interupted to is picked up and reported as the vector. NOTE: if the vector reported is not the vector set up by you; there is a problem and AUTO SIZING should not be done.

8.6 SOFTWARE SWITCH REGISTER

If the diagnostic is run on an 11/04 or other CPU without a switch register then a software switch register is used to allow user the same switch options as described previously. If the hardware switch register does not exist or if one does and it contains all ones (177777) this software switch register is used.

Control:

To obtain control at any allowable time during execution of the diagnostic the operator types a CTRL G on the console terminal keyboard. As soon as the CTRL G is recognized, by the diagnostic, the following message will be displayed:

SWR=XXXXXX NEW?

Where XXXXXX is the current contents of the software switch register in octal. The software control routine will then await operator action. At which time the operator is required to type one or more of the legal characters: 1) 0 - 7, 2) line feed(<LF>), 3) carriage return(<CR>), or 4) control-U (CTRL U). No check is made for legality. If the input character is not a <LF>, <CR>, or CTRL U it is assumed to be an octal digit.

To change the contents of the SSR the operator simply types the new desired value in octal - leading zeros need not be typed. And terminates the input string with a <CR> or <LF> depending on the program action desired as described below. The input value will be truncated to the last 6 digits typed. At least one digit must be typed on any given input string prior to the terminator before a change to the SSR will occur.

When the input string is terminated with a <CR> the diagnostic will continue execution from the point at which it was interrupted. If a <CR> is the only thing typed the program will continue without changing the SSR. The <LF> differs from the <CR> by restarting the program as if it were restarted at address 200.

If a CTRL U is typed at any point in the input string prior to the terminator the input value will be disregarded and the prompt displayed (SWR = XXXXXX NEW?).

To set the SSR for the starting switches, first load the diagnostic, then hit CTRL G, then start the diagnostic.

DOCUMENT
****
DZDMH LST
****

```
6       MAINDEC-11-DZDMH-A   DMC11 LOCAL CROM, JUMP, AND FREE RUNNING TESTS
        COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
        ------------------------------------------------------------------

1626    ************************** TEST 1 ****************************
        TEST OF BR RIGHT SHIFT
        VERIFY THAT A DEST OF BR RSH (011) OF A MICRO-INSTRUCTION
        SHIFTS THE RESULTING BR DATA RIGHT ONCE.

1666    ************************** TEST 2 ****************************
        IOP CRAM WRITE/READ TEST
        FLOAT A 1 THROUGH EACH CRAM LOCATION

1700    ************************** TEST 3 ****************************
        IOP CRAM WRITE/READ TEST
        FLOAT A 0 THROUGH EACH CRAM LOCATION

1737    ************************** TEST 4 ****************************
        IOP CRAM DUAL ADDRESSING TEST
        WRITE EACH ADDRESS INTO ITSELF,READ EACH
        ADDRESS TO VERIFY CORRECT ADDRESSING

1783    ************************** TEST 5 ****************************
        IOP CRAM READ TEST
        THIS TEST WRITES THE CRAM WITH THE CROM MICRO-CODE MAP
        THEN READS IT BACK AND COMPARES EACH ADDRESS WITH THE
        DUPLICATE OF THE CROM MICRO-CODE.

1820    ************************** TEST 6 ****************************
        IOP MAIN MEMORY TEST
        FLOAT A 1 THROUGH ALL MAIN MEMORY LOCATIONS

1866    ************************** TEST 7 ****************************
        IOP MAIN MEMORY TEST
        FLOAT A 0 THROUGH ALL MAIN MEMORY LOCATIONS

1914    ************************** TEST 10 ***************************
        IOP MAIN MEMORY DUAL ADDRESSING TEST
        LOAD EACH MEMORY LOCATION WITH ITS OWN ADDRESS
        READ BACK EACH LOCATION TO VERIFY CORRECT ADDRESSING

1982    ************************** TEST 11 ***************************
        IOP MAR TEST
        PERFORM DUAL ADDRESSING TEST
        USING MAR AUTO-INC FEATURE

2022    ************************** TEST 12 ***************************
        IOP (CRAM) ODT BITS TEST
        LOAD MAR WITH A 0 INC MAR UNTIL IT OVERFLOWS (2000 TIMES)
        VERIFY THAT IBUS* 10 BIT5 IS SET ONLY WHEN MAR BIT 8 IS A ONE
        AND THAT IBUS* 10 BIT6 IS SET ON MAR OVERFLOW(2000)
```

```
2083    ************************** TEST 13 ****************************
        CROM READ TEST
        THIS TEST READS EACH ROM LOCATION AND COMPARES

2086    IT TO A SOFTWARE DUPLICATE OF THE CROM, THIS TEST
        ALSO TESTS THE JUMP(I) MICRO-PROCESSOR INSTRUCTION.

2132    ************************** TEST 14 ****************************
        CROM TEST OF JUMP(I) NEVER MICRO-PROCESSOR INSTRUCTION.
        PERFORM THE JUMP INSTRUCTION
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).

2189    ************************** TEST 15 ****************************
        CROM TEST OF JUMP(I) ALWAYS MICRO-PROCESSOR INSTRUCTION.
        PERFORM THE JUMP INSTRUCTION
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2242    ************************** TEST 16 ****************************
        CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
        SET THE C BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2298    ************************** TEST 17 ****************************
        CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
        SET THE Z BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2354    ************************** TEST 20 ****************************
        CROM TEST OF JUMP(I) ON BR0 SET MICRO-PROCESSOR INSTRUCTION.
        SET THE BR0 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2410    ************************** TEST 21 ****************************
        CROM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION.
        SET THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2466    ************************** TEST 22 ****************************
        CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION.
        SET THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2522    ************************** TEST 23 ****************************
        CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION.
        SET THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2578    ************************** TEST 24 ****************************
        CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
        CLEAR THE C BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
```

```
2635    ************************** TEST 25 ****************************
        CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION,
        CLEAR THE Z BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE),

2692    ************************** TEST 26 ****************************
        CROM TEST OF JUMP(I) ON BR0 SET MICRO-PROCESSOR INSTRUCTION,
        CLEAR THE BR0 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE),

2749    ************************** TEST 27 ****************************
        CROM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION,
        CLEAR THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE),

2806    ************************** TEST 30 ****************************
        CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION,
        CLEAR THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE),

2863    ************************** TEST 31 ****************************
        CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION,
        CLEAR THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE),

2920    ************************** TEST 32 ****************************
        CRAM TEST OF JUMP(I) NEVER MICRO-PROCESSOR INSTRUCTION,
        PERFORM THE JUMP INSTRUCTION
        VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT

2926    THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
        THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
        THEN PORT4 CONTAINS A 37

2982    ************************** TEST 33 ****************************
        CRAM TEST OF JUMP(I) ALWAYS MICRO-PROCESSOR INSTRUCTION,
        PERFORM THE JUMP INSTRUCTION
        VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
        THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
        THEN PORT4 WILL CONTAIN A 37
```

```
3041    ************************** TEST 34 ****************************
        CRAM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION,
        SET THE C BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
        THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
        THEN PORT4 WILL CONTAIN A 37

3103    ************************** TEST 35 ****************************
        CRAM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION,
        SET THE Z BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
        THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
        THEN PORT4 WILL CONTAIN A 37

3165    ************************** TEST 36 ****************************
        CRAM TEST OF JUMP(I) ON BR0 SET MICRO-PROCESSOR INSTRUCTION,
        SET THE BR0 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
        THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
        THEN PORT4 WILL CONTAIN A 37

3227    ************************** TEST 37 ****************************
        CRAM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION,
        SET THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
        THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
        THEN PORT4 WILL CONTAIN A 37

3289    ************************** TEST 40 ****************************
        CRAM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION,
        SET THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
        THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
        THEN PORT4 WILL CONTAIN A 37
```

```
3351    ************************** TEST 41 ****************************
        CRAM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION,
        SET THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
        THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
        THEN PORT4 WILL CONTAIN A 37

3413    ************************** TEST 42 ****************************
        CRAM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION,
        CLEAR THE C BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
        THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
        THEN PORT4 CONTAINS A 37

3475    ************************** TEST 43 ****************************
        CRAM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION,
        CLEAR THE Z BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
        THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
        THEN PORT4 CONTAINS A 37

3537    ************************** TEST 44 ****************************
        CRAM TEST OF JUMP(I) ON BR0 SET MICRO-PROCESSOR INSTRUCTION,
        CLEAR THE BR0 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE

3542    BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
        THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
        THEN PORT4 CONTAINS A 37

3599    ************************** TEST 45 ****************************
        CRAM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION,
        CLEAR THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
        THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
        THEN PORT4 CONTAINS A 37
```

```
3661    ************************** TEST 46 ****************************
        CRAM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION,
        CLEAR THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
        THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
        THEN PORT4 CONTAINS A 37

3723    ************************** TEST 47 ****************************
        CRAM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION,
        CLEAR THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
        IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
        BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
        THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
        THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
        THEN PORT4 CONTAINS A 37

3785    ************************** TEST 50 ****************************
        FREE RUNNING FLAG MODE DATA TEST
        TRANSMIT A MESSAGE AND VERIFY THE RECEIVED DATA
        IF NO TURNAROUND CONNECTOR IS ON LINE UNIT LOOP IS SET,
        ALL FOLLOWING TESTS ARE FREE RUNNING AND ARE PERFORMED
        ONLY ON DMC'S WITH LINE UNITS, IF YOU WISH TO PERFORM
        THESE FREE RUNNING TESTS ON A KMC (NORMALLY THE FREE RUNNING TESTS
        WILL FAIL ON A KMC, THE TIMER IS TOO FAST) THEN YOU MUST
        MANUALLY SET BIT0 OF STAT3 IN THE STATUS MAP,

3960    ************************** TEST 51 ****************************
        OVERUN TEST
        IN FREE RUNNING MODE SEND MESSAGE WITH NO RECEIVE
        BUFFER AVAILABLE, VERIFY THAT AN OVERRUN ERROR OCCURS

4016    ************************** TEST 52 ****************************
        LOST DATA TEST
        IN FREE RUNNING MODE SEND A MESSAGE LONGER THAN THE RECEIVE
        BUFFER, VERIFY THAT A LOST DATA ERROR OCCURS,

4065    ************************** TEST 53 ****************************
        TRANSMIT NON-EXISTENT MEMORY TEST
        IN FREE RUNNING MODE, LOAD A TRANSMIT BA THAT WILL TIME OUT
        VERIFY THAT A NON-EXISTENT MEMORY ERROR OCCURS

4111    ************************** TEST 54 ****************************
        RECEIVE NON-EXISTENT MEMORY TEST
        IN FREE RUNNING MODE, LOAD A RECEIVE BA THAT WILL TIME OUT
        VERIFY THAT A NON-EXISTENT MEMORY ERROR OCCURS
```

```
4160    ************************** TEST 55 ***************************
        PROCESSOR ERROR TEST
        IN FREE RUNNING MODE, DO A BASE TRANSFER REQUEST AFTER A
        BASE HAS BEEN SET UP, VERIFY THAT A PROCESSOR ERROR OCCURS,

4204    ************************** TEST 56 ***************************
        PROCESSOR ERROR TEST
        IN FREE RUNNING MODE DO A RQI WITH AN ILLEGAL 10 CODE
        VERIFY THAT A PROCESSOR ERROR OCCURS

4248    ************************** TEST 57 ***************************
        HALF DUPLEX TEST
        IN FREE RUNNING MODE, SET HALF DUPLEX AND L U LOOP
        SEND A MESSAGE AND VERIFY THAT THERE ARE NO DONES

4288    ************************** TEST 60 ***************************
        FREE RUNNING DATA TEST (INTERRUPT DRIVEN EXERCISER)
        THIS TEST REPEATEDLY QUEUES UP 7 RECEIVE BUFFERS AND
        7 TRANSMIT BUFFERS AND CHECKS DATA WHEN ALL 7 BUFFERS
        ARE RECEIVED, TRANSMIT COUNTS RANGE FROM 1 TO 104, ALSO
        ODD AND EVEN TRANSMIT AND RECEIVE BA'S ARE USED, DATA
        IS A BINARY COUNT PATTERN, THE RESUME FUNCTION IS CHECKED IN THIS TEST
```

```
                                ;*MAINDEC-11-DZDMH-A   DMC11 LOCAL CROM, JUMP, AND FREE RUNNING TESTS
                                ;*COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
                                ;*********************************************************************

                                ;STARTING PROCEDURE
                                ;LOAD PROGRAM
                                ;LOAD ADDRESS 000200
                                ;SWR=0  AUTOSIZE DMC11
                                ;SW07=1  USE CURRENT DMC11 PARAMETERS
                                ;SW00=1  INPUT NEW DMC11 PARAMETERS
                                ;PRESS START
                                ;PROGRAM WILL TYPE "MAINDEC-11-DZDMH-A   DMC11 LOCAL CROM, JUMP, AND FREE RUNNIN
                                ;PROGRAM WILL TYPE STATUS MAP
                                ;PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
                                ;AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
                                ;AND THEN RESUME TESTING
                                ;SUBSEQUENT RESTARTS WILL NOT TYPE PROGRAM TITLE



                                ;SWITCH REGISTER OPTIONS
                                ;======================

100000                          SW15=100000             ;=1,HALT ON ERROR
040000                          SW14=40000              ;=1,LOOP ON CURRENT TEST
020000                          SW13=20000              ;=1,INHIBIT ERROR TYPEOUT
010000                          SW12=10000              ;=1,DELETE TYPEOUT/BELL ON ERROR,
004000                          SW11=4000               ;=1,INHIBIT ITERATIONS
002000                          SW10=2000               ;=1,ESCAPE TO NEXT TEST ON ERROR
001000                          SW09=1000               ;=1,LOOP WITH CURRENT DATA
000400                          SW08=400                ;=1,LOOP ON ERROR
000200                          SW07=200                ;=1,USE CURRENT DMC11 PARAMETERS, =0,AUTOSIZE DMC11
000100                          SW06=100                ;=1, HALT BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION
000040                          SW05=40
000020                          SW04=20
000010                          SW03=10                 ;RESELECT DMC11'S TO BE TESTED (ACTIVE)
000004                          SW02=4                  ;LOCK ON TEST SELECT
000002                          SW01=2                  ;RESTART PROGRAM AT SELECTED TEST
000001                          SW00=1                  ;INPUT DMC11 PARAMETERS
```



```

                                ;REGISTER DEFINITIONS
                                ;====================

000000                          R0=%0           ;GENERAL REGISTER
000001                          R1=%1           ;GENERAL REGISTER
000002                          R2=%2           ;GENERAL REGISTER
000003                          R3=%3           ;GENERAL REGISTER
000004                          R4=%4           ;GENERAL REGISTER
000005                          R5=%5           ;GENERAL REGISTER
000006                          SP=%6           ;PROCESSOR STACK POINTER
000007                          PC=%7           ;PROGRAM COUNTER

                                ;LOCATION EQUIVALENCIES
                                ;======================

177776                          PS=177776       ;PROCESSOR STATUS WORD
001200                          STACK=1200      ;START OF PROCESSOR STACK

                                ;INSTRUCTION DEFINITIONS
                                ;=======================

005746                          PUSH1SP=5746    ;DECREMENT PROCESSOR STACK 1 WORD
005726                          POP1SP=5726     ;INCREMENT PROCESSOR STACK 1 WORD
010046                          PUSHR0=10046    ;SAVE R0 ON STACK
012600                          POPR0=12600     ;RESTORE R0 FROM STACK
024646                          PUSH2SP=24646   ;DECREMENT STACK TWICE
022626                          POP2SP=22626    ;INCREMENT STACK TWICE
                                .EQUIV  EMT,HLT ;BASIC DEFINITION OF ERROR CALL

                                ;BIT DEFINITIONS
                                ;===============

100000                          BIT15=100000
040000                          BIT14=40000
020000                          BIT13=20000
010000                          BIT12=10000
004000                          BIT11=4000
002000                          BIT10=2000
001000                          BIT9=1000
000400                          BIT8=400
000200                          BIT7=200
000100                          BIT6=100
000040                          BIT5=40
000020                          BIT4=20
000010                          BIT3=10
000004                          BIT2=4
000002                          BIT1=2
000001                          BIT0=1
```

```
                                        ;!*********************************************************************
                                        ;---------------------------------------------------------------------
                                                ;TRAPCATCAER FOR ILLEGAL INTERRUPTS
                                                ;THE STANDARD "TRAP CATCHER" IS PLACED
                                                ;BETWEEN ADDRESS 0 TO ADDRESS 776.
                                                ;IT LOOKS LIKE "PC+2 HALT".
                                        ;---------------------------------------------------------------------
                                        ;!*********************************************************************

          000000                        .=0
                                        ;STANDARD INTERRUPT VECTORS
                                        ;--------------------------

          000024                        .=24
000024  005240                                  .PFAIL                  ;POWER FAIL HANDLER
000026  000340                                  340                     ;SERVICE AT LEVEL 7
000030  004656                                  .HLT                    ;ERROR HANDLER
000032  000340                                  340                     ;SERVICE AT LEVEL 7
000034  004624                                  .TRPSRV                 ;GENERAL HANDLER DISPATCH SERVICE
000036  000340                                  340                     ;SERVICE AT LEVEL 7
          000040
          000040                        .=40
000040  000000                                  0                       ;SAVE FOR ACT-11 OR XXDP
000042  000000                                  0                       ;RETURN ADDRESS IF UNDER ACT-11 OR XXDP
000044  000000                                  0                       ;SAVE FOR ACT-11 OR XXDP
000046  003432                                  $ENDAD                  ;FOR USE WITH ACT-11 OR XXDP
          000052
          000052                        .=52
000052  000000                                  0                       ;ACT-11 PROGRAM CHARACTERISTICS

          000174                        .=174
000174  000000                        DISPREG:0                        ;SOFTWARE DISPLAY REGISTER
000176  000000                        SWREG:  0                        ;SOFTWARE SWITCH REGISTER

          000200                        .=200
000200  000137  002002                          JMP     .START          ;GO TO START OF PROGRAM

          001000                        .=1000
001000  005377  040515  047111         MTITLE: .ASCII  <377><12>/MAINDEC-11-DZDMH-A/<377>
(2) 001025     104  041515  030461                .ASCIZ  /DMC11 LOCAL CROM, JUMP, AND FREE RUNNING TESTS/<377>
(2)
          001200                        .=1200

                                        ;INDIRECT POINTERS TO SWITCH REGISTER AND LIGHT DISPLAY
                                        ;-----------------------------------------------------

001200  177570                        DISPLAY:177570
001202  177570                        SWR:    177570
```



```

                                        ;INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS
                                        ;---------------------------------------------------

001204  177560                        TKCSR:  177560                   ;TELETYPE KEYBOARD CONTROL REGISTER
001206  177562                        TKDBR:  177562                   ;TELETYPE KEYBOARD DATA BUFFER
001210  177564                        TPCSR:  177564                   ;TELEPRINTER CONTROL REGISTER
001212  177566                        TPDBR:  177566                   ;TELEPRINTER DATA BUFFER

                                        ;PROGRAM CONTROL PARAMETERS
                                        ;--------------------------

001214  000000                        RETURN: 0                        ;SCOPE ADDRESS FOR LOOP ON TEST
001216  000000                        NEXT:   0                        ;ADDRESS OF NEXT TEST TO BE EXECUTED
001220  000000                        LOCK:   0                        ;ADDRESS FOR LOCK ON CURRENT DATA
001222  000003                        ICOUNT: 3                        ;NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE
001224  000000                        LPCNT:  0                        ;NUMBER OF ITEREATIONS COMPLETED
001226  000000                        TSTNO:  0                        ;NUMBER OF TEST IN PROGRESS
001230  000000                        PASCNT: 0                        ;NUMBER OF PASSES COMPLETED
001232  000000                        ERRCNT: 0                        ;TOTAL NUMBER OF ERRORS
001234  000000                        LSTERR: 0                        ;PC OF LAST ERROR CALL

                                        ;PROGRAM VARIABLES
                                        ;-----------------

001236  000000                        STRTSW: 0                        ;SWITCHES AT START OF PROGRAM
001240  000000                        STAT:   0                        ;DM STATUS WORD STORAGE
001242  000000                        CLKX:   0
001244  000000                        MASKX:  0
001246  000000                        TEMP1:  0                        ;TEMPORARY STORAGE
001250  000000                        TEMP2:  0                        ;TEMPORARY STORAGE
001252  000000                        TEMP3:  0                        ;TEMPORARY STORAGE
001254  000000                        TEMP4:  0                        ;TEMPORARY STORAGE
001256  000000                        TEMP5:  0                        ;TEMPORARY STORAGE
001260  000000                        SAVR0:  0                        ;R0 STORAGE
001262  000000                        SAVR1:  0                        ;R1 STORAGE
001264  000000                        SAVR2:  0                        ;R2 STORAGE
001266  000000                        SAVR3:  0                        ;R3 STORAGE
001270  000000                        SAVR4:  0                        ;R4 STORAGE
001272  000000                        SAVR5:  0                        ;R5 STORAGE
001274  000000                        SAVSP:  0                        ;STACK POINTER STORAGE
001276  000000                        SAVPC:  0                        ;PROGRAM COUNTER STORAGE
001300  000000                        ZERO:   0
001302  000001                        ONE:    1
001304  000000                        MEMLIM: 0                        ;HIGHEST LOCATION FOR NPR'S
001306  000001                        DMACTV: .BLKW 1                  ;DMC11'S SELECTED ACTIVE,
001310  000001                        DMNUM:  .BLKW 1                  ;OCTAL NUMBER OF DMC11'S,
001312  000001                        SAVACT: .BLKW 1                  ;ORIGINAL ACTV  DEVICES
001314  000001                        SAVNUM: .BLKW 1                  ;WORKABLE NUMBER
001316  000000                        RUN:    0                        ;POINTER TO RUNNING DEVICE,
                                        .EVEN
001320  001472                        CREAM:  DM.MAP-6                 ;TABLE POINTER,
001322  001676                        MILK:   CNT.MAP-4                ;TABLE POINTER
```

```
                                        ;PROGRAM CONTROL FLAGS
                                        ;---------------------
001324  000                             INIFLG: .BYTE   0               ;PROGRAM INITIALIZATION FLAG
001325  000                             ERRFLG: .BYTE   0               ;ERROR OCCURED FLAG
001326  000                             LOKFLG: .BYTE   0               ;LOCK ON CURRENT TEST FLAG
001327  000                             QV.FLG: .BYTE   0               ;QUICK VERIFY FLAG,
                                                                        ;ON FIRST PASS OF EACH DMC11 ITERATIONS WILL BE
                                        .EVEN

                                                ;DEFINITIONS FOR TRAP SUBROUTINE CALLS
                                                ;POINTERS TO SUBROUTINES CAN BE FOUND
                                                ;IN THE TABLE IMMEDIATLY FOLLOWING THE DEFINITIONS

                                        ;!*******************************************************************
                                        ;-------------------------------------------------------------------
001330                                  .TRPTAB:
        104400                          SCOPE=TRAP+0            ;CALL TO SCOPE LOOP AND ITERATION HANDLER
001330  003506                                  .SCOPE
        104401                          SCOP1=TRAP+1            ;CALL TO LOOP ON CURRENT DATA HANDLER
001332  003644                                  .SCOP1
        104402                          TYPE=TRAP+2             ;CALL TO TELETYPE OUTPUT ROUTINE
001334  003674                                  .TYPE
        104403                          INSTR=TRAP+3            ;CALL TO ASCII STRING INPUT ROUTINE
001336  003756                                  .INSTR
        104404                          INSTER=TRAP+4           ;CALL TO INPUT ERROR HANDLER
001340  004062                                  .INSTER
        104405                          PARAM=TRAP+5            ;CALL TO NUMERICAL DATA INPUT ROUTINE
001342  004102                                  .PARAM
        104406                          SAV05=TRAP+6            ;CALL TO REGISTER SAVE ROUTINE
001344  004302                                  .SAV05
        104407                          RES05=TRAP+7            ;CALL TO REGISTER RESTORE ROUTINE
001346  004342                                  .RES05
        104410                          CONVRT=TRAP+10          ;CALL TO DATA OUTPUT ROUTINE
001350  004374                                  .CONVRT
        104411                          CNVRT=TRAP+11           ;CALL TO DATA OUTPUT ROUNTINE WITHOUT CR/LF,
001352  004400                                  .CNVRT
        104412                          MSTCLR=TRAP+12          ;CALL TO ISUE A MASTER CLEAR
001354  005370                                  .MSTCLR
        104413                          DELAY=TRAP+13           ;CALL TO DELAY
001356  005340                                  .DELAY
        104414                          ROMCLK=TRAP+14          ;CALL TO CLOCK ROM ONCE
001360  005406                                  .ROMCLK
        104415                          DATACLK=TRAP+15         ;CALL TO CLK DATA
001362  005454                                  .DATACLK
        104416                          TIMER=TRAP+16           ;CALL TO DELAY A CLOCK TICK
001364  005520                                  .TIMER

                                        ;-------------------------------------------------------------------
                                        ;!*******************************************************************
```



```
                                        ;DMC11 CONTROL INDICATORS FOR CURRENT DMC11 UNDER TEST
                                        ;-----------------------------------------------

001366  000000                          STAT1:  0
001370  000000                          STAT2:  0
001372  000000                          STAT3:  0

                                        ;DMC11 VECTOR AND REGISTER INDIRECT POINTERS
                                        ;-------------------------------------------

001374  000000                          DMRVEC: 0               ;POINTER TO DMC11 RECEIVER INTERRUPT VECTOR
001376  000000                          DMRLVL: 0               ;POINTER TO DMC11 RECEIVER INTERRUPT SERVICE PS
001400  000000                          DMTVEC: 0               ;POINTER TO DMC11 TRANSMITTER INTERRUPT VECTOR
001402  000000                          DMTLVL: 0               ;POINTER TO DMC11 TRANSMITTER INTERRUPT SERVICE PS
001404  000000                          DMCSR:  0               ;POINTER TO DMC11 CONTROL STATUS REGISTER
001406  000000                          DMCSRH: 0               ;POINTER TO DMC11 CONTROL STATUS REGISTER HIGH BYTE,
001410  000000                          DMCTL:  0               ;POINTER TO DMC11 CONTOL OUT REGISTER
001412  000000                          DMPO4:  0               ;POINTER TO DMC11 PORT REGISTER(SEL 4)
001414  000000                          DMPO6:  0               ;POINTER TO DMC11 PORT REGISTER(SEL 6)

                                        ;TEMP STORAGE
                                        ;------------

001416  000000                          TEMP:   0
        001460                          .=.+40

                                        ;DMC11 STATUS TABLE AND ADDRESS ASSIGNMENTS
                                        ;------------------------------------------

        001500                          .=1500
001500                                  DM.MAP:
001500  000001                          DMCR00: .BLKW   1               ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 00
001502  000001                          DMS100: .BLKW   1               ;VECTOR FOR DMC11 NUMBER 00
001504  000001                          DMS200: .BLKW   1               ;DDCMP LINE# FOR DMC11 NUMBER 00
001506  000001                          DMS300: .BLKW   1               ;3RD STATUS WORD

001510  000001                          DMCR01: .BLKW   1               ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 01
001512  000001                          DMS101: .BLKW   1               ;VECTOR FOR DMC11 NUMBER 01
001514  000001                          DMS201: .BLKW   1               ;DDCMP LINE# FOR DMC11 NUMBER 01
001516  000001                          DMS301: .BLKW   1               ;3RD STATUS WORD

001520  000001                          DMCR02: .BLKW   1               ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 02
001522  000001                          DMS102: .BLKW   1               ;VECTOR FOR DMC11 NUMBER 02
001524  000001                          DMS202: .BLKW   1               ;DDCMP LINE# FOR DMC11 NUMBER 02
001526  000001                          DMS302: .BLKW   1               ;3RD STATUS WORD

001530  000001                          DMCR03: .BLKW   1               ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 03
001532  000001                          DMS103: .BLKW   1               ;VECTOR FOR DMC11 NUMBER 03
001534  000001                          DMS203: .BLKW   1               ;DDCMP LINE# FOR DMC11 NUMBER 03
001536  000001                          DMS303: .BLKW   1               ;3RD STATUS WORD

001540  000001                          DMCR04: .BLKW   1               ;CONTROL STATUS REGISTEP FOR DMC11 NUMBER 04
001542  000001                          DMS104: .BLKW   1               ;VECTOR FOR DMC11 NUMBER 04
001544  000001                          DMS204: .BLKW   1               ;DDCMP LINE# FOR DMC11 NUMBER 04
001546  000001                          DMS304: .BLKW   1               ;3RD STATUS WORD
```

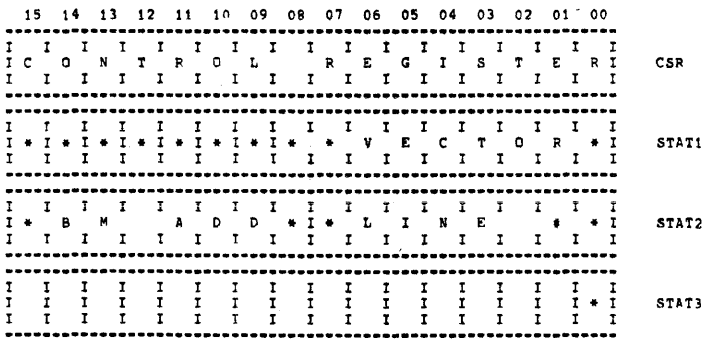
```
001550 000001          DMCR05: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 05
001552 000001          DMS105: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 05
001554 000001          DMS205: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 05
001556 000001          DMS305: .BLKW   1          ;3RD STATUS WORD

001560 000001          DMCR06: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 06
001562 000001          DMS106: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 06
001564 000001          DMS206: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 06
001566 000001          DMS306: .BLKW   1          ;3RD STATUS WORD

001570 000001          DMCR07: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 07
001572 000001          DMS107: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 07
001574 000001          DMS207: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 07
001576 000001          DMS307: .BLKW   1          ;3RD STATUS WORD

001600 000001          DMCR10: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 10
001602 000001          DMS110: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 10
001604 000001          DMS210: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 10
001606 000001          DMS310: .BLKW   1          ;3RD STATUS WORD

001610 000001          DMCR11: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 11
001612 000001          DMS111: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 11
001614 000001          DMS211: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 11
001616 000001          DMS311: .BLKW   1          ;3RD STATUS WORD

001620 000001          DMCR12: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 12
001622 000001          DMS112: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 12
001624 000001          DMS212: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 12
001626 000001          DMS312: .BLKW   1          ;3RD STATUS WORD

001630 000001          DMCR13: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 13
001632 000001          DMS113: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 13
001634 000001          DMS213: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 13
001636 000001          DMS313: .BLKW   1          ;3RD STATUS WORD

001640 000001          DMCR14: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 14
001642 000001          DMS114: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 14
001644 000001          DMS214: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 14
001646 000001          DMS314: .BLKW   1          ;3RD STATUS WORD

001650 000001          DMCR15: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 15
001652 000001          DMS115: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 15
001654 000001          DMS215: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 15
001656 000001          DMS315: .BLKW   1          ;3RD STATUS WORD

001660 000001          DMCR16: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 16
001662 000001          DMS116: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 16
001664 000001          DMS216: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 16
001666 000001          DMS316: .BLKW   1          ;3RD STATUS WORD

001670 000001          DMCR17: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 17
001672 000001          DMS117: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 17
001674 000001          DMS217: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 17
001676 000001          DMS317: .BLKW   1          ;3RD STATUS WORD
```



```
001700 000000          DM.END: 000000
```

```
                                ;DMC11 PASS COUNT AND ERROR COUNT TABLE
                                ;--------------------------------------
001702                          CNT.MAP:
001702  000000                  PACT00: 0               ;PASS COUNT FOR DMC11 NUMBER 00
001704  000000                  ERCT00: 0               ;ERROR COUNT FOR DMC11 NUMBER 00

001706  000000                  PACT01: 0               ;PASS COUNT FOR DMC11 NUMBER 01
001710  000000                  ERCT01: 0               ;ERROR COUNT FOR DMC11 NUMBER 01

001712  000000                  PACT02: 0               ;PASS COUNT FOR DMC11 NUMBER 02
001714  000000                  ERCT02: 0               ;ERROR COUNT FOR DMC11 NUMBER 02

001716  000000                  PACT03: 0               ;PASS COUNT FOR DMC11 NUMBER 03
001720  000000                  ERCT03: 0               ;ERROR COUNT FOR DMC11 NUMBER 03

001722  000000                  PACT04: 0               ;PASS COUNT FOR DMC11 NUMBER 04
001724  000000                  ERCT04: 0               ;ERROR COUNT FOR DMC11 NUMBER 04

001726  000000                  PACT05: 0               ;PASS COUNT FOR DMC11 NUMBER 05
001730  000000                  ERCT05: 0               ;ERROR COUNT FOR DMC11 NUMBER 05

001732  000000                  PACT06: 0               ;PASS COUNT FOR DMC11 NUMBER 06
001734  000000                  ERCT06: 0               ;ERROR COUNT FOR DMC11 NUMBER 06

001736  000000                  PACT07: 0               ;PASS COUNT FOR DMC11 NUMBER 07
001740  000000                  ERCT07: 0               ;ERROR COUNT FOR DMC11 NUMBER 07

001742  000000                  PACT10: 0               ;PASS COUNT FOR DMC11 NUMBER 10
001744  000000                  ERCT10: 0               ;ERROR COUNT FOR DMC11 NUMBER 10

001746  000000                  PACT11: 0               ;PASS COUNT FOR DMC11 NUMBER 11
001750  000000                  ERCT11: 0               ;ERROR COUNT FOR DMC11 NUMBER 11

001752  000000                  PACT12: 0               ;PASS COUNT FOR DMC11 NUMBER 12
001754  000000                  ERCT12: 0               ;ERROR COUNT FOR DMC11 NUMBER 12

001756  000000                  PACT13: 0               ;PASS COUNT FOR DMC11 NUMBER 13
001760  000000                  ERCT13: 0               ;ERROR COUNT FOR DMC11 NUMBER 13

001762  000000                  PACT14: 0               ;PASS COUNT FOR DMC11 NUMBER 14
001764  000000                  ERCT14: 0               ;ERROR COUNT FOR DMC11 NUMBER 14

001766  000000                  PACT15: 0               ;PASS COUNT FOR DMC11 NUMBER 15
001770  000000                  ERCT15: 0               ;ERROR COUNT FOR DMC11 NUMBER 15

001772  000000                  PACT16: 0               ;PASS COUNT FOR DMC11 NUMBER 16
001774  000000                  ERCT16: 0               ;ERROR COUNT FOR DMC11 NUMBER 16

001776  000000                  PACT17: 0               ;PASS COUNT FOR DMC11 NUMBER 17
002000  000000                  ERCT17: 0               ;ERROR COUNT FOR DMC11 NUMBER 17
```


FORMAT OF STATUS TABLE



DEFINITION OF FORMAT

CSR: CONTAINS DMC11 CSR ADDRESS

STAT1: BITS 00-08 IS DMC11 VECTOR ADDRESS
BIT15=1 MICRO-PROCESSOR HAS CRAM
BIT15=0 MICRO-PROCESSOR HAS CROM
BIT14=1 ???? TURNAROUND CONNECTOR IS ON
BIT14=0 NO TURNAROUND CONNECTOR
BIT13=0 LINE UNIT IS AN M8201
BIT13=1 LINE UNIT IS AN M8202
BIT12=1 NO LINE UNIT
BITS 09-11 IS DMC11 BR PRIORITY LEVEL

STAT2: LOW BYTE IS SWITCH PAC#1 (DDCMP LINE NUMBER)
HIGH BYTE IS SWITCH PAC#2 (BM873 BOOT ADD)

STAT3: BIT0=1 DO FREE RUNNING TESTS ON KMC
(MUST BE SET TO A ONE MANUALLY [PROGRAMS G AND H ONLY])

```
                                        ;PROGRAM INITIALIZATION
                                        ;LOCK OUT INTERRUPTS
                                        ;SET UP PROCESSOR STACK
                                        ;SET UP POWER FAIL VECTOR
                                        ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
                                        ;TYPE TITLE MESSAGE

002002  012737  000340  177776  .START: MOV     #340,PS         ;LOCK OUT INTERRUPTS
002010  012706  001200                  MOV     #STACK,SP       ;SET UP STACK
002014  012737  005240  000024          MOV     #.PFAIL,@#24    ;SET UP POWER FAIL VECTOR
002022  013737  001310  001314          MOV     DMNUM,SAVNUM    ;SAVE NUMBER OF DEVICES IN SYSTEM.
002030  005037  007556                  CLR     SWFLG           ;CLEAR SOFT TYPEOUT FLAG
002034  105037  001325                  CLRB    ERRFLG          ;CLEAR ERROR FLAG
002040  105037  001327                  CLRB    QV.FLG          ;ZERO QUICK VERIFY FLAG
002044  012737  001470  001320          MOV     #DM,MAP=10,CREAM;GET MAP POINTER,
002052  012737  001676  001322          MOV     #CNT,MAP=4,MILK ;GET PASS COUNT MAP POINTER
002060  012737  100000  001316          MOV     #BIT15,RUN      ;POINT POINTER TO FIRST DEVICE.
002066  012700  001702                  MOV     #CNT,MAP,R0     ;PASS COUNT POINTER TO R0
002072  005020                  23$:    CLR     (R0)+           ;CLEAR TABLE
002074  022700  002002                  CMP     #CNT,MAP+100,R0 ;DONE YET?
002100  001374                          BNE     23$             ;KEEP GOING
002102  005037  001234                  CLR     LSTERR          ;CLEAR LAST ERROR POINTER
002106  012737  000001  001226          MOV     #1,TSTNO        ;SET UP FOR TEST 1
002114  012737  002002  001214          MOV     #.START,RETURN  ;SET UP FOR POWER FAIL BEFORE
                                        ;TESTING STARTS
002122  013746  000006                  MOV     @#6,-(SP)       ;SAVE CURRENT VECTORS
002126  013746  000004                  MOV     @#4,-(SP)       ;
002132  012737  002166  000004          MOV     #6$,@#4         ;SET UP FOR TIMEOUT
002140  012737  177570  001202          MOV     #177570,SWR     ;SET SWR TO HARD SWR ADDRESS
002146  012737  177570  001200          MOV     #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
002154  022777  177777  177020          CMP     #-1,@SWR        ;REFERENCE HARDWARE SWITCH REGISTER
002162  001402                          BEQ     6$+2            ;IF = -1 USE SOFT SWR ANYWAY
002164  000407                          BR      7$              ;IF IT EXISTS AND NOT = -1 USE HARD SWR
002166  022626                  6$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
002170  012737  000176  001202          MOV     #SWREG,SWR      ;POINTER TO SOFT SWR
002176  012737  000174  001200          MOV     #DISPREG,DISPLAY;POINTER TO SOFT DISPLAY REG
002204  012637  000004                  MOV     (SP)+,@#4       ;RESTORE VECTORS
002210  012637  000006                  MOV     (SP)+,@#6       ;
002214  105737  001324                  TSTB    INIFLG          ;HAS INITIALIZATION BEEN PERFORMED
002220  001006                          BNE     20$             ;BR IF YES
002222  022737  003432  000042          CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
002230  001402                          BEQ     20$
002232  104402  001000                  TYPE    ,MTITLE         ;TYPE TITLE MESSAGE
002236  004737  007362          20$:    JSR     PC,CK$WR        ;CHECK FOR SOFT SWR
002242  017737  176734  001236          MOV     @SWR,STRTSW     ;STORE STARTING SWITCHES
002250  005737  000042                  TST     @#42            ;IS IT RUNNING IN AUTO MODE?
002254  001402                          BEQ     .+6             ;BR IF NO
002256  005037  001236                  CLR     STRTSW          ;IF YES, CLEAR SWITCHES
002262  032737  000001  001236          BIT     #SW00,STRTSW    ;IF SW00=1, QUESTIONS ARE ASKED.
002270  001012                          BNE     17$             ;BR IF SW00=1
002272  105737  001236                  TSTB    STRTSW          ;BIT7=1??
002276  100007                          BPL     17$             ;BR IF SW07=0
002300  005737  001306                  TST     DMACTV          ;ARE ANY DEVICES SELECTED?
002304  001006                          BNE     16$             ;BR IF YES
002306  104402  007056                  TYPE,   NOACT           ;NO DEVICES SELECTED.
```



```
002312  000000                          HALT                    ;STOP THE SHOW
002314  000776                          BR      .-2             ;DISQUALIFY CONTINUE SWITCH
002316  004737  010252          17$:    JSR     PC,AUTO.SIZE    ;GO DO THE AUTO SIZE
002322  105737  001324          16$:    TSTB    INIFLG          ;FIRST TIME?
002326  001410                          BEQ     21$             ;BR IF YES
002330  105737  001236                  TSTB    STRTSW          ;IF USING SAME PARAMETERS DONT TYPE MAP
002334  100431                          BMI     1$
002336  032737  000006  001236          BIT     #BIT1!BIT2,STRTSW;IS TEST NO, OR LOCK SELECTED
002344  001403                          BEQ     24$             ;IF NO THEN TYPE STATUS
002346  000424                          BR      1$              ;IF YES DO NOT TYPE STATUS
002350  005137  001324          21$:    COM     INIFLG          ;SET FLAG
002354  104402  006126          24$:    TYPE    ,XHEAD          ;TYPE HEADER
002360  012704  001500                  MOV     #DM,MAP,R4      ;SET POINTER
002364  010437  001246          5$:     MOV     R4,TEMP1        ;SET ADDRESS
002370  012437  001250                  MOV     (R4)+,TEMP2     ;SET CSR
002374  001411                          BEQ     1$              ;ALL DONE IF ZERO
002376  012437  001252                  MOV     (R4)+,TEMP3     ;SET STAT1
002402  012437  001254                  MOV     (R4)+,TEMP4     ;SET STAT2
002406  012437  001256                  MOV     (R4)+,TEMP5     ;SET STAT3
002412  104410                          CONVRT                  ;TYPE OUT STATUS MAP
002414  007230                          XSTATQ                  ;
002416  000762                          BR      5$
002420  012700  001500          1$:     MOV     #DM,MAP,R0      ;R0 POINTS TO STATUS TABLE

                                        ;;*********************************************************************
                                        ;;*AUTO SIZE TEST
                                        ;;*THIS TEST VERIFYS THAT THE DMC11S AND/OR KMC11S ARE AT THE CORRECT FLOATING
                                        ;;*ADDRESSES FOR YOUR SYSTEM. IF THIS TEST FAILS, IT IS NOT A HARDWARE ERROR.
                                        ;;*CHECK THE ADDRESSES OF ALL FLOATING DEVICES (DJ,DH,DQ,DU,DUP,LK,DMC,DZ,KMC).
                                        ;;*IF THERE ARE NO OTHER FLOATING DEVICES BEFORE THE DMC11, THE FIRST
                                        ;;*DMC11 ADDRESS IS 760070, KMC11 IS 760110. NO DEVICE SHOULD EVER BE AT
                                        ;;*ADDRESS 760000.
                                        ;;*********************************************************************

002424  013746  000004                  MOV     @#4,-(SP)       ;SAVE LOC 4
002430  013746  000006                  MOV     @#6,-(SP)       ;SAVE LOC 6
002434  005037  000006                  CLR     @#6             ;CLEAR VEC+2
002440  005037  001252                  CLR     TEMP3           ;CLEAR FLAG
002444  005005                          CLR     R5              ;R5=0=DMC, R5=-1=KMC
002446  011037  001404          AUSTRT: MOV     (R0),DMCSR      ;GET NEXT DMC CSR
002452  001530                          BEQ     AUDONE          ;BR IF DONE
002454  005705                          TST     R5              ;DMC OR KMC?
002456  001005                          BNE     1$              ;BR IF KMC
002460  032760  100000  000002          BIT     #BIT15,2(R0)    ;CHECK FOR DMC CSR
002466  001044                          BNE     OK              ;SKIP IF NOT DMC
002470  000404                          BR      2$              ;ITS A DMC SO CONTINUE
002472  032760  100000  000002  1$:     BIT     #BIT15,2(R0)    ;CHECK FOR KMC CSR
002500  001437                          BEQ     OK              ;SKIP IF NOT KMC
002502  012737  002606  000004  2$:     MOV     #NODEV,@#4      ;SET UP FOR TIMEOUT
002510  005705                          TST     R5              ;DMC OR KMC?
002512  001003                          BNE     3$              ;BR IF KMC
002514  012703  000006                  MOV     #6,R3           ;R3 IS COUNT OF DEVICES BEFORE DMC
002520  000402                          BR      4$              ;GO ON
002522  012703  000010          3$:     MOV     #10,R3          ;R3 IS COUNT OF DEVICES BEFORE KMC
002526  012702  002722          4$:     MOV     #DEVTAB,R2      ;R2 IS DEVICE TABLE POINTER
002532  012701  160010                  MOV     #160010,R1      ;START WITH ADDRESS 160010
```

```
                                FLOAT:  TST     (R1)            ;CHECK ADDRESS IN R1
                                        MOVB    (R2),R4         ;IF NO TIMEOUT, GET NEXT ADDRESS
                                        ADD     R4,R1           ;IN R1
                                        INC     R1              ;
                                        BIC     R4,R1           ;
                                        TST     R3              ;ANY MORE DEVICES TO CHECK FOR?
                                        BNE     FLOAT           ;BR IF YES
                                        MOV     #ERR,@#4        ;OK ONLY DMC'S ARE LEFT, SET UP FOR TIMEOUT
                                FY:     TST     (R1)            ;CHECK DMC ADDRESS
                                        CMP     R1,DMCSR        ;DOES IT MATCH
                                        BEQ     OK              ;BR IF YES
                                        ADD     #10,R1          ;GET NEXT DMC ADDRESS
                                        BR      FY              ;DO IT AGAIN
                                OK:     ADD     #10,R0          ;SKIP TO NEXT DMC CSR
                                        BR      AUSTRT          ;CONTINUE
                                NODEV:  CMPB    (R2)+,-(R3)     ;ON TIMEOUT, INC R2, DEC R3
                                        RTI                     ;RETURN
                                ERR:    TST     TEMP3           ;CHECK FLAG IF = 0 TYPE HEADER
                                        BNE     1$              ;SKIP HEADER
                                        TYPE                    ;TYPEOUT HEADER MESSAGE
                                        CONERR                  ;CONFIGURATION ERROR!!!!
                                        MOV     #ERR,SAVPC      ;SAVE PC FOR TYPEOUT
                                        CNVRT                   ;TYPE OUT ERROR PC
                                        ERRPC                   ;
                                        TYPE                    ;TYPE REST OF HEADER
                                        CNERR                   ;
                                        MOV     #-1,TEMP3       ;SET FLAG SO IT ONLY GETS TYPED ONCE
                                1$:     MOV     R1,SAVR1        ;SAVE R1 FOR TYPEOUT
                                        CONVRT
                                        CONTAB                  ;TYPE CSR VALUES
                                        TST     R5              ;DMC OR KMC ?
                                        BNE     3$              ;BR IF KMC
                                        TYPE
                                        DMCM
                                        BR      4$              ;CONTINUE
                                3$:     TYPE
                                        KMCM
                                4$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
                                        BR      OK              ;BR TO GET OUT
                                ERRPC:  1
                                        .BYTE   6,2
                                        SAVPC
                                CONTAB: 2
                                        .BYTE   6,4
                                        SAVR1
                                        .BYTE   6,2
                                        DMCSR
                                DEVTAB: .BYTE   7               ;DJ
                                        .BYTE   17              ;DH
                                        .BYTE   7               ;DQ
                                        .BYTE   7               ;DU
                                        .BYTE   7               ;DUP
                                        .BYTE   7               ;LK
                                        .BYTE   7               ;DMC
                                        .BYTE   7               ;DZ
                                        .BYTE   7               ;KMC
```



```
                                        .EVEN
                                AUDONE: TST     R5              ;DMC?
                                        BNE     1$              ;BR IF KMC AND ALL DONE
                                        MOV     #-1,R5          ;SET R5 TO -1 (KMC)
                                        MOV     #DM,MAP,R0      ;RESET R0 TO START OF TABLE
                                        BR      AUSTRT          ;GO DO KMC'S
                                1$:     MOV     (SP)+,@#6       ;RESTORE LOC 6
                                        MOV     (SP)+,@#4       ;RESTORE LOC 4
                                        BIT     #SW03,STRTSW    ;SELECT SPECIFIC DEVICES??
                                        BEQ     3$              ;BR IF NO.
                                        TYPE    ,MNEW           ;TYPE THE MESSAGE.
                                        CLR     R0              ;ZERO DATA LIGHTS
                                        HALT                    ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
                                        CMP     @SWR,SAVACT     ;IS THE NUMBER VALID?
                                        BLOS    2$              ;BR IF NUMBER IS OK.
                                        TYPE    ,MERR3          ;TELL USER OF INVALID NUMBER.
                                        HALT                    ;STOP EVERY THING.
                                        BR      .-2             ;RESTART THE PROGRAM AGAIN.
                                2$:     MOV     @SWR,DMACTV     ;GET NEW DEVICE PATTERN
                                        MOV     DMACTV,R0       ;SHOW THE USER WHAT HE SELECTED.
                                        HALT                    ;CONTINUE DYNAMIC SWITCHES.
                                3$:     MOV     #300,R0         ;PREPARE TO CLEAR THE FLOATING
                                        MOV     #302,R1         ;VECTOR AREA. 300-776
                                4$:     MOV     R1,(R0)+        ;START PUTTING "PC+2 = HALT"
                                        CLR     (R1)+           ;IN VECTOR AREA.
                                        CMP     (R0)+,(R1)+     ;POP POINTERS
                                        CMP     #1000,R0        ;ALL DONE??
                                        BNE     4$              ;BR IF NO.

                                ;TEST START AND RESTART
                                ;----------------------

                                .BEGIN: MOV     #STACK,SP       ;SET UP STACK
                                        MOV     @#6,-(SP)       ;SAVE LOC 6
                                        MOV     @#4,-(SP)       ;SAVE LOC 4
                                        CLR     R0              ;START AT 0
                                        MOV     #2$,@#4         ;SET UP FOR TIME OUT
                                        CLR     @#6             ;TO AUTOSIZE MEMORY
                                6$:     TST     (R0)+           ;CHECK ADDRESS IN R0
                                        CMP     #157776,R0      ;IS IT AT LEAST 28K
                                        BNE     6$              ;BR IF NO
                                        SUB     #7776,R0        ;SAVE 2K FOR MONITORS
                                7$:     MOV     R0,MEMLIM       ;STORE MEMORY LIMIT
                                        MOV     (SP)+,@#4       ;RESTORE LOC 4
                                        MOV     (SP)+,@#6       ;RESTORE LOC 6
                                        BR      10$             ;CONTINUE
                                2$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
                                        SUB     #4,R0           ;GET LAST GOOD ADDRESS
                                        SUB     #7776,R0        ;SAVE 2K FOR MONITORS
                                        CMP     #30000,R0       ;IS IT 8K?
                                        BNE     7$              ;BR IF NO
                                        MOV     #37400,R0       ;IF 8K DON'T SAVE 2K
                                        BR      7$              ;
                                10$:    MOV     #340,PS         ;LOCK OUT INTERRUPTS
                                        BIT     #BIT2,STRTSW    ;CHECK FOR LOCK ON TEST
                                        BEQ     1$              ;BR IF NO LOCK DESIRED.
```

```
003210  104402  005745                  TYPE    ,MLOCK          ;TYPE LOCK SELECTED.
003214  012737  000240  003522          MOV     #NOP,TTST       ;ADJUST SCOPE ROUTINE.
003222  012737  000240  003524          MOV     #NOP,TTST+2     ;SET UP TO LOCK
003230  000406                          BR      3$              ;CONTINUE ALONG.
003232  013737  003640  003522  1$:     MOV     BRW,TTST        ;PREPARE NORMAL SCOPE ROUTINE
003240  013737  003642  003524          MOV     BRX,TTST+2      ;LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
003246  012737  007620  001214  3$:     MOV     #CYCLE,RETURN   ;START AT "CYCLE" FIND WHICH DEVICE TO TEST
003254  032737  000002  001236  4$:     BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED?
003262  001002                          BNE     5$              ;BR IF YES
003264  104402  005657                  TYPE    ,MR             ;TYPE R
003270  000177  175720          5$:     JMP     @RETURN         ;START TESTING
```



```
                                        ;END OF PASS
                                        ;TYPE NAME OF TEST
                                        ;UPDATE PASS COUNT
                                        ;CHECK FOR EXIT TO ACT-11
                                        ;RESTART TEST

003274  000005                  ,EOP:   RESET                   ;MAKE THE  WORLD CLEAN AGAIN.
003276  005037  001234                  CLR     LSTERR          ;CLEAR LAST ERROR PC
003302  105037  001325                  CLRB    ERRFLG          ;CLEAR ERROR FLAG
003306  005237  001230                  INC     PASCNT          ;UPDATE PASS COUNT
003312  013777  001230  175660          MOV     PASCNT,@DISPLAY ;DISPLAY PASS COUNT
003320  104402  005635                  TYPE    ,MEPASS         ;TYPE END PASS
003324  104402  005734                  TYPE    ,MCSRX          ;TYPE CSR
003330  104411  003456                  CNVRT   ,XCSR           ;SHOW IT
003334  104402  006002                  TYPE    ,MVECX          ;TYPE VECTOR
003340  104411  003464                  CNVRT   ,XVEC           ;SHOW IT
003344  104402  006010                  TYPE    ,MPASSX         ;TYPE PASSES
003350  104411  003472                  CNVRT   ,XPASS          ;SHOW IT
003354  104402  006021                  TYPE    ,MERRX          ;TYPE ERRORS
003360  104411  003500                  CNVRT   ,XERR           ;SHOW IT
003364  013700  001322                  MOV     MILK,R0         ;GET POINTER TO PASS COUNT
003370  013720  001230                  MOV     PASCNT,(R0)+    ;STORE PASS COUNT FOR THIS DMC11
003374  013720  001232                  MOV     ERRCNT,(R0)+    ;STORE ERROR COUNT FOR THIS DMC11
003400  005337  001314                  DEC     SAVNUM          ;ARE ALL DEVICES TESTED?
003404  001017                          BNE     RESTRT          ;BR IF NO,
003406  112737  000377  001327          MOVB    #377,QV,FLG     ;SET THE QUICK VERIFY FLAG,
003414  013737  001310  001314          MOV     DMNUM,SAVNUM    ;RESTORE THE COUNT
003422  013701  000042                  MOV     @#42,R1         ;CHECK FOR ACT-11 OR DDP
003426  001406                          BEQ     RESTRT          ;IF NOT, CONTINUE TESTING
003430  000005                          RESET                   ;STOP THE SHOW--CLEAR THE WORLD
003432                          $ENDAD:
003432  004711                          JSR     PC,(R1)
003434  000240                          NOP
003436  000240                          NOP
003440  000240                          NOP
003442  000240                          NOP
003444  012737  007620  001214  RESTRT: MOV     #CYCLE,RETURN
003452  000137  007620                  JMP     CYCLE
003456  000001                  XCSR:   1
003460     006     002                  .BYTE   6,2
003462  001404                          DMCSR
003464  000001                  XVEC:   1
003466     004     002                  .BYTE   4,2
003470  001374                          DMRVEC
003472  000001                  XPASS:  1
003474     006     002                  .BYTE   6,2
003476  001230                          PASCNT
003500  000001                  XERR:   1
003502     006     002                  .BYTE   6,2
003504  001232                          ERRCNT

                                        ;SCOPE LOOP AND INTERATION HANDLER
                                        ;--------------------------------

003506  004737  007362          ,SCOPE: JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
003512  010046                          MOV     R0,(SP)         ;SAVE R0 ON THE STACK
```

```
003514  032777  040000  175460          BIT     #BIT14,@SWR     ;"LOOP ON THIS TEST"?
003522  001407                  TTST:   BEQ     1$              ;BR IF NO.  (IF LOCK SW01=1; THIS LOC =240)
003524  000437                          BR      3$              ;GOTO 3s    (IF LOCK SW01=1; THIS LOC =240)
003526  105777  175452                  TSTB    @TKCSR          ;KEYBOAD DONE?
003532  100034                          BPL     3$              ;BR IF NO. (LOCK: HIT KEY TO GOTO NEXT TEST)
003534  017700  175446                  MOV     @TKDBR,R0       ;CLEAR DONE BIT
003540  000415                          BR      2$              ;CONTINUE
003542  032777  004000  175432  1$:     BIT     #SW11,@SWR      ;DELETE ITERATION?  (QUICK PASS)
003550  001011                          BNE     2$              ;BR IF YES
003552  105737  001327                  TSTB    QV.FLG          ;HAVE PASSES BEECOMPLETED?
003556  001406                          BEQ     2$              ;BR IF QUICK PASS.
003560  005237  001224                  INC     LPCNT           ;UPDATE ITERATION COUNTER
003564  023737  001224  001222          CMP     LPCNT,ICOUNT    ;ARE ALL ITERATIONS DONE??
003572  101414                          BLOS    3$              ;BR IF NOT YET
003574  105037  001325          2$:     CLRB    ERRFLG          ;PREPARE FOR NEW TEST
003600  005037  001224                  CLR     LPCNT           ;START ICOUNTER AT 0
003604  005037  001220                  CLR     LOCK
003610  012737  000020  001222          MOV     #20,ICOUNT      ;RESET ITERATIONS
003616  013737  001216  001214          MOV     NEXT,RETURN     ;GET NEXT TEST
003624  011600                  3$:     MOV     (SP),R0         ;POP R0 OFF OF THE STACK
003626  022626                          POP2SP                  ;FAKE AN "RTI"
003630  013701  001404                  MOV     DMCSR,R1        ;R1 CONTAINS BASE DMC ADDRESS
003634  000177  175354                  JMP     @RETURN         ;GO DO THE TEST
003640  001407                  BRW:    1407
003642  000437                  BRX:    437

                                        ;CHECK FOR FREEZE ON CURRENT DATA
                                        ;--------------------------------

003644  004737  007362          .SCOP1: JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
003650  032777  001000  175324          BIT     #SW09,@SWR      ;IS SW09=1(SET)?
003656  001405                          BEQ     1$              ;BR IF NOT SET.
003660  005737  001220                  TST     LOCK
003664  001402                          BEQ     1$
003666  013716  001220                  MOV     LOCK,(SP)       ;GOTO THE ADDRESS IN LOCK.
003672  000002                  1$:     RTI                     ;GO BACK.

                                        ;TELETYPE OUTPUT ROUTINE
                                        ;-----------------------

003674  010546                  .TYPE:  MOV     R5,-(SP)        ;SAVE R5 ON THE STACK.
003676  017605  000002                  MOV     @2(SP),R5       ;GET ADDRESS OF MESSAGE.
003702  062766  000002  000002          ADD     #2,2(SP)        ;POP OVER ADDRESS.
003710  005737  007556          4$:     TST     SWFLG           ;SOFT SWR MESSAGE?
003714  001004                          BNE     1$              ;IF YES TYPE IT OUT REGARDLESS OF SW12
003716  032777  010000  175256          BIT     #SW12,@SWR      ;INHIBIT ALL PRINT OUT??
003724  001012                          BNE     3$              ;BR IF NO PRINT OUT WANTED (SW12=1)
003726  105715                  1$:     TSTB    (R5)            ;IS NUMBER MINUS? (MSB=1(BIT7))
003730  100002                          BPL     2$              ;BR IF NUMBER IS PLUS
003732  104402  005574                  TYPE    ,MCRLF          ;TYPE A CR/LF!
003736  105777  175246          2$:     TSTB    @TPCSR          ;TTY READY?
003742  100375                          BPL     2$              ;BR IF NO.
003744  112577  175242                  MOVB    (R5)+,@TPDBR    ;PRINT CURRENT CHAR.
003750  001357                          BNE     4$              ;IF NOT ZERO KEEP PRINTING!
003752  012605                  3$:     MOV     (SP)+,R5        ;END OF OUTPUT. RESTORE R5
003754  000002                          RTI                     ;GO HOME
```



```
                                        ;--------------------------

003756  010346                  .INSTR: MOV     R3,-(SP)        ;SAVE R3 ON STACK
003760  010446                          MOV     R4,-(SP)        ;SAVE R4 ON STACK
003762  017637  000004  004000          MOV     @4(SP),.MSG
003770  062766  000002  000004          ADD     #2,4(SP)
003776  104402                  .INST1: TYPE
004000  000000                  .MSG:   0
004002  012704  007256                  MOV     #INBUF,R4
004006  012703  000007                  MOV     #7,R3
004012  105777  175166          1$:     TSTB    @TKCSR
004016  100375                          BPL     1$
004020  117714  175162                  MOVB    @TKDBR,(R4)
004024  142714  000200                  BICB    #200,(R4)
004030  122427  000015                  CMPB    (R4)+,#15
004034  001417                          BEQ     INSTR2
004036  105777  175146          2$:     TSTB    @TPCSR
004042  100375                          BPL     2$
004044  017777  175136  175140          MOV     @TKDBR,@TPDBR
004052  005303                          DEC     R3
004054  001356                          BNE     1$
004056  012604                          MOV     (SP)+,R4
004060  012603                          MOV     (SP)+,R3
004062  104402  005570          .INSTE: TYPE    ,MQM
004066  010346                          MOV     R3,-(SP)
004070  010446                          MOV     R4,-(SP)
004072  000741                          BR      .INST1
004074  012604                  INSTR2: MOV     (SP)+,R4        ;RESTORE R4
004076  012603                          MOV     (SP)+,R3        ;RESTORE R3
004100  000002                          RTI

                                        ;CONVERT ASCII STRING TO OCTAL
                                        ;-----------------------------

004102  010546                  .PARAM: MOV     R5,-(SP)
004104  010446                          MOV     R4,-(SP)
004106  016605  000004                  MOV     4(SP),R5
004112  012537  004272                  MOV     (R5)+,LOLIM
004116  012537  004274                  MOV     (R5)+,HILIM
004122  012537  004276                  MOV     (R5)+,DEVADR
004126  112537  004300                  MOVB    (R5)+,LOBITS
004132  112537  004301                  MOVB    (R5)+,ADRCNT
004136  010566  000004                  MOV     R5,4(SP)
004142  005005                  PARAM1: CLR     R5
004144  012704  007256                  MOV     #INBUF,R4
004150  122714  000015                  CMPB    #15,(R4)
004154  001420                          BEQ     PARERR
004156  121427  000060          1$:     CMPB    (R4),#60
004162  002415                          BLT     PARERR
004164  121427  000067                  CMPB    (R4),#67
004170  003012                          BGT     PARERR
004172  142714  000060                  BICB    #60,(R4)
004176  152405                          BISB    (R4)+,R5
004200  122714  000015                  CMPB    #15,(R4)
004204  001406                          BEQ     LIMITS
004206  006305                          ASL     R5
```

```
873  004210  006305                          ASL     R5
874  004212  006305                          ASL     R5
875  004214  000760                          BR      1$
876  004216  104404                  PARERR: INSTER
877  004220  000750                          BR      PARAM1
878
879                                          ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
880                                          ;--------------------------------------
881
882  004222  020537  004274          LIMITS: CMP     R5,HILIM
883  004226  101373                          BHI     PARERR
884  004230  020537  004272                  CMP     R5,LOLIM
885  004234  103770                          BLO     PARERR
886  004236  133705  004300                  BITB    LOBITS,R5
887  004242  001365                          BNE     PARERR
888
889                                          ;STORE NUMBER AT SPECIFIED ADDRESS
890
891  004244  013704  004276                  MOV     DEVADR,R4
892  004250  010524                  1$:     MOV     R5,(R4)+
893  004252  062705  000002                  ADD     #2,R5
894  004256  105337  004301                  DECB    ADRCNT
895  004262  001372                          BNE     1$
896  004264  012604                          MOV     (SP)+,R4
897  004266  012605                          MOV     (SP)+,R5
898  004270  000002                          RTI
899  004272  000000                  LOLIM:  0
900  004274  000000                  HILIM:  0
901  004276  000000                  DEVADR: 0
902  004300  000000                  LOBITS: 0
903           004301                 ADRCNT=LOBITS+1
904
905                                          ;SAVE PC OF TEST THAT FAILED AND R0-R5
906                                          ;--------------------------------------
907
908  004302  016637  000004  001276  .SAV05: MOV     4(SP),SAVPC     ;SAVE R7 (PC)
909
910                                          ;SAVE R0-R5
911
912  004310  010537  001272          SV05:   MOV     R5,SAVR5        ;SAVE R5
913  004314  010437  001270                  MOV     R4,SAVR4        ;SAVE R4
914  004320  010337  001266                  MOV     R3,SAVR3        ;SAVE R3
915  004324  010237  001264                  MOV     R2,SAVR2        ;SAVE R2
916  004330  010137  001262                  MOV     R1,SAVR1        ;SAVE R1
917  004334  010037  001260                  MOV     R0,SAVR0        ;SAVE R0
918  004340  000002                          RTI                     ;LEAVE.
919
920                                          ;RESTORE R0-R5
921
922  004342  013700  001260          .RES05: MOV     SAVR0,R0        ;RESTORE R0
923  004346  013701  001262                  MOV     SAVR1,R1        ;RESTORE R1
924  004352  013702  001264                  MOV     SAVR2,R2        ;RESTORE R2
925  004356  013703  001266                  MOV     SAVR3,R3        ;RESTORE R3
926  004362  013704  001270                  MOV     SAVR4,R4        ;RESTORE R4
927  004366  013705  001272                  MOV     SAVR5,R5        ;RESTORE R5
928  004372  000002                          RTI                     ;LEAVE
```



```
929
930                                          ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
931                                          ;------------------------------------------------------
932
933  004374  104402  005574          .CONVR: TYPE    ,MCRLF
934  004400  010046                  .CNVRT: MOV     R0,-(SP)
935  004402  010146                          MOV     R1,-(SP)
936  004404  010346                          MOV     R3,-(SP)
937  004406  010446                          MOV     R4,-(SP)
938  004410  010546                          MOV     R5,-(SP)
939  004412  017601  000012                  MOV     @12(SP),R1
940  004416  062766  000002  000012          ADD     #2,12(SP)
941  004422  012137  004616                  MOV     (R1)+,WRDCNT
942  004430  112137  004620          1$:     MOVB    (R1)+,CHRCNT
943  004434  112137  004621                  MOVB    (R1)+,SPACNT
944  004440  013137  004622                  MOV     @(R1)+,BINWRD
945  004444  122737  000003  004620          CMPB    #3,CHRCNT
946  004452  001003                          BNE     2$
947  004454  042737  177400  004622          BIC     #177400,BINWRD
948  004462  013704  004622          2$:     MOV     BINWRD,R4
949  004466  113705  004620                  MOVB    CHRCNT,R5
950  004472  012700  001416                  MOV     #TEMP,R0
951  004476  010403                  3$:     MOV     R4,R3
952  004500  042703  177770                  BIC     #177770,R3
953  004504  062703  000060                  ADD     #060,R3
954  004510  110320                          MOVB    R3,(R0)+
955  004512  000241                          CLC
956  004514  006004                          ROR     R4
957  004516  000241                          CLC
958  004520  006004                          ROR     R4
959  004522  000241                          CLC
960  004524  006004                          ROR     R4
961  004526  005305                          DEC     R5
962  004530  001362                          BNE     3$
963  004532  012703  007320                  MOV     #MDATA,R3
964  004536  114023                  4$:     MOVB    -(R0),(R3)+
965  004540  105337  004620                  DECB    CHRCNT
966  004544  001374                          BNE     4$
967  004546  105737  004621                  TSTB    SPACNT
968  004552  001405                          BEQ     6$
969  004554  112723  000040          5$:     MOVB    #040,(R3)+
970  004560  105337  004621                  DECB    SPACNT
971  004564  001373                          BNE     5$
972  004566  105013                  6$:     CLRB    (R3)
973  004570  104402  007320                  TYPE    ,MDATA
974  004574  005337  004616                  DEC     WRDCNT
975  004600  001313                          BNE     1$
976  004602  012605                          MOV     (SP)+,R5
977  004604  012504                          MOV     (SP)+,R4
978  004606  012603                          MOV     (SP)+,R3
979  004610  012601                          MOV     (SP)+,R1
980  004612  012600                          MOV     (SP)+,R0
981  004614  000002                          RTI
982  004616  000000                  WRDCNT: 0
983  004620  000000                  CHRCNT: 0
984           004621                 SPACNT=CHRCNT+1
```

```
  985  004622  000000                        BINWRD: 0
  986
  987
  988                                                ;TRAP DISPATCH SERVICE
  989                                                ;ARGUMENT OF TRAP IS EXTRACTED
  990                                                ;AND USED AS OFFSET TO OBTAIN POINTER
  991                                                ;TO SELECTED SUBROUTINE
  992
  993  004624  011646                        .TRPSR: MOV     (SP),-(SP)      ;GET PC OF RETURN
  994  004626  162716  000002                        SUB     #2,(SP)         ;=PC OF TRAP
  995  004632  017616  000000                        MOV     @(SP),(SP)      ;GET TRP
  996  004636  006316                        TRPOK:  ASL     (SP)            ;MULTIPLY TRAP ARG BY 2
  997  004640  042716  177001                        BIC     #177001,(SP)    ;CLEAR UNWANTED BITS
  998  004644  062716  001330                        ADD     #.TRPTAB,(SP)   ;POINTER TO SUBROUTINE ADDRESS
  999  004650  017616  000000                        MOV     @(SP),(SP)      ;SUBROUTINE ADDRESS
 1000  004654  000136                        JMP     @(SP)+          ;GO TO SUBROUTINE
 1001
 1002                                                ;ERROR HANDLER
 1003                                                ;-------------
 1004
 1005  004656  004737  007362                .HLT:   JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
 1006  004662  032777  010000  174312                BIT     #SW12,@SWR      ;BELL ON ERROR?
 1007  004670  001406                        BEQ     XBX             ;BR IF NO BELL
 1008  004672  105777  174312                TSTB    @TPCSR          ;TTY READY,
 1009  004676  100003                        BPL     XBX             ;DON'T WAIT IF TTY NOT READY,
 1010  004700  112777  000207  174304        MOVB    #207,@TPDBR     ;PUSH A BELL AT THE TTY,
 1011  004706  032777  020000  174266        XBX:    BIT     #SW13,@SWR      ;DELETE ERROR PRINT OUT?
 1012  004714  001105                        BNE     HALTS           ;BR IF NO PRINT OUT WANTED,
 1013  004716  021637  001234                CMP     (SP),LSTERR     ;WAS THIS ERROR FOUND LAST TIME?
 1014  004722  001404                        BEQ     1$              ;BR IF YES
 1015  004724  011637  001234                MOV     (SP),LSTERR     ;RECORD BEING HERE
 1016  004730  105037  001325                CLRB    ERRFLG          ;PREPARE HEADER
 1017  004734  104406                        1$:     SAV05           ;SAVE ALL PROC REGISTERS
 1018  004736  011605                        MOV     (SP),R5         ;GET THE PC OF ERROR
 1019  004740  162705  000002                SUB     #2,R5           ;GET ADDRESS OF TRAP CALL
 1020  004744  011504                        MOV     (R5),R4         ;GET HLT INSTRUCTION
 1021  004746  006304                        ASL     R4              ;MULT BY TWO
 1022  004750  061504                        ADD     (R5),R4         ;DOUBLE IT
 1023  004752  006304                        ASL     R4              ;MULT AGAIN
 1024  004754  042704  177001                BIC     #177001,R4      ;CLEAR JUNK
 1025  004760  062704  037270                ADD     #.ERRTAB,R4     ;GET POINTER
 1026  004764  012437  005100                MOV     (R4)+,ERRMSG    ;GET ERROR MESSAGE
 1027  004770  012437  005112                MOV     (R4)+,DATAHD    ;GET DATA HEADRER
 1028  004774  011437  005124                MOV     (R4),DATABP     ;GET DATA TABLE
 1029  005000  105737  001325                TSTB    ERRFLG          ;TYPE HEADREER
 1030  005004  001403                        BEQ     TYPMSG          ;BR IF YES
 1031  005006  005737  005124                TST     DATABP          ;DOES DATA TABLE EXIST?
 1032  005012  001040                        BNE     TYPDAT          ;BR IF YES,
 1033  005014  104402  005574                TYPMSG: TYPE    ,MCRLF
 1034  005020  104402  005574                TYPE    ,MCRLF
 1035  005024  005737  001220                TST     LOCK
 1036  005030  001402                        BEQ     1$
 1037  005032  104402  006044                TYPE    ,MASTEK
 1038  005036  104402  006032                1$:     TYPE    ,MTSTN
 1039  005042  104411  005232                CNVRT   ,XTSTN          ;SHOW IT
 1040  005046  104402  006121                TYPE    ,MERRPC         ;TYPE PC,
```



```
 1041  005052  104411  005224                CNVRT   ,ERTABO         ;SHOW IT
 1042  005056  104402  005574                TYPE    ,MCRLF          ;GIVE A CR/LF
 1043  005062  112737  177777  001325        MOVB    #-1,ERRFLG      ;NO MORE HEADER UNLESS NO DATA TABLE,
 1044  005070  005737  005100                TST     ERRMSG          ;IS THERE AN ERROR MESSAGE?
 1045  005074  001402                        BEQ     WRKO,FM         ;BR IF NO,
 1046  005076  104402                        TYPE                    ;TYPE
 1047  005100  000000                        ERRMSG: 0               ;       ERROR MESSAGE
 1048  005102                                WRKO,FM:                ;
 1049  005102  005737  005112                TST     DATAHD          ;DATA HEADER?
 1050  005106  001402                        BEQ     TYPDAT          ;BR IF NO
 1051  005110  104402                        TYPE                    ;TYPE
 1052  005112  000000                        DATAHD: 0               ;       DATA HEADER
 1053  005114  005737  005124                TYPDAT: TST     DATABP          ;DATA TABLE?
 1054  005120  001402                        BEQ     RESREG          ;BR IF NO,
 1055  005122  104410                        CONVRT                  ;SHOW
 1056  005124  000000                        DATABP: 0               ;       DATA TABLE
 1057  005126  104407                        RESREG: RES05           ;RESTORE PROC REGISTERS
 1058  005130  022737  003432  000042        HALTS:  CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, HALT!!
 1059  005136  001403                        BEQ     1$
 1060  005140  005777  174036                TST     @SWR            ;HALT ON ERROR?
 1061  005144  100005                        BPL     EXITER          ;BR IF NO HALT ON ERROR
 1062  005146  010046                        1$:     PUSHR0          ;SAVE R0
 1063  005150  016600  000002                MOV     2(SP),R0        ;SHOW ERROR PC IN DATA LIGHTS
 1064  005154  000000                        HALT                    ;HALT
 1065  005156  012600                        POPR0                   ;GET R0
 1066  005160  005237  001232                EXITER: INC     ERRCNT          ;UPDATE ERROR COUNT
 1067  005164  032777  000400  174010        BIT     #SW08,@SWR      ;GOTO TOP OF TEST?
 1068  005172  001007                        BNE     1$              ;BR IF YES
 1069  005174  032777  002000  174000        BIT     #SW10,@SWR      ;GOTO NEXT TEST?
 1070  005202  001407                        BEQ     2$              ;BR IF NO
 1071  005204  013737  001216  001214        MOV     NEXT,RETURN     ;SET FOR NEXT TEST
 1072  005212  012706  001200                1$:     MOV     #STACK,SP       ;RESET SP
 1073  005216  000177  173772                JMP     @RETURN         ;GOTO SPECIFIED TEST
 1074  005222  000002                        2$:     RTI             ;RETURN
 1075  005224  000001                        ERTABO: 1
 1076  005226     006     002                        .BYTE   6,2
 1077  005230  001276                        SAVPC
 1078  005232  000001                        XTSTN:  1
 1079  005234     003     002                        .BYTE   3,2
 1080  005236  001226                        TSTNO
 1081                                                ;ENTER HERE ON POWER FAILURE
 1082                                                ;--------------------------
 1083
 1084
 1085  005240                                .PFAIL:
 1086  005240  012737  005252  000024                MOV     #RESTART,24             ;SET UP FOR POWER UP TRAP
 1087  005246  000000                        HALT                            ;HALT ON POWER DOWN NORMAL
 1088  005250  000777                        BR      .
 1089
 1090                                                ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
 1091
 1092  005252                                RESTAR:
 1093  005252  012737  005240  000024                MOV     #.PFAIL,24      ;SET UP FOR POWER FAILURE
 1094  005260  012706  001200                MOV     #STACK,SP       ;RESET THE STACK POINTER
 1095  005264  013701  001404                MOV     DMCSR,R1        ;RESTORE R1
 1096  005270  005037  001416                CLR     TEMP            ;READY FOR TIMMER
```

```
1097  005274  005237  001416                    INC     TEMP            ;PLUS ONE TO THE TIMER!
1098  005300  001375                            BNE     .-4             ;BR IF MORE TO GO
1099  005302  104402  005577                    TYPE    ,MPFAIL         ;TYPE THE MESSAGE
1100  005306  104411  005332                    CNVRT   ,PFTAB          ;TELL WHAT TEST TO RETURN TO,
1101  005312  105037  001325                    CLRB    ERRFLG          ;START CLEAN
1102  005316  005037  001234                    CLR     LSTERR          ;**********
1103  005322  005011                            CLR     (R1)            ;CLEAR MAINT BITS
1104  005324  104412                            MSTCLR                  ;START CLEAN UP OF DEVICE
1105  005326  000177  173662                    JMP     @RETURN         ;START DOING THAT TEST AGAIN,
1106  005332  000001                    PFTAB:  1
1107  005334     003     002             .BYTE   3,2
1108  005336  001226                            TSTNO
1109
1110  005340                             .DELAY:
1111  005340  012777  000020  174044            MOV     #20,@DMPO4
1112  005346  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1113  005350  121111                            121111                  ;POKE CLOCK DELAY BIT
1114  005352                             1$:
1115  005352  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1116  005354  121224                            121224                  ;PORT4_IBUS*11
1117  005356  032777  000020  174026            BIT     #BIT4,@DMPO4    ;IS CLOCK BIT SET?
1118  005364  001772                            BEQ     1$              ;BR IF NO
1119  005366  000002                            RTI
1120
1121  005370                             .MSTCLR:
1122  005370  152777  000100  174010            BISB    #BIT6,@DMCSRH   ;SET MASTER CLEAR
1123  005376  142777  000300  174002            BICB    #BIT6!BIT7,@DMCSRH  ;CLEAR MASTER CLEAR AND RUN
1124  005404  000002                            RTI                     ;RETURN
1125
1126  005406                             .ROMCLK:
1127  005406  152777  000002  173772            BISB    #BIT1,@DMCSRH   ;SET ROMI
1128  005414  013677  173774                    MOV     @(SP)+,@DMPO6   ;LOAD INSTRUCTION IN SEL6
1129  005420  062746  000002                    ADD     #2,-(SP)        ;ADJUST STACK
1130  005424  032777  000100  173550            BIT     #SW06,@SWR      ;HALT IF SW06 =1
1131  005432  001401                            BEQ     1$              ;BR IF SW06 =0
1132  005434  000000                            HALT                    ;HALT BEFORE CLOCKING INSTRUCTION
1133  005436  152777  000003  173742    1$:     BISB    #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
1134  005444  142777  000007  173734            BICB    #BIT2!BIT1!BIT0,@DMCSRH  ;CLEAR ROMO, ROMI, STEP
1135  005452  000002                            RTI
1136
1137  005454                             .DATACLK:
1138  005454  013637  001416                    MOV     @(SP)+,TEMP     ;PUT TICK COUNT IN TEMP
1139  005460  062746  000002                    ADD     #2,-(SP)        ;ADJUST STACK
1140  005464  152777  000020  173714    1$:     BISB    #BIT4,@DMCSRH   ;SET STEP LU
1141  005472  027777  173706  173704            CMP     @DMCSR,@DMCSR   ;WASTE TIME
1142  005500  142777  000020  173700            BICB    #BIT4,@DMCSRH   ;CLEAR STEP LU
1143  005506  005337  001416                    DEC     TEMP            ;DEC TICK COUNT
1144  005512  001364                            BNE     1$              ;BR IF NOT DONE
1145  005514  000002                            RTI                     ;RETURN
1146  005516  000001                    3$:     .BLKW 1
1147
1148  005520                             .TIMER:
1149  005520  013637  001416                    MOV     @(SP)+,TEMP     ;MOVE COUNT TO TEMP
1150  005524  062746  000002                    ADD     #2,-(SP)        ;ADJUST STACK
1151  005530                             1$:
1152  005530  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```



```
1153  005532  021364                            021364                  ;PORT4_IBUS* REG11
1154  005534  032777  000002  173650            BIT     #2,@DMPO4       ;IS PGM CLOCK BIT CLEAR?
1155  005542  001772                            BEQ     1$              ;BR IF YES
1156  005544                             2$:
1157  005544  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1158  005546  021364                            021364                  ;PORT4_IBUS* REG11
1159  005550  032777  000002  173634            BIT     #2,@DMPO4       ;IS PGM CLOCK BIT SET?
1160  005556  001372                            BNE     2$              ;BR IF YES
1161  005560  005337  001416                    DEC     TEMP            ;DEC COUNT
1162  005564  001361                            BNE     1$              ;BR IF NOT DONE
1163  005566  000002                            RTI                     ;RETURN
1164
1165  005570  020040  000077            NOM:    .ASCIZ  / ?/
 (2)  005574  005015     000            NCRLF:  .ASCIZ  <15><12>
 (2)  005577     377  053520  020122    MPFAIL: .ASCIZ  <377>/PWR FAILED, RESTART AT TEST /
 (2)  005635     377  047105  020104    MEPASS: .ASCIZ  <377>/END PASS DZDMH  /
 (2)  005657     377  000122            MR:     .ASCIZ  <377>/R/
 (2)  005662  047377  020117  042504    MERR2:  .ASCIZ  <377>/NO DEVICES PRESENT,/
 (2)  005707     377  047111  052523    MERR3:  .ASCIZ  <377>/INSUFFICIENT DATA!/
 (2)  005733     377  042524  052123    MTSTPC: .ASCIZ  <377>/TEST PC=/
 (2)  005745     377  047514  045503    MLOCK:  .ASCIZ  <377>/LOCK ON SELECTED TEST/
 (2)  005774  051503  035122  000040    MCSRX:  .ASCIZ   /CSR: /
 (2)  006002  042526  035103  000040    MVECX:  .ASCIZ  /VEC: /
 (2)  006010  040520  051523  051505    MPASSX: .ASCIZ  /PASSES: /
 (2)  006021     105  051122  051117    MERRX:  .ASCIZ  /ERRORS: /
 (2)  006032  042524  052123  047040    MTSTN:  .ASCIZ  /TEST NO: /
 (2)  006044  000052                    MASTEK: .ASCIZ  /*/
 (2)  006046  051777  052105  051440    MNEW:   .ASCIZ  <377>/SET SWITCH REG TO DMC11'S DESIRED ACTIVE,/
 (2)  006121     120  035103  000040    MERRPC: .ASCIZ  /PC: /
 (2)  006126  020212  020040  020040    XHEAD:  .ASCII  <212>/           MAP OF DMC11 STATUS/
 (2)  006165     377  020040  020040            .ASCII  <377>/           ------------------/
 (2)  006224  020212  050040  020103            .ASCII  <212>/  PC      CSR     STAT1    STAT2    STAT3/
 (2)  006276  026777  026455  026455            .ASCIZ  <377>/------   ------   ------   ------   ------/
 (2)  006352  044377  053517  046440    NUM:    .ASCIZ  <377>/HOW MANY DMC11'S TO BE TESTED?/
 (2)  006412  041777  051123  040440    CSR:    .ASCIZ  <377>/CSR ADDRESS?/
 (2)  006430  053377  041505  047524    VEC:    .ASCIZ  <377>/VECTOR ADDRESS?/
 (2)  006451     377  051102  050040    PRIO:   .ASCIZ  <377>/BR PRIORITY LEVEL? (4,5,6,7)?/
 (2)  006510  044777  020106  046504    CRAM:   .ASCIZ  <377>/IF DMC HAS CRAM (M8204) TYPE "Y", IF CROM (M8200) TYPE "N"
 (2)  006606  053777  044510  044103    MODU:   .ASCIZ  <377>/WHICH LINE UNIT? IF NONE TYPE "N", IF M8201 TYPE "1", IF M
 (2)  006720  051777  044527  041524    LINE:   .ASCIZ  <377>/SWITCH PAC#1 (DDCMP LINE #)?/
 (2)  006756  051777  044527  041524    BM:     .ASCIZ  <377>/SWITCH PAC#2 (BM873 BOOT ADD)?/
 (2)  007016  044777  020123  044124    CONN:   .ASCIZ  <377>/IS THE LOOP BACK CONNECTOR ON?/
 (2)  007056  047377  020117  042504    NOACT:  .ASCIZ  <377>/NO DEVICES ARE SELECTED/
 (2)  007107     377  051412  051127    SWMES:  .ASCIZ  <377><12>/SWR= /
 (2)  007117     116  053505  020077    SWMES1: .ASCIZ  /NEW? /
 (2)  007125     377  042377  041515    CONERR: .ASCIZ  <377><377>/DMC11 CONFIGURATION ERROR  PC: /
 (2)  007167     377  054105  042520    CNERR:  .ASCIZ  <377>/EXPECTED  FOUND/
 (2)  007210  024040  046504  024503    DMCM:   .ASCIZ  / (DMC) /
 (2)  007220  024040  046513  024503    KMCM:   .ASCIZ  / (KMC) /
 (2)                                    .EVEN
 (2)  007230  000005                    XSTATQ: 5
1166  007232     006     003                    .BYTE   6,3
1167  007234  001246                            TEMP1
1168  007236     006     003                    .BYTE   6,3
1169  007240  001250                            TEMP2
1170  007242     006     003                    .BYTE   6,3
```

```
1171  007244  001252                          TEMP3
1172  007246     006     003                  .BYTE   6,3
1173  007250  001254                          TEMP4
1174  007252     006     002                  .BYTE   6,2
1175  007254  001256                          TEMP5
1176                                  .EVEN
1177
1178                                          ;BUFFERS FOR INPUT-OUTPUT
1179
1180  007256  000000                  INBUF:  0
1181          007320                  .=.+40
1182  007320  000000                  MDATA:  0
1183          007362                  .=.+40
1184
1185
1186                                          ;ROUTINE USED TO CHANGE SOFTWARE SWITCH
1187                                          ;REGISTER USING THE CONSOLE TERMINAL
1188                                          ;--------------------------------------
1189
1190  007362  022737  000176  001202  CKSWR:  CMP     #SWREG,SWR      ;IS THE SOFT SWR BEING USED?
1191  007370  001071                          BNE     CKSWR5          ;BR IF NO
1192  007372  022777  000007  171606          CMP     #7,@TKDBR       ;WAS CTRL G TYPED? (7 BIT ASCII)
1193  007400  001404                          BEQ     1$              ;BR IF YES
1194  007402  022777  000207  171576          CMP     #207,@TKDBR     ;WAS CTRL G TYPED? (8 BIT ASCII)
1195  007410  001061                          BNE     CKSWR5          ;BR IF NO
1196  007412  010246                  1$:     MOV     R2,-(SP)        ;STORE R2
1197  007414  010346                          MOV     R3,-(SP)        ;STORE R3
1198  007416  010446                          MOV     R4,-(SP)        ;STORE R4
1199  007420  012737  177777  007556          MOV     #-1,SWFLG       ;SET SOFT TYPE OUT FLAG
1200  007426  005002                  CKSWR1: CLR     R2              ;CLEAR NEW SWR CONTENTS
1201  007430  012704  177777                  MOV     #-1,R4          ;SET FLAG TO ALL ONES
1202  007434  104402  007107                  TYPE    ,SWMES          ;TYPE "SWR= "
1203  007440  104411                  CKSWR2: CNVRT                   ;TYPE OUT PRESENT CONTENTS
1204  007442  007612                          SOFTSW                  ;OF SOFT SWITCH REGISTER
1205  007444  104402  007117          CKSWR3: TYPE    ,SWMES1         ;TYPE "NEW? "
1206  007450  004737  007560          CKSWR4: JSR     PC,INCHAR       ;GET RESPONSE
1207  007454  022703  000015                  CMP     #15,R3          ;WAS IT A CR?
1208  007460  001424                          BEQ     5$              ;BR IF YES
1209  007462  022703  000012                  CMP     #12,R3          ;WAS IT A LF?
1210  007466  001416                          BEQ     4$              ;BR IF YES
1211  007470  022703  000025                  CMP     #25,R3          ;WAS IT CTRL U?
1212  007474  001754                          BEQ     CKSWR1          ;BR IF YES(START OVER)
1213  007476  022703  000007                  CMP     #7,R3           ;IF CNTL G GET NEXT CHAR
1214  007502  001762                          BEQ     CKSWR4
1215  007504  005004                          CLR     R4              ;IT MUST BE A DIGIT SO CLR FLAG
1216  007506  042703  177770                  BIC     #177770,R3      ;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
1217  007512  006302                          ASL     R2              ;SHIFT R2 3 TIMES
1218  007514  006302                          ASL     R2
1219  007516  006302                          ASL     R2
1220  007520  050302                          BIS     R3,R2           ;ADD LAST DIGIT
1221  007522  000752                          BR      CKSWR4          ;GET NEXT CHARACTER
1222  007524  012766  002002  000006  4$:     MOV     #,START,6(SP)   ;LF WAS TYPED SO GO TO START
1223  007532  005704                  5$:     TST     R4              ;IS FLAG CLEAR?
1224  007534  001002                          BNE     6$              ;IF NOT DON'T CHANGE SOFT SWR
1225  007536  010277  171440                  MOV     R2,@SWR         ;IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
1226  007542  005037  007556          6$:     CLR     SWFLG           ;CLEAR TYPEOUT FLAG
```



```
1227  007546  012604                          MOV     (SP)+,R4        ;RESTORE R4
1228  007550  012603                          MOV     (SP)+,R3        ;RESTORE R3
1229  007552  012602                          MOV     (SP)+,R2        ;RESTORE R2
1230  007554  000207                  CKSWR5: RTS     PC              ;RETURN
1231
1232  007556  000000                  SWFLG:  0
1233
1234  007560  105777  171420          INCHAR: TSTB    @TKCSR
1235  007564  100375                          BPL     .-4
1236  007566  017703  171414                  MOV     @TKDBR,R3
1237  007572  105777  171412                  TSTB    @TPCSR
1238  007576  100375                          BPL     .-4
1239  007600  010377  171406                  MOV     R3,@TPDBR
1240  007604  042703  000200                  BIC     #BIT7,R3
1241  007610  000207                          RTS     PC
1242
1243  007612  000001                  SOFTSW: 1
1244  007614     006     002                  .BYTE   6,2
1245  007616  000176                          SWREG
```

```
                                                ;
                                                ;ROUTINE USED TO "CYCLE" THROUGH UP TO 16 DMC11'S
                                                ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
                                                ;AND RUNS THE SPECIFIED DMC11'S.   THIS ROUTINE *MUST*
                                                ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
                                                ;SETUP NECESSARY.
                                                ;
007620  005737  001306          CYCLE:  TST     DMACTV          ;ARE ANY DMC11'S TO BE TESTED?
007624  001004                          BNE     1$              ;BR IF OK.
007626  104402  007056                  TYPE    ,NOACT          ;NO DMC11'S SELECTED!!
007632  000000                          HALT                    ;STOP THE SHOW.
007634  000776                          BR      .-2             ;DISQUALIFY CONT. SW.
007636  000241                  1$:     CLC                     ;CLEAR PROC. CARRY BIT.
007640  006137  001316                  ROL     RUN             ;UPDATE POINTER
007644  005537  001316                  ADC     RUN             ;CATCH CARRY FROM RUN
007650  062737  000004  001322          ADD     #4,MILK         ;UPDATE POINTER
007656  062737  000010  001320          ADD     #10,CREAM       ;UPDATE ADDRESS POINTER.
007664  022737  001700  001320          CMP     #DM.MAP+200,CREAM
007672  001006                          BNE     2$              ;KEEP GOING; NOT ALL TESTED FOR.
007674  012737  001500  001320          MOV     #DM.MAP,CREAM   ;RESET ADDRESS POINTER.
007702  012737  001702  001322          MOV     #CNT.MAP,MILK   ;RESET PASS COUNT POINTER
007710  033737  001316  001306  2$:     BIT     RUN,DMACTV      ;IS THIS ONE ACTIVE?
007716  001747                          BEQ     1$              ;BR IF NO
007720  013700  001320                  MOV     CREAM,R0        ;GET ADDRESS POINTER
007724  013702  001322                  MOV     MILK,R2         ;GET PASS COUNT POINTER
007730  012037  001404                  MOV     (R0)+,DMCSR     ;LOAD SYSTEM CTRL. REG
007734  011037  001374                  MOV     (R0),DMRVEC     ;LOAD VECTOR
007740  042737  177000  001374          BIC     #177000,DMRVEC  ;CLEAR UNWANTED BITS
007746  012037  001366                  MOV     (R0)+,STAT1     ;LOAD STAT1
007752  012037  001370                  MOV     (R0)+,STAT2     ;LOAD STAT2
007756  012037  001372                  MOV     (R0)+,STAT3     ;LOAD STAT3
007762  012237  001230                  MOV     (R2)+,PASCNT    ;LOAD PASS COUNT
007766  012237  001232                  MOV     (R2)+,ERRCNT    ;LOAD ERROR COUNT
007772  012700  000002                  MOV     #2,R0           ;SAVE CORE THIS WAY!
007776  013737  001404  001406          MOV     DMCSR,DMCSRH
010004  005237  001406                  INC     DMCSRH
010010  013737  001406  001410          MOV     DMCSRH,DMCTL
010016  005237  001410                  INC     DMCTL
010022  013737  001410  001412          MOV     DMCTL,DMPO4
010030  060037  001412                  ADD     R0,DMPO4
010034  013737  001412  001414          MOV     DMPO4,DMPO6
010042  060037  001414                  ADD     R0,DMPO6

010046  013737  001374  001376          MOV     DMRVEC,DMRLVL   ;PTY LVL
010054  060037  001376                  ADD     R0,DMRLVL       ;
010060  013737  001376  001400          MOV     DMRLVL,DMTVEC   ;TX VEC
010066  060037  001400                  ADD     R0,DMTVEC       ;
010072  013737  001400  001402          MOV     DMTVEC,DMTLVL   ;TX LVL
010100  060037  001402                  ADD     R0,DMTLVL

010104  032737  000002  001236          BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED
010112  001450                          BEQ     7$              ;BR IF NO
010114                          4$:
010114  005737  000042                  TST     @#42            ;RUNNING IN AUTO MODE?
```



```
010120  001045                          BNE     7$              ;BR IF YES
010122  104402  005574                  TYPE    ,MCRLF
010126  104403                          INSTR                   ;GET TEST NO.
010130  006032                          MTSTN
010132  104405                          PARAM
010134  000001                          1
010136  001000                          1000
010140  001226                          TSTNO
010142     000                  ,BYTE   0
010143     001                  ,BYTE   1
010144  012700  015766                  MOV     #TST1,R0
010150  022710                  5$:     CMP     (PC)+,(R0)      ;CMP FIRST WORD TO 12737
010152  012737                          MOV     (PC)+,@(PC)+
010154  001020                          BNE     6$              ;BR IF NOT SAME
010156  023760  001226  000002          CMP     TSTNO,2(R0)     ;DOES TSTNO MATCH?
010164  001014                          BNE     6$              ;BR IF NO
010166  022760  001226  000004          CMP     #TSTNO,4(R0)    ;IS LAST WORD OK?
010174  001010                          BNE     6$              ;BR IF NO
010176  010037  001214                  MOV     R0,RETURN       ;IT IS A LEGAL TEST SO DO IT
010202  104402  005657                  TYPE    ,MR
010206  042737  000002  001236          BIC     #SW01,STRTSW
010214  000412                          BR      8$
010216  005720                  6$:     TST     (R0)+           ;POP R0
010220  020027  031442                  CMP     R0,#TLAST+10    ;AT END YET?
010224  001351                          BNE     5$              ;BR IF NO
010226  104402  005570                  TYPE    ,MQM            ;YES ILLEGAL TEST NO.
010232  000730                          BR      4$              ;TRY AGAIN

010234  012737  015766  001214  7$:     MOV     #TST1,RETURN    ;PREPARE RETURN ADDRESS
010242  013701  001404  8$:     MOV     DMCSR,R1        ;R1 = BASE DMC11 ADDRESS
010246  000177  170742                  JMP     @RETURN         ;GO START TESTING.


                                                ;ROUTINE USED TO "AUTO SIZE" THE DMC11
                                                ;CSR AND VECTOR.
                                                ;NOTE:  THE CSR MAY BE ANY WHERE IN THE FLOATING
                                                ;       ADDRESS RANGE (160000:164000)
                                                ;       AND THE VECTOR MAY BE ANY WHERE IN THE
                                                ;       FLOATING VECTOR RANGE (300:770)
                                                ;

010252                          AUTO.SIZE:
010252  000005                          RESET                   ;INSURE A BUS INIT.
010254  012702  001500          CSRMAP: MOV     #DM.MAP,R2      ;LOAD MAP POINTER.
010260  005022                  1$:     CLR     (R2)+           ;ZERO ENTIRE MAP
010262  022702  001700                  CMP     #DM.END,R2      ;ALL DONE?
010266  001374                          BNE     1$              ;BR IF NO
010270  005037  001310                  CLR     DMNUM           ;SET OCTAL NUMBER OF DMC11'S TO 0
010274  012702  001500                  MOV     #DM.MAP,R2      ;R2 POINTS TO DMC MAP
010300  005037  001306                  CLR     DMACTV          ;CLEAR ACTIVE
010304  032737  000001  001236          BIT     #SW00,STRTSW    ;QUESTIONS?
010312  001002                          BNE     .+6             ;BR IF YES
010314  000137  010744                  JMP     7$              ;IF NO SKIP QUESTIONS
010320  012737  000001  001256          MOV     #1,TEMP5        ;START WITH 1
010326  104403                          INSTR
010330  006352                          NUM
```

```
1358  010332  104405                          PARAM
1359  010334  000001                          1
1360  010336  000020                          16.
1361  010340  001252                          TEMP3
1362  010342     000                          .BYTE   0
1363  010343     001                          .BYTE   1
1364  010344  013737  001252  001310          MOV     TEMP3,DMNUM     ;DMNUM = HOW MANY
1365  010352  104402  005574          12$:    TYPE    ,MCRLF
1366  010356  104410                          CONVRT                  ;TYPE WHICH DMC IS BEING DONE
1367  010360  011450                          WHICH                   ;TEMP5 IS WHICH DMC
1368  010362  005237  001256                  INC     TEMP5
1369  010366  104403                          INSTR
1370  010370  006412                          CSR
1371  010372  104405                          PARAM
1372  010374  160000                          160000
1373  010376  164000                          164000
1374  010400  001254                          TEMP4
1375  010402     000                          .BYTE   0
1376  010403     001                          .BYTE   1
1377  010404  013722  001254                  MOV     TEMP4,(R2)+     ;STORE CSR IN MAP
1378  010410  104403                          INSTR
1379  010412  006430                          VEC
1380  010414  104405                          PARAM
1381  010416  000000                          0
1382  010420  000776                          776
1383  010422  001254                          TEMP4
1384  010424     000                          .BYTE   0
1385  010425     001                          .BYTE   1
1386  010426  013712  001254                  MOV     TEMP4,(R2)      ;STORE VECTOR IN MAP
1387  010432  104402                  10$:    TYPE
1388  010434  006451                          PRIO                    ;ASK WHAT BR LEVEL
1389  010436  004737  011734                  JSR     PC,INTTY        ;GET RESPONSE
1390  010442  022703  000024                  CMP     #24,R3          ;
1391  010446  101014                          BHI     50$             ;BR IF LESS THAN 4
1392  010450  022703  000027                  CMP     #27,R3          ;
1393  010454  103411                          BLO     50$             ;BR IF GREATER THAN 7
1394  010456  012704  000011                  MOV     #11,R4          ;R4 = NUMBER OF SHIFTS
1395  010462  006303                          ASL     R3              ;SHIFT R3 LEFT
1396  010464  005304                          DEC     R4              ;DEC SHIFT COUNT
1397  010466  001375                          BNE     .-4             ;BR IF NOT DONE
1398  010470  042703  170777                  BIC     #170777,R3      ;BIC UNWANTED BITS
1399  010474  050312                          BIS     R3,(R2)         ;PUT BR LEVEL IN STATUS MAP
1400  010476  000403                          BR      8$              ;CONTINUE
1401  010500  104402                  50$:    TYPE
1402  010502  005570                          MQM                     ;RESPONSE IS OUT OF LIMITS
1403  010504  000752                          BR      10$             ;TRY AGAIN
1404  010506  104402                  8$:     TYPE
1405  010510  006510                          CRAM                    ;DOES DMC HAVE CRAM?
1406  010512  004737  011734                  JSR     PC,INTTY        ;GET REPLY
1407  010516  022703  000131                  CMP     #131,R3
1408  010522  001406                          BEQ     9$              ;YES
1409  010524  022703  000116                  CMP     #116,R3         ;NO
1410  010530  001405                          BEQ     16$             ;NOT A Y OR N
1411  010532  104402                          TYPE
1412  010534  005570                          MQM                     ;TYPE "?"
1413  010536  000763                          BR      8$              ;ASK AGAIN
```



```
1414  010540  052712  100000          9$:     BIS     #BIT15,(R2)     ;SET BIT 15 IF CRAM
1415  010544  104402                  16$:    TYPE
1416  010546  006606                          MODU                    ;ASK WHICH LINE UNIT
1417  010550  004737  011734                  JSR     PC,INTTY        ;GET REPLY
1418  010554  022703  000021                  CMP     #21,R3          ;"1"
1419  010560  001417                          BEQ     30$
1420  010562  022703  000022                  CMP     #22,R3          ;"2"
1421  010566  001412                          BEQ     31$
1422  010570  022703  000116                  CMP     #116,R3         ;"N"
1423  010574  001403                          BEQ     32$
1424  010576  104402                          TYPE
1425  010600  005570                          MQM                     ;IF NOT A 1,2 OR N TYPE "?"
1426  010602  000760                          BR      16$             ;TRY AGIAN
1427  010604  052722  010000          32$:    BIS     #BIT12,(R2)+    ;SET BIT 12 IN STAT2 IF NO LU
1428  010610  022222                          CMP     (R2)+,(R2)+     ;POP OVER STAT2 AND STAT3
1429  010612  000447                          BR      33$
1430  010614  052712  020000          31$:    BIS     #BIT13,(R2)     ;SET BIT 13 IN STAT2 IF M8202
1431  010620  104402                  30$:    TYPE
1432  010622  007016                          CONN                    ;ASK IF LOOP-BACK IS ON
1433  010624  004737  011734                  JSR     PC,INTTY        ;GET REPLY
1434  010630  022703  000131                  CMP     #131,R3         ;Y
1435  010634  001406                          BEQ     17$
1436  010636  022703  000116                  CMP     #116,R3         ;N
1437  010642  001406                          BEQ     18$
1438  010644  104402                          TYPE
1439  010646  005570                          MQM                     ;IF NOT Y OR N TYPE "?"
1440  010650  000763                          BR      30$             ;TRY AGAIN
1441  010652  052722  040000          17$:    BIS     #BIT14,(R2)+    ;TURNAROUND IS CONNECTED
1442  010656  000402                          BR      19$
1443  010660  042722  040000          18$:    BIC     #BIT14,(R2)+    ;NO TURNAROUND
1444  010664                          19$:
1445  010664  104403                          INSTR
1446  010666  006720                          LINE
1447  010670  104405                          PARAM
1448  010672  000000                          0
1449  010674  000377                          377
1450  010676  001254                          TEMP4
1451  010700     000                          .BYTE   0
1452  010701     001                          .BYTE   1
1453  010702  113722  001254                  MOVB    TEMP4,(R2)+     ;STORE SWITCH PAC IN MAP
1454  010706  104403                          INSTR
1455  010710  006756                          BM
1456  010712  104405                          PARAM
1457  010714  000000                          0
1458  010716  000377                          377
1459  010720  001254                          TEMP4
1460  010722     000                          .BYTE   0
1461  010723     001                          .BYTE   1
1462  010724  113722  001254                  MOVB    TEMP4,(R2)+     ;STORE SWITCH PAC IN MAP
1463  010730  005722                          TST     (R2)+           ;POP OVER STAT3
1464  010732  005337  001252          33$:    DEC     TEMP3           ;DEC DMC COUNT
1465  010736  001205                          BNE     12$             ;BR IF MORE TO DO
1466  010740  000137  011350                  JMP     13$             ;CONTINUE
1467  010744  012701  160000          7$:     MOV     #160000,R1      ;SET FOR FIRST ADDRESS TO BE TESTED
1468  010750  012737  011442  000004          MOV     #6$,@#4         ;SET FOR NON-EXISTANT DEVICE TIME OUT
1469  010756  005011                  2$:     CLR     (R1)            ;CLEAR SEL0
```

```
1470  010760  005711                          TST     (R1)            ;IF DMC11 DMCSR S/B 0
1471  010762  001162                          BNE     3$              ;IF NO DEV ; TRAP TO 4, IF NO BIT 8 THEN NO DMC1
1472  010764  005061  000006                  CLR     6(R1)           ;CLEAR SEL6
1473  010770  005761  000006                  TST     6(R1)           ;IF DMC11 THEN DMRIC S/B =01
1474  010774  001155                          BNE     3$              ;BR IF NOT DMC11
1475  010776  012711  002000                  MOV     #BIT10,(R1)     ;SET ROMO
1476  011002  005061  000004                  CLR     4(R1)           ;CLEAR SEL4
1477  011006  012761  125252  000006          MOV     #125252,6(R1)   ;WRITE THIS TO SEL6
1478  011014  052711  020000                  BIS     #BIT13,(R1)     ;WRITE IT1
1479  011020  022761  125252  000004          CMP     #125252,4(R1)   ;WAS IT WRITTEN?
1480  011026  001004                          BNE     21$             ;IF NO IT IS NOT CRAM
1481  011030  052762  100000  000002          BIS     #BIT15,2(R2)    ;SET BIT15 IF CRAM
1482  011036  000421                          BR      22$
1483  011040  012711  001000          21$:    MOV     #BIT9,(R1)      ;SET ROM1
1484  011044  012761  100400  000006          MOV     #100400,6(R1)   ;PUT INSTRUCTION IN SEL6
1485  011052  012711  001400                  MOV     #BIT9!BIT8,(R1) ;CLOCK INSTRUCTION (MICRO PROC PC TO 0)
1486  011056  012711  002000                  MOV     #BIT10,(R1)     ;SET ROMO
1487  011062  022761  063220  000006          CMP     #63220,6(R1)    ;IS IT CROM
1488  011070  001404                          BEQ     22$             ;BR IF YES
1489  011072  022761  177777  000006          CMP     #-1,6(R1)       ;IF = -1 IT HAS NO CROM
1490  011100  001113                          BNE     3$              ;BR IF NOT DMC11
1491                                  ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DMC11 CSR ADDRESS.
1492  011102  010122                  22$:    MOV     R1,(R2)+        ;STORE CSR IN CORE TABLE.
1493  011104  012711  001000          15$:    MOV     #BIT9,(R1)      ;CLEAR LINE UNIT LOOP
1494  011110  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
1495  011114  012761  122113  000006          MOV     #122113,6(R1)   ;LOAD INSTRUCTION (CLR DTR)
1496  011122  052711  000400                  BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
1497  011126  012761  021264  000006          MOV     #021264,6(R1)   ;LOAD INSTRUCTION
1498  011134  052711  000400                  BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
1499  011140  122761  000377  000004          CMPB    #377,4(R1)      ;IS IT ALL ONES?
1500  011146  001003                          BNE     .+10            ;BR IF NO
1501  011150  052712  010000                  BIS     #BIT12,(R2)     ;IF YES, NO LINE UNIT, SET STATUS BIT
1502  011154  000436                          BR      20$
1503  011156  032761  000002  000004          BIT     #BIT1,4(R1)     ;IS SWITCH A ONE?
1504  011164  001403                          BEQ     .+10            ;BR IF M8201
1505  011166  052712  060000                  BIS     #BIT13!BIT14,(R2) ;M8202 ASSUME CONNECTOR
1506  011172  000427                          BR      20$             ;CONNECTOR ON)
1507  011174  032761  000010  000004          BIT     #BIT3,4(R1)     ;IS MRDY SET
1508  011202  001023                          BNE     20$             ;BR IF M8201 NO CONNECTOR (ON LINE)
1509  011204  012761  000100  000004          MOV     #BIT6,4(R1)     ;LOAD PORT4
1510  011212  012761  122113  000006          MOV     #122113,6(R1)   ;LOAD INSTRUCTION
1511  011220  052711  000400                  BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(SET DTR)
1512  011224  012761  021264  000006          MOV     #021264,6(R1)   ;LOAD INSTRUCTION
1513  011232  052711  000400                  BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(READ MODEM REG)
1514  011236  032761  000010  000004          BIT     #BIT3,4(R1)     ;IS MRDY SET NOW?
1515  011244  001402                          BEQ     20$             ;BR IF NO CONNECTOR
1516  011246  052712  040000                  BIS     #BIT14,(R2)     ;SET STATUS BIT FOR CONNECTOR
1517  011252  005722                  20$:    TST     (R2)+           ;POP POINTER
1518  011254  012761  021324  000006          MOV     #021324,6(R1)   ;PUT INSTRUCTION IN PORT6
1519  011262  012711  001400                  MOV     #BIT9!BIT8,(R1) ;PORT4_LU 15
1520  011266  156122  000004                  BISB    4(R1),(R2)+     ;STORE DDCMP LINE # IN TABLE
1521  011272  012761  021344  000006          MOV     #021344,6(R1)   ;PORT6_INSTRUCTION
1522  011300  012711  001400                  MOV     #BIT8!BIT9,(R1) ;CLOCK INSTR.
1523  011304  156122  000004                  BISB    4(R1),(R2)+     ;STORE BM873 ADD IN TABLE
1524  011310  005722                          TST     (R2)+           ;POP OVER STAT3
1525  011312  005011                          CLR     (R1)            ;CLEAR ROM1
```



```
1526  011314  005237  001310                  INC     DMNUM           ;UPDATE DEVICE COUNTER
1527  011320  022737  000020  001310          CMP     #20,DMNUM       ;ARE MAX. NO. OF DEV FOUND?
1528  011326  001410                          BEQ     13$             ;YES DON'T LOOK FOR ANY MORE.
1529  011330  005011                  3$:     CLR     (R1)            ;CLEAR BIT 10
1530  011332  005061  000006                  CLR     6(R1)           ;CLEAR SEL 6
1531  011336  062701  000010          14$:    ADD     #10,R1          ;UPDATE CSR POINTER ADDRESS
1532  011342  022701  164000                  CMP     #164000,R1
1533  011346  001203                          BNE     2$              ;BR IF MORE ADDRESS TO CHECK.
1534  011350  005037  001306          13$:    CLR     DMACTV
1535  011354  005737  001310                  TST     DMNUM           ;WERE ANY DMC11'S FOUND AT ALL?
1536  011360  001423                          BEQ     5$              ;ERROR AUTO SIZER FOUND NO DMC11'S IN THIS SYS.
1537  011362  013701  001310                  MOV     DMNUM,R1
1538  011366  010137  001314                  MOV     R1,SAVNUM       ;SAVE NUMBER OF DEVICES
1539  011372  000241                  4$:     CLC
1540  011374  006137  001306                  ROL     DMACTV          ;GENERATE ACTIVE REGISTER OF DEVICES.
1541  011400  005237  001306                  INC     DMACTV          ;SET THE BIT
1542  011404  005301                          DEC     R1
1543  011406  001371                          BNE     4$              ;BR IF MORE TO GENERATE
1544  011410  012737  000006  000004          MOV     #6,@#4          ;RESTORE TRAP VECTOR
1545  011416  013737  001306  001312          MOV     DMACTV,SAVACT   ;SAVE ACTIVE REGISTER
1546  011424  000137  011456                  JMP     VECMAP          ;GO FIND THE VECTOR NOW.
1547  011430  104402  005662          5$:     TYPE    ,MERR2          ;NOTIFY OPR THAT NO DMC11'S FOUND.
1548  011434  005000                          CLR     R0              ;MAKE DATA LIGHTS ZERO
1549  011436  000000                          HALT                    ;STOP THE SHOW
1550  011440  000776                          BR      .-2             ;DISABLE CONT, SW.
1551  011442  012716  011336          6$:     MOV     #14$,(SP)       ;ENTERED BY NON-EXISTANT TIME-OUT.
1552  011446  000002                          RTI                     ;RETURN TO MAINSTREAM
1553
1554  011450  000001                  WHICH:  1
1555  011452      002      002                .BYTE   2,2
1556  011454  001256                          TEMP5
1557
1558  011456  032737  000001  001236  VECMAP: BIT     #SW00,STRTSW
1559  011464  001114                          BNE     5$
1560  011466  012737  000340  000022          MOV     #340,@#22       ;SET IOT TRAP PRIO TO 7
1561  011474  012737  011650  000020          MOV     #4$,@#20        ;SET IOT TRAP VECTOR
1562  011502  012702  001500                  MOV     #DM.MAP,R2      ;SET SOFTWARE POINTER
1563  011506  012700  000300                  MOV     #300,R0         ;FLOATING VECTORS START HERE.
1564  011512  012701  000302                  MOV     #302,R1         ;PC OF IOT INSTR.
1565  011516  010120                  1$:     MOV     R1,(R0)+        ;START FILLING VECTOR AREA
1566  011520  012721  000004                  MOV     #4,(R1)+        ;WITH .+2; IOT
1567  011524  022021                          CMP     (R0)+,(R1)+     ;ADD 2 TO R0 +R1
1568  011526  020127  001000                  CMP     R1,#1000
1569  011532  101771                          BLOS    1$              ;BR IF MORE TO FILL
1570  011534  013737  001306  001246          MOV     DMACTV,TEMP1    ;STORE TEMPORALLY
1571  011542  006037  001246          2$:     ROR     TEMP1           ;BRING OUT A BIT
1572  011546  103063                          BCC     5$              ;BR IF ALL DONE
1573  011550  012704  000012                  MOV     #12,P4          ;R4 IS INDEX REGISTER
1574  011554  016437  011720  177776          MOV     BRLVL(R4),PS    ;SET PS TO 7
1575  011562  011201                          MOV     (R2),R1
1576  011564  012761  000200  000004          MOV     #200,4(R1)
1577  011572  012711  001000                  MOV     #BIT9,(R1)      ;SET ROM1
1578  011576  012761  121111  000006          MOV     #121111,6(R1)   ;PUT INSTRUCTION IN PORT6
1579  011604  012711  001400                  MOV     #BIT9!BIT8,(R1) ;FORCE AN INTERRUPT
1580  011610  105200                  7$:     INCB    R0              ;STALL
1581  011612  001376                          BNE     .-2             ;FOR TIME TO INTERRUPT
```

```
1582  011614  162704  000002                        SUB     #2,R4           ;GET NEXT LOWEST PS LEVEL
1583  011620  001404                                BEQ     6$              ;BR IF R4 = 0
1584  011622  016437  011720  177776                MOV     BRLVL(R4),PS    ;MOVE NEXT LOWER LEVEL IN PS
1585  011630  000767                                BR      7$              ;BR TO DELAY
1586  011632  052762  005300  000002          6$:   BIS     #5300,2(R2)     ;NO INTERUPT ASSUME 300 AT LEVEL 5 AND FIX DMC11
1587  011640  005011                          3$:   CLR     (R1)            ;CLEAR ROMI
1588  011642  062702  000010                        ADD     #10,R2          ;POP SOFTWARE POINTER
1589  011646  000735                                BR      2$              ;KEEP GOING
1590  011650  051662  000002                  4$:   BIS     (SP),2(R2)      ;GET VECTOR ADDRESS
1591  011654  042762  000007  000002                BIC     #7,2(R2)        ;CLEAR JUNK
1592  011662  016405  011722                        MOV     BRLVL+2(R4),R5  ;GET BR LEVEL OF DMC11
1593  011666  006305                                ASL     R5              ;SHIFT LEVEL 4 PLACES
1594  011670  006305                                ASL     R5              ;TO THE LEFT FOR THE
1595  011672  006305                                ASL     R5              ;STATUS TABLE
1596  011674  006305                                ASL     R5
1597  011676  042705  170777                        BIC     #170777,R5      ;CLEAR UNWANTED BITS
1598  011702  050562  000002                        BIS     R5,2(R2)        ;PUT BR LEVEL IN STATUS TABLE
1599  011706  022626                                CMP     (SP)+,(SP)+     ;POP IOT JUNK OFF STACK
1600  011710  012716  011640                        MOV     #3$,(SP)        ;SET FOR RETURN
1601  011714  000002                                RTI
1602  011716  000207                          5$:   RTS     PC              ;ALL DONE WITH "AUTO SIZING"
1603
1604  011720  000000                          BRLVL:  0       ;LEVEL 0
1605  011722  000000                                  0       ;LEVEL 0
1606  011724  000200                                  200     ;LEVEL 4
1607  011726  000240                                  240     ;LEVEL 5
1608  011730  000300                                  300     ;LEVEL 6
1609  011732  000340                                  340     ;LEVEL 7
1610
1611
1612  011734  105777  167244                  INTTY:  TSTB    @TKCSR          ;WAIT FOR DONE
1613  011740  100375                                  BPL     .-4
1614  011742  017703  167240                        MOV     @TKDBR,R3       ;PUT CHAR IN R3
1615  011746  105777  167236                        TSTB    @TPCSR          ;WAIT UNTIL PRINTER IS READY
1616  011752  100375                                BPL     .-4
1617  011754  010377  167232                        MOV     R3,@TPDBR       ;ECHO CHAR
1618  011760  042703  000240                        BIC     #BIT7!BIT5,R3   ;MASK OFF LOWER CASE
1619  011764  000207                                RTS     PC              ;RETURN
1620
1621                                  15300
1622  011766                          15400   ROMMAP:
```

```
                        .TITLE  DMC-11 MICROPROCESSOR INSTRUCTIONS
                        .SBTTL  MACRO DEFINITIONS
                        ;
                        .SBTTL REVISION 00
                        .SBTTL FEBRUARY 25, 1975
                        .SBTTL
                        .SBTTL REVISION 01
                        .SBTTL MARCH 18,1975
                        .SBTTL NEW CSR BOARD CHANGES
                        .SBTTL
                        .SBTTL HARVEY M. SCHLESINGER
                        ;
                        .SBTTL COPYRIGHT 1975 DIGITAL EQUIPMENT CORPORATION
                        ;
```

```
                        NEW=0
                        ;MICROPROCESSOR INSTRUCTION WORD DEFINITIONS
000000                  MOVE=0                  ;OPCODE MOVE
100000                  JUMP=100000             ;OPCODE JUMP
020000                  IBUS=20000              ;SOURCE IBUS
000000                  IMM=0                   ;SOURCVE IMMEDIATE
040000                  MEMX=40000              ;SOURCE MEMORY
060000                  BRX=60000               ;SOURCE BR
060000                  BR=60000                ;SOURCE BR

060000                  DP=60000                ;SOURCE BR
010000                  LDMAR=10000             ;MA=LOAD MAR LO
014000                  INCMAR=14000            ;MA=INCREMENT MAR
000400                  WRTEBR=400              ;DEST=WRITE BR
001000                  WROUTX=1000             ;DEST=EXTENDED IBUS
001400                  SHFTBR=1400             ;DEST=SHIFT BR LEFT
002000                  WROUT=2000              ;DEST=WRITE OUTPUT
002400                  WRMEM=2400              ;DEST=WRITE MEMORY
003000                  SPX=3000                ;DEST=WRITE SP
003400                  SPBRX=3400              ;DEST=WRITE SP AND BR
                        ;FUNCTIONS
000200                  SELA=200                ;FUNCTION=SELECT A
000220                  SELB=220                ;FUNCTION=SELECT B
000240                  AORNB=240               ;FUNCTION=A OR NOT B
000260                  AANDB=260               ;FUNCTION A  AND B
000300                  AORB=300                ;FUNCTION=A OR B
000320                  AXORB=320               ;FUNCTION A XOR B
000340                  SUB=340                 ;SUBTRACT
000360                     SUBTC=360            ;FUNCTION= TWOS COMPLEMENT SUBTRACT
000000                  ADD=0                   ;ADD A+B
000020                  ADDC=20                 ;A+B+CARRY
000040                  SUBC=40                 ;A-B-C
000060                  INCA=60                 ;INCREMENT A
000100                  AC=100                  ;A PLUS CARRY
000120                  AA=120                  ;A PLUS A
000140                  AAC=140                 ;A PLUS A PLUS C
000160                  DECA=160                ;DECREMENT A
                        ;END FUNCTIONS
                        PAGE1=4000
                        PAGE2=10000
                        PAGE3=14000
001000                  CCOND=1000              ;CONDITION C
001400                  ZCOND=1400              ;CONDITION Z
000400                  ALCOND=400              ;ALWAYS
002000                  BR0CON=2000             ;CONDITION BR0
002400                  BR1CON=2400             ;CONDITION BR1
003000                  BR4CON=3000             ;CONDITION BR4
003400                  BR7CON=3400             ;CONDITION BR7
```



```
                        .SBTTL  MICRO INSTRUCTION DEFINITIONS
                        .SBTTL          BRANCH INSTRUCTIONS
                ;
100000                  JUMP=100000             ;JUMP OP CODE
                ;
                        .MACRO  $ZERO
                        MICPC=MICPC+1
                        000000

                        .ENDM   $ZERO
                ;
                        .MACRO ALWAYS ADDRES    ;JUMP ALWAYS
                        MICPC=MICPC+1
                        <JUMP!ALCOND!<ADDRES-INIT&3000*4>!<ADDRES-INIT&777/2>>

                        .ENDM
                ;
                        .MACRO  BR0 ADDRES      ;JUMP IF BR0 SET
                        MICPC=MICPC+1
                        <JUMP!BR0CON!<ADDRES-INIT&3000*4>!<ADDRES-INIT&777/2>>

                        .ENDM
                ;
                        .MACRO  BR1 ADDRES      ;JUMP IF BR1 SET
                        MICPC=MICPC+1
                        <JUMP!BR1CON!<ADDRES-INIT&3000*4>!<ADDRES-INIT&777/2>>

                        .ENDM
                ;
                        .MACRO  BR4 ADDRES      ;JUMP IF BR4 SET
                        MICPC=MICPC+1
                        <JUMP!BR4CON!<ADDRES-INIT&3000*4>!<ADDRES-INIT&777/2>>

                        .ENDM
                ;
                        .MACRO  BR7 ADDRES      ;JUMP IF BR7 SET
                        MICPC=MICPC+1
                        <JUMP!BR7CON!<ADDRES-INIT&3000*4>!<ADDRES-INIT&777/2>>

                        .ENDM
                ;
                        .MACRO  Z ADDRES        ;JUMP IF Z BIT SET
                        MICPC=MICPC+1
                        <JUMP!ZCOND!<ADDRES-INIT&3000*4>!<ADDRES-INIT&777/2>>

                        .ENDM
                ;
                        .MACRO  C ADDRES        ;JUMP IF C BIT SET
                        MICPC=MICPC+1
                        <JUMP!CCOND!<ADDRES-INIT&3000*4>!<ADDRES-INIT&777/2>>

                        .ENDM
                        .SBTTL          INDEXED BRANCH INSTRUCTIONS
                ;
                        .MACRO  .ALWAY SPC,FUNC,SPLOC ;INDEXED JUMP ALWAYS
                        MICPC=MICPC+1
```

```
                        <JUMP!ALCOND!SRC!FUNC!SPLOC>

                        .ENDM
                ;
                        .MACRO  .BR0 SRC,FUNC,SPLOC ;INDEXED JUMP ON BR0 SET
                        MICPC=MICPC+1
                        <JUMP!BR0CON!SRC!FUNC!SPLOC>

                        .ENDM
                ;
                        .MACRO  .BR1 SRC,FUNC,SPLOC ;INDEXED JUMP ON BR1 SET
                        MICPC=MICPC+1
                        <JUMP!BR1CON!SRC!FUNC!SPLOC>

                        .ENDM
                ;
                        .MACRO  .BR4 SRC,FUNC,SPLOC ;INDEXED JUMP ON BR4 SET
                        MICPC=MICPC+1
                        <JUMP! BR4CON!SRC!FUNC!SPLOC>

                        .ENDM
                ;
                        .MACRO  .BR7 SRC,FUNC,SPLOC ;INDEXED JUMP ON BR7 SET
                        MICPC=MICPC+1
                        <JUMP!BR7CON!SRC!FUNC!SPLOC>

                        .ENDM
                ;
                        .MACRO  .Z SRC,FUNC,SPLOC ;INDEXED JUMP ON Z BIT SET
                        MICPC=MICPC+1
                        <JUMP!ZCOND!SRC!FUNC!SPLOC>

                        .ENDM
                ;
                        .MACRO  .C SRC,FUNC,SPLOC ;INDEXED JUMP ON C BIT SET
                        MICPC=MICPC+1
                        <JUMP!CCOND!SRC!FUNC!SPLOC>

                        .ENDM
                        .SBTTL          MOVE INSTRUCTIONS
                ;
000000          MOVE=0          ;MOVE OPCODE
                ;
                        .MACRO  BRSHFT  ;BR SHIFT RIGHT
                        MICPC=MICPC+1
                        <MOVE!SHFTBR!WRTEBR!SELB>

                        .ENDM
                ;
                        .MACRO  BSHFTB  ;BR ROTATE
                        MICPC=MICPC+1
                        <MOVE!SHFTBR!SELB!BR>

                        .ENDM
                ;
                        .MACRO  SP SRC,FUNC,SPLOC ;LOAD SCRATCH-PAD
```



```
                        MICPC=MICPC+1
                        <MOVE!SPX!SRC!FUNC!SPLOC>

                        .ENDM
                ;
                        .MACRO  SPBR SRC,FUNC,SPLOC ;LOAD SP AND BR
                        MICPC=MICPC+1
                        <MOVE!SPBRX!SRC!FUNC!SPLOC>

                        .ENDM
                ;
                        .MACRO  MEM SRC,DATA ;MOVE TO MEMORY
                        MICPC=MICPC+1
                        <MOVE!WRMEM!SRC!<DATA>>

                        .ENDM
                ;
                        .MACRO  MEMADR  ADDRES,FUNC     ;WRITE ADDRESS TO MEMORY
                        MICPC=MICPC+1
                        .IF B   FUNC
                        <MOVE!WRMEM!<ADDRES-INIT&777/2>>
                        .IFF
                        <MOVE!WRMEM!FUNC!<ADDRES-INIT&777/2>>
                        .ENDC
                        .ENDM
                ;
                        .MACRO MEMINC SRC,DATA ;MOVE TO MEM, INCR MAR
                        MICPC=MICPC+1
                        <MOVE!WRMEM!INCMAR!SRC!<DATA>>

                        .ENDM
                ;
                        .MACRO BRWRTE SRC,DATA ;MOVE TO BR
                        MICPC=MICPC+1
                        <MOVE!WRTEBR!SRC!<DATA>>

                        .ENDM
                ;
                        .MACRO  BRADDR ADDRES   ;PUT RETURN ADDR (1 BYTE) IN BR
                        MICPC=MICPC+1
                        <MOVE!WPTEBR!<ADDRES-INIT&777/2>>

                        .ENDM
                ;
                        .MACRO OUTPUT SRC,DATA  ;WRITE OUTPUT
                        MICPC=MICPC+1
                        <MOVE!WROUT!SRC!<DATA>>

                        .ENDM
                ;
                        .MACRO OUT SRC,DATA
                        MICPC=MICPC+1
                        <MOVE!WROUTX!SRC!<DATA>>

                        .ENDM
                ;
```

```
                        ;MACRO LDMA SRC,DATA    ;LOAD MEMORY ADDRESS REG
                        MICPC=MICPC+1
                        .IF IDN SRC,IMM
                        <MOVE!LDMAR!IMM!<DATA&377>>
                        .IFF
                        <MOVE!LDMAR!SRC!<DATA>>
                        .ENDC

                        .ENDM
                ;
                        ;MACRO LDMAP SRC,DATA   ;LOAD MEMORY PAGE NUMBER
                        MICPC=MICPC+1
                        .IF IDN SRC,IMM
                        <MOVE!LDMAPG!IMM!<DATA/400>>
                        .IFF
                        <MOVE!LDMAPG!SRC!<DATA>>
                        .ENDC

                        .ENDM
                ;
                        ;MACRO  LDADDR  DATA    ;LOAD A LINE TABLE ADDRESS
                        BRWRTE  IMM,DATA
                        LDMA    BR,<ADD!SP,RM0>
                        .ENDM
                ;
                        ;MACRO CMP SRC,SPADDR   ;COMPARE SOURCE AND SP
                        MICPC=MICPC+1
                        <SUBTC!SRC!SPADDR>

                        .ENDM
                ;
                        ;MACRO  NOP SRC,FUNC,SPADDR ;NOP=SOURCE, FUNC, NO DEST
                        MICPC=MICPC+1
                        <SRC!FUNC!SPADDR>

                        .ENDM

                        ;MACRO  CALL    REG,ADDRES      ;SUBROUTINE CALL
                        DISP=<MICPC+1>&377
                        BRWRTE  IMM,DISP+3
                        SP      BR,SELB,REG
                        ALWAYS  ADDRES
                        .ENDM

                        ;MACRO  RETURN  REG,PAGE        ;SUBROUTINE RETURN
                        .ALWAY  BR,SELA,<REG!PAGE>
                        .ENDM
```



```
                        .SBTTL  INPUT/OUTPUT ASSIGNMENTS
                        ;IBUS   ASSIGNMENTS
100000                  INCON=0+100000                  ;IN CONTROL CSR
100020                  MAIN=20+100000                  ;MAINTAINENCE REGISTER
100040                  OCON=40+100000                  ;OUT CONTROL CSR
100060                  UBADDR=60+100000                ;UNUSED
100100                  PORT1=100+100000                ;CSR4
100120                  PORT2=120+100000                ;CSR5
100140                  PORT3=140+100000                ;CSR6
100160                  PORT4=160+100000                ;CSR7
100200                  NPR=200+100000                  ;NPR CONTROL
100220                  UBBR=220+100000                 ;BR(INTERRUPT)CONTROL
000000                  INDAT1=0                        ;INPUT DATA LOW BYTE
000020                  INDAT2=20                       ;INPUT DATA HIGH BYTE
000140                  IOBA1=140                       ;OUTPUT BA LOW BYTE
000160                  IOBA2=160                       ;OUTPUT BA HIGH BYTE
000100                  IIBA1=100                       ;INPUT BA LOW BYTE
000120                  IIBA2=120                       ;INPUT BA HIGH BYTE
000200                  RCVDAT=200                      ;RECEIVE DATA
000220                  TMTCON=220                      ;TMTR CONTROL
000240                  RCVCON=240                      ;RCVR CONTROL
000260                  MODEM=260                       ;MODEM CONTROL
000300                  SYNREG=300                      ;SYN REGISTER
000320                  LNOSW=320                       ;LINE NUMBER SWITCH
000340                  BM873=340                       ;BM873 ADDRESS
000360                  LUMAIN=360                      ;LINE UNIT MAINTAINENCE
                        ;OBUS   ASSIGNMENTS
                        ;EXTENDED OBUS
000000                  OINCON=0                        ;IN CONTROL CSR
000001                  OMAIN=1                         ;MAINT
000002                  OOCON=2                         ;OUT CONTROL CSR
000003                  OUBADD=3                        ;UNUSED
000004                  OPORT1=4                        ;CSR4
000005                  OPORT2=5                        ;CSR5
000006                  OPORT3=6                        ;CSR6
000007                  OPORT4=7                        ;CSR7
000010                  ONPR=10                         ;NPR CONTROL
000011                  OBR=11                          ;BR CONTROL
                        ;UNEXTENDED OBUS
000002                  OUTDA1=2                        ;OUTPUT DATA LOW BYTE
000003                  OUTDA2=3                        ;OUTPUT DATA HIGH BYTE
000006                  OBA1=6                          ;OUTPUT BA LOW BYTE
000007                  OBA2=7                          ;OUTPUT BA HIGH BYTE
000004                  IBA1=4                          ;INPUT BA LOW BYTE
000005                  IBA2=5                          ;INPUT BA HIGH BYTE
000010                  TMTDAT=10                       ;TMTR DATA
000011                  OTMTCO=11                       ;TMTR CONTROL
000012                  OPCVCO=12                       ;RCVR CONTROL
000013                  OMODEM=13                       ;MODEM CONTROL
000014                  SYNC=14                         ;SYN REGISTER
000017                  OLUMAN=17                       ;LINE UNIT MAINT.
```

```
                                ;SBTTL PROTOCOL DEPENDANT MACROS
                                ;MACRO  RSTATE  STATE                   ;UPDATE RECEIVE STATE POINTER
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!IMM!<STATE-INIT&777/2>>
                                MICPC=MICPC+1
                                <MOVE!SPX!BR!SELB!SP1>
                                ;ENDM
                                ;
                                ;MACRO  TSTATE  STATE
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!IMM!<STATE-INIT&777/2>>
                                MICPC=MICPC+1
                                <MOVE!SPX!BR!SELB!SP2>
                                ;ENDM
                                ;
                                ;MACRO  STATE   ADDR
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!IMM!<ADDR-INIT&777/2>>
                                ;ENDM
                                ;
                                ;MACRO  PSTATE  STATE
                                MEM     IMM,<<STATE-INIT&777/2>>
                                ;ENDM
                                ;
                                ;MACRO  PSTATI  STATE
                                MEMINC  IMM,<<STATE-INIT&777/2>>
                                ;ENDM
                                ;
                                ;MACRO  SYNMAC
                                SP      BR,SELB,SP2             ;UPDATE STATE POINTER FROM BR
                                SYNOUT
                                ALWAYS  IDLE
                                ;ENDM
                                ;
                                ;MACRO  SYNOUT
                                LDMA    IMM,UNMSG               ;LOAD PTR TO UNNUMB MESSAGE SKELETON
                                OUTPUT  <MEMX!INCMAR>,<SELB!OTMTCO>     ;SOM TO TMTR CONTROL
                                OUTPUT  <MEMX!INCMAR>,<SELB!TMTDAT>     ;SYNC TO TMTR SILO
                                ;ENDM

        177777                  MICPC=177777    ;INIT MICRO PC
```



```
                                ;SBTTL  DMC11 DDCMP MICRO CODE ASSEMBLED FOR USE  WITH THE M8201 LINE UNIT
        000000                  LOW=0
```

```
                        .TITLE DMC11 DDCMP PROTOCOL IMPLEMENTATION
                        .SBTTL VERSION 00A FEBRUARY 26,1975
                        .SBTTL
                        .SBTTL HARVEY M. SCHLESINGER
                        .SBTTL
                        .SBTTL COPYRIGHT 1975, DIGITAL EQUIPMENT CORPORATION
                        .SBTTL
                        .SBTTL  VERSION 00B     MARCH 17,1975
                        .SBTTL  CSR AND MICROPROCESSOR CHANGES
                        .SBTTL
                        .SBTTL  VERSION 00C     NOVEMBER 6, 1975
                        .SBTTL  RETRANSMISSION CHANGES
                        .SBTTL
                        .SBTTL  VERSION 00D     DECEMBER 3,1975
                        .SBTTL  TRANSMIT DONE CHANGES
                        .SBTTL
                        .SBTTL  THE LATEST MODIFICATIONS WERE ADDED ON:
                        .SBTTL          NOVEMBER 16, 1976
```



```
                        .SBTTL  MICROPROCESSOR MAIN MEMORY ASSIGNMENTS
                        ;ALLOCATION OF MICROPROCESSOR MAIN MEMORY
000000                  NAKSR=0         ;NAKS RECD--DYNAMIC
000001                  NAKST=NAKSR+1   ;NAKS TMTED--DYNAMIC
000002                  REPSR=NAKST+1   ;REPS RECD--DYNAMIC
000003                  REPST=REPSR+1   ;REPS TMTED--DYNAMIC
000006                  NP=REPST+3      ;CONSTANT 0
000007                  NTLR=NP+1       ;NAKS=MSG NO BUFFERS CUMUL.
000010                  NHDR=NTLR+1     ;NAKS=MSG HEADER BAD
000011                  NDATR=NHDR+1    ;NAKS=DATA BAD
000012                  NTLS=NDATR+1    ;NAK SENT --NO BUFFERS
000013                  NHDS=NTLS+1     ;NAK SENT BAD HEADER
000014                  NDATS=NHDS+1    ;NAK SENT BAD DATA
000015                  REPCS=NDATS+1   ;REPS SENT CUMUL
000016                  REPCR=REPCS+1   ;REPS RECD CUMUL
000017                  BASE=REPCR+1            ;CORE TABLE BASE ADDRESS
000022                  SRC=BASE+3              ;START OF INPUT CHAIN--NEXT RECV DONE
000023                  ERC=SRC+1               ;END OF INPUT CHAIN
000024                  RCL1=ERC+1              ;RECEIVE LINK #1
000031                  RCL2=RCL1+5             ;   "       "   #2
000036                  RCL3=RCL2+5             ;   "       "   #3
000043                  RCL4=RCL3+5
000050                  RCL5=RCL4+5
000055                  RCL6=RCL5+5
000062                  RCL7=RCL6+5
000067                  STC=RCL7+5              ;START OF OUTPUT CHAIN---NEXT TMT DONE
000070                  ETC=STC+1               ;END OF TRANSMIT CHAIN
000071                  TML1=ETC+1              ;TRANSMIT LINK #1
000077                  TML2=TML1+6             ;   "       "   #2
000105                  TML3=TML2+6             ;   "       "   #3
000113                  TML4=TML3+6
000121                  TML5=TML4+6
000127                  TML6=TML5+6
000135                  TML7=TML6+6
000143                  TML8=TML7+6
000151                  T=TML8+6                ;TYPE FIELD
000152                  ST=T+1          ;SUBTYPE FIELD
000153                  ISP17=ST+1      ;MSG ACKED IMAGE
000154                  IMG10=ISP17+1   ;IMAGE OF BIT 1 OF SP10
000155                  IMG11=IMG10+1   ;IMAGE OF SP11
000156                  IMG12=IMG11+1   ;IMAGE OF SP12
000157                  IMG14=IMG12+1   ;IMAGE OF SP14
000160                  IMG16=IMG14+1   ;IMAGE OF SP16
000161                  IMG17=IMG16+1   ;IMAGE OF SP17
000162                  TYPTAB=IMG17+1  ;TYPE TABLE---
                                        ;72 TYPE TABLE REP
                                        ;73  "      "   NAK
000164                  TYPSTT=TYPTAB+2 ;74  "      "   START
                                        ;75  "      "   STACK
                                        ;
                                        ;
000167                  BC=TYPSTT+3     ;RECEIVE BYTE COUNT
000171                  ISP11=BC+2      ;SP11 IMAGE
000172                  ISP12=ISP11+1   ;SP12 IMAGE
000173                  INCONS=ISP12+1  ;IN CONTROL CSR IMAGE
000174                  RTHRS=INCONS+1  ;RECV THRESHOLD LINK
```

```
                        ;ALL LOCATIONS FROM 200 ON ARE NOT WRITTEN OUT DURING A TABLE UPDATE

000210                  TABST=210       ;TABLE UPDATE STATE
000211                  PRTST=TABST+1   ;PORT STATE
000240                  NXTINT=240      ;NEXT INTERRUPT POSITION
000241                  NXTSP=NXTINT+1  ;END OF INTERUPT CHAIN
000242                  INTSTK=NXTSP+1  ;STACK OF INTERUPTS
000400                  MMEND=400       ;MAIN MEMORY END
```



```
                        .SBTTL  SCRATCH PAD ASSIGNMENTS
000000                  SP0=0   ;SP0---SCRATCH REGISTER
000001                  SP1=1   ;SP1---PORT STATUS WORD
                                ;BIT ASSIGNMENTS
                                ;BIT0--INIT MODE
                                ;BIT1--SEC STATION SELECT(UNUSED)
                                ;BIT2--NO BUFFER ASSIGNED IN BOOT MODE
                                ;BIT3--DLE RECEIVED WHILE NOT IN MAINT MODE
                                ;BIT4--INTERRUPT PENDING
                                ;BIT6--DISCONNECT ERROR
                                ;BIT7--BOOT MODE
000002                  SP2=2   ;SP2---TRANSMIT STATE POINTER
000003                  SP3=3   ;SP3---RECEIVE STATE POINTER
000004                  SP4=4   ;SP4---END RECV ADDRESS
000005                  SP5=5   ;SP5---END RECEIVE ADDRESS
000006                  SP6=6   ;SP6---END TRANSMIT ADDRESS
000007                  SP7=7   ;SP7---END TRANSMIT ADDRESS
000010                  SP10=10 ;SP10---LINE STATUS WORD
                                ;BIT ASSIGNMENTS
                                ;BIT0--UNNUMB PENDING
                                ;BIT1--MESSAGE IN PROGRESS
                                ;BIT2--LINE HAS GONE IDLE
                                ;BIT3--START RECEIVED
                                ;BIT4--CLEAR ACTIVE ON END
                                ;BIT5--START MODE
                                ;BIT6--HALF DUPLEX
                                ;BIT7--OK TO SEND
000011                  SP11=11 ;SP11---R FIELD
000012                  SP12=12 ;SP12---N FIELD
000013                  SP13=13 ;SP13---TYPE
000014                  SP14=14 ;SP14---RECEIVE LINK IMAGE
000015                  SP15=15 ;SP15---TIMER ENTRY---NUMBER OF ONE SECOND TICKS
000016                  SP16=16 ;SP16---POINTER TO TMT LINK COPY IN MAIN MEM
000017                  SP17=17 ;SP17---LAST MESSAGE ACKNOWLEDGED
```

```
                                        .SBTTL  INIT--INITIALIZATION ROUTINE
                                        ;ZEROS MAIN MEMORY
                                        ;LOOPS WAITING FOR RECEIVE DATA(BOOT?)
                                        ;OR FOR RQI TO BE SET
                                        ;WILL ACCEPT ONLY BASE FORMAT, ALL OTHERS WILL RETURN A PROCEDURE ERROR
                                        ;
                                        ;AT INITIALIZATION --- THE HARDWARE CLEARS THE BR AND MAR
                 011766                 .=11766
011766                          INIT:   SP      BR,SELB,SP0             ;CLEAR   SP0
(1)              000000                 MICPC=MICPC+1
(1)  011766      063720                 <MOVE!SPX!BR!SELB!SP0>
(1)
011770                                  SP      BR,SELB,SP3             ;PAGE ONE TRANSFER ADDRESS
(1)              000001                 MICPC=MICPC+1
(1)  011770      063223                 <MOVE!SPX!BR!SELB!SP3>
(1)
011772                                  SP      BR,SELB,SP17            ;CLEAR SP17
(1)              000002                 MICPC=MICPC+1
(1)  011772      063237                 <MOVE!SPX!BR!SELB!SP17>
(1)
011774                                  OUT     BR,<SELA!OINCON>        ;ZERO THE IN CONTROL CSR
(1)              000003                 MICPC=MICPC+1
(1)  011774      061200                 <MOVE!WROUTX!BR!<SELA!OINCON>>
(1)
011776                                  OUT     BR,<SELA!OOCON>         ;ZERO THE OUT CONTROL CSR
(1)              000004                 MICPC=MICPC+1
(1)  011776      061202                 <MOVE!WROUTX!BR!<SELA!OOCON>>
(1)
012000                                  SP      IMM,370,SP10            ;WRITE 5 ONE BITS TO THE HIGH ORDER
(1)              000005                 MICPC=MICPC+1
(1)  012000      003370                 <MOVE!SPX!IMM!370!SP10>
(1)
                                                                        ;BITS OF SP10
012002                          5$:     SP      BR,AA,SP10              ;SHIFT SP10 LEFT SETTING CARRY THE
(1)              000006                 MICPC=MICPC+1
(1)  012002      063130                 <MOVE!SPX!BR!AA!SP10>
(1)
                                                                        ;FIRST 5 TIMES THRU THE LOOP
012004                                  MEMINC  BR,ADDC!SP3             ;WRITE A ONE TO THE FIRST 5 MEMORY
(1)              000007                 MICPC=MICPC+1
(1)  012004      076423                 <MOVE!WRMEM!INCMAR!BR!<ADDC!SP3>>
(1)
                                                                        ;LOCATIONS AND ZERO THE REST
012006                                  SP      BR,INCA,SP0             ;INCREMENT COUNTER
(1)              000010                 MICPC=MICPC+1
(1)  012006      063060                 <MOVE!SPX!BR!INCA!SP0>
(1)
012010                                  Z       10$                     ;ALL DONE
(1)              000011                 MICPC=MICPC+1
(1)  012010      101413                 <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
(1)
012012                                  ALWAYS  5$                      ;KEEP GOING
(1)              000012                 MICPC=MICPC+1
(1)  012012      100406                 <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
(1)
012014                          10$:    SPBR    IMM,1,SP1               ;WRITE A 1 TO THE BR AND SP1
```



```
(1)              000013                 MICPC=MICPC+1
(1)  012014      003401                 <MOVE!SPBRX!IMM!1!SP1>
(1)
012016                                  SP      BR,SELB,SP11            ;WRITE A 1 TO SP11
(1)              000014                 MICPC=MICPC+1
(1)  012016      063231                 <MOVE!SPX!BR!SELB!SP11>
(1)
012020                                  SP      BR,SELB,SP12            ;WRITE A 1 TO SP12
(1)              000015                 MICPC=MICPC+1
(1)  012020      063232                 <MOVE!SPX!BR!SELB!SP12>
(1)
012022                                  LDMA    IMM,TYPTAB              ;POINT MAR TO TYPE TABLE
(1)              000016                 MICPC=MICPC+1
(1)                 001                 .IF IDN IMM,IMM
(1)  012022      010162                 <MOVE!LDMAR!IMM!<TYPTAB&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<TYPTAB>>
(1)                 000                 .ENDC
(1)
012024                                  BPWRTE  IMM,226                 ;WRITE SYNC TO MEMORY
(1)              000017                 MICPC=MICPC+1
(1)  012024      000626                 <MOVE!WRTEBR!IMM!<226>>
(1)
012026                                  OUTPUT  BR,SELB!SYNC    ;LOAD THE SYNC REGISTER
(1)              000020                 MICPC=MICPC+1
(1)  012026      062234                 <MOVE!WROUT!BR!<SELB!SYNC>>
(1)
012030                                  MEMINC  IMM,3                   ;REP
(1)              000021                 MICPC=MICPC+1
(1)  012030      016403                 <MOVE!WRMEM!INCMAR!IMM!<3>>
(1)
012032                                  MEM     IMM,2                   ;NAK
(1)              000022                 MICPC=MICPC+1
(1)  012032      002402                 <MOVE!WRMEM!IMM!<2>>
(1)
012034                                  SP      MEMX!INCMAR,SELB,SP15           ;SET STARTING COUNT
(1)              000023                 MICPC=MICPC+1
(1)  012034      057235                 <MOVE!SPX!MEMX!INCMAR!SELB!SP15>
(1)
012036                                  MEMINC  IMM,6                   ;START
(1)              000024                 MICPC=MICPC+1
(1)  012036      016406                 <MOVE!WRMEM!INCMAR!IMM!<6>>
(1)
012040                                  MEMINC  IMM,7                   ;STACK
(1)              000025                 MICPC=MICPC+1
(1)  012040      016407                 <MOVE!WRMEM!INCMAR!IMM!<7>>
(1)
012042                                  MEMINC  IMM,1                   ;ACK
(1)              000026                 MICPC=MICPC+1
(1)  012042      016401                 <MOVE!WRMEM!INCMAR!IMM!<1>>
(1)
012044                                  LDMA    IMM,TABST               ;POINT TO TABLE UPDATE STATE
(1)              000027                 MICPC=MICPC+1
(1)                 001                 .IF IDN IMM,IMM
(1)  012044      010210                 <MOVE!LDMAR!IMM!<TABST&377>>
(1)                                     .IFF
```

```
(1)                                     <MOVE!LDMAR!IMM!<TABST>>
(1)             000                     .ENDC
(1)
537  012046                             PSTATI  I3                      ;INITIALIZE IT
(1)  012046                             MEMINC  IMM,<<I3-INIT&777/2>>
(2)          000030                     MICPC=MICPC+1
(2)  012046  016460                     <MOVE!WRMEM!INCMAR!IMM!<<I3-INIT&777/2>>>
(2)
538  012050                             PSTATI  NIDLE2                  ;INITIALIZE PORT STATUS
(1)  012050                             MEMINC  IMM,<<NIDLE2-INIT&777/2>>
(2)          000031                     MICPC=MICPC+1
(2)  012050  016533                     <MOVE!WRMEM!INCMAR!IMM!<<NIDLE2-INIT&777/2>>>
(2)
539  012052                             LDMA    IMM,STC                 ;LOAD ADDRESS OF LAST TMT CHAIN
(1)          000032                     MICPC=MICPC+1
(1)             001                     .IF IDN IMM,IMM
(1)  012052  010067                     <MOVE!LDMAR!IMM!<STC&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<STC>>
(1)             000                     .ENDC
(1)
540  012054                             MEMINC  IMM,TML1                ;STORE ADDRESS OF FIRST TMT LINK
(1)          000033                     MICPC=MICPC+1
(1)  012054  016471                     <MOVE!WRMEM!INCMAR!IMM!<TML1>>
(1)
541  012056                             MEM     IMM,TML1
(1)          000034                     MICPC=MICPC+1
(1)  012056  002471                     <MOVE!WRMEM!IMM!<TML1>>
(1)
542  012060                             SP      MEMX,SELB,SP16          ;INITIALIZE LAST XMIT POINTER
(1)          000035                     MICPC=MICPC+1
(1)  012060  043236                     <MOVE!SPX!MEMX!SELB!SP16>
(1)
543  012062                             LDMA    IMM,SRC                 ;LOAD ADDRESS OF LAST RECV CHAIN
(1)          000036                     MICPC=MICPC+1
(1)             001                     .IF IDN IMM,IMM
(1)  012062  010022                     <MOVE!LDMAR!IMM!<SRC&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<SRC>>
(1)             000                     .ENDC
(1)
544  012064                             MEMINC  IMM,RCL1                ;SET UP ADDRESS OF FIRST RECV LINK
(1)          000037                     MICPC=MICPC+1
(1)  012064  016424                     <MOVE!WRMEM!INCMAR!IMM!<RCL1>>
(1)
545  012066                             MEM     IMM,RCL1
(1)          000040                     MICPC=MICPC+1
(1)  012066  002424                     <MOVE!WRMEM!IMM!<RCL1>>
(1)
546  012070                             SP      MEMX,SELB,SP14
(1)          000041                     MICPC=MICPC+1
(1)  012070  043234                     <MOVE!SPX!MEMX!SELB!SP14>
(1)
547  012072                             LDMA    IMM,NXTINT              ;ADDRESS OF NEXT INTERRUPT POINTER TO MAR
(1)          000042                     MICPC=MICPC+1
(1)             001                     .IF IDN IMM,IMM
```



```
(1)  012072  010240                     <MOVE!LDMAR!IMM!<NXTINT&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<NXTINT>>
(1)             000                     .ENDC
(1)
548  012074                             MEMINC  IMM,INTSTK                      ;INITIALIZE NEXT INTERRUPT POINTER
(1)          000043                     MICPC=MICPC+1
(1)  012074  016642                     <MOVE!WRMEM!INCMAR!IMM!<INTSTK>>
(1)
549  012076                             MEM     IMM,INTSTK                      ;INITIALIZE INSERTION POINTER
(1)          000044                     MICPC=MICPC+1
(1)  012076  002642                     <MOVE!WRMEM!IMM!<INTSTK>>
(1)
550  012100                             BRWRTE  IMM,200                 ;WRITE THE RUN BIT TO THE BR
(1)          000045                     MICPC=MICPC+1
(1)  012100  000600                     <MOVE!WRTEBR!IMM!<200>>
(1)
551  012102                             OUT     BR,<SELB!OMAIN>         ;WRITE  THE RUN BIT TO MAINT CSR
(1)          000046                     MICPC=MICPC+1
(1)  012102  061221                     <MOVE!WROUTX!BR!<SELB!OMAIN>>
(1)
552                                                                     ;FALL INTO IDLE LOOP
553             001                     .IF NDF $LOW
554  012104                             ALWAYS  TEOM2
(1)          000047                     MICPC=MICPC+1
(1)  012104  110740                     <JUMP!ALCOND!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
(1)
555                                     ;
556  012106                         REXIT:  SP      BR,SELB,SP3
(1)          000050                     MICPC=MICPC+1
(1)  012106  063223                     <MOVE!SPX!BR!SELB!SP3>
(1)
557             000                     .ENDC
```

```
                                        .SBTTL  IDLE--PROGRAM IDLE LOOP
                                        ;PROGRAM IDLE LOOP
                                        ;DISPATCHES TO APPROPRIATE SERVICE ROUTINES
                                        ;USES STATE POINTERS FOR TMT,RCV,CSR ACTIVITY
                                        ;
012110                          IDLE:   BRWRTE  BR,<SELA!SP10>          ;READ TRANSMIT STATUS WORD FROM SP10 TO BR
(1)     000051                          MICPC=MICPC+1
(1)  012110  060610                     <MOVE!WRTEBR!BR!<SELA!SP10>>
(1)
012112                                  BR1     TMTDA                   ;IF DATA TO SEND-- BRANCH
(1)     000052                          MICPC=MICPC+1
(1)  012112  112400                     <JUMP!BR1CON!<TMTDA-INIT&3000*4>!<TMTDA-INIT&777/2>>
(1)
012114                                  BR0     TMTDA                   ;IF DATA TO SEND-- BRANCH
(1)     000053                          MICPC=MICPC+1
(1)  012114  112000                     <JUMP!BR0CON!<TMTDA-INIT&3000*4>!<TMTDA-INIT&777/2>>
(1)
                                        .IF DF  $LOW
                                        ALWAYS  I1
                                        ;
                                XEXIT:  SP      BR,SELB,SP2
                                        .ENDC
012116                          I1:     BRWRTE  IBUS,RCVCON             ;READ LINE UNIT RECEIVE CONTROL WORD
(1)     000054                          MICPC=MICPC+1
(1)  012116  020640                     <MOVE!WRTEBR!IBUS!<RCVCON>>
(1)
012120                                  .BR4    BR,SELA,SP3!PAGE1       ;BRANCH BASED UPON RECV STATE
(1)     000055                          MICPC=MICPC+1
(1)  012120  167203                     <JUMP! BR4CON!BR!SELA!SP3!PAGE1>
(1)
012122                          I2:     LDMA    IMM,TABST               ;POINT TO TABLE UPDATE STATE
(1)     000056                          MICPC=MICPC+1
(1)        001                          .IF IDN IMM,IMM
(1)  012122  010210                     <MOVE!LDMAR!IMM!<TABST&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<TABST>>
(1)                                     .ENDC
(1)
012124                                  .ALWAY  MEMX,SELB,0
(1)     000057                          MICPC=MICPC+1
(1)  012124  140620                     <JUMP!ALCOND!MEMX!SELB!0>
(1)
012126                          I3:
                                        .IF NDF $LOW
012126                                  STATE   TMTA+2                  ;GET IDLE TRANSMIT STATE + 1
(1)     000060                          MICPC=MICPC+1
(1)  012126  000404                     <MOVE!WRTEBR!IMM!<TMTA+2-INIT&777/2>>
012130                                  NOP     BR,SUB,SP2              ;SUBTRACT FROM CURRENT STATE
(1)     000061                          MICPC=MICPC+1
(1)  012130  060342                     <BR!SUB!SP2>
(1)
012132                                  C       TMTDA                   ;NON-IDLE STATE
(1)     000062                          MICPC=MICPC+1
(1)  012132  111000                     <JUMP!CCOND!<TMTDA-INIT&3000*4>!<TMTDA-INIT&777/2>>
(1)
                                        .ENDC
```



```
012134                          IDLE0:  SPBR    IBUS,UBBR,SP0           ;TIMER EXPIRES?
(1)     000063                          MICPC=MICPC+1
(1)  012134  123620                     <MOVE!SPBRX!IBUS!UBBR!SP0>
(1)
012136                                  BR4     TIMSRV
(1)     000064                          MICPC=MICPC+1
(1)  012136  113255                     <JUMP!BR4CON!<TIMSRV-INIT&3000*4>!<TIMSRV-INIT&777/2>>
(1)
012140                                  SP      IBUS,RCVCON,SP0         ;READ THE RECEIVE CONTROL REGISTER
(1)     000065                          MICPC=MICPC+1
(1)  012140  023240                     <MOVE!SPX!IBUS!RCVCON!SP0>
(1)
012142                                  BRWRTE  BR,AA!SP0               ;SHIFT IT LEFT
(1)     000066                          MICPC=MICPC+1
(1)  012142  060520                     <MOVE!WRTEBR!BR!<AA!SP0>>
(1)
012144                                  BR7     I1                      ;RECEIVE ACTIVE, DON'T DO PORT STATUS
(1)     000067                          MICPC=MICPC+1
(1)  012144  103454                     <JUMP!BR7CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
(1)
012146                                  LDMA    IMM,PRTST               ;ADDRESS PORT STATE
(1)     000070                          MICPC=MICPC+1
(1)        001                          .IF IDN IMM,IMM
(1)  012146  010211                     <MOVE!LDMAR!IMM!<PRTST&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<PRTST>>
(1)                                     .ENDC
(1)
012150                                  .ALWAY  MEMX,SELB,0             ;INDEX
(1)     000071                          MICPC=MICPC+1
(1)  012150  140620                     <JUMP!ALCOND!MEMX!SELB!0>
(1)
```

```
                                .SBTTL  BASSRV---- BASE SERVICE ROUTINE
012152                  BASSRV: PSTATE  NIDLE2
012152                          MEM     IMM,<<NIDLE2-INIT&777/2>>
                                MICPC=MICPC+1
000072                          <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
012152  002533
                                LDMA    IMM,BASE                        ;CLEAR TO MAR SO IT POINTS TO BASE POINT
                                MICPC=MICPC+1
000073                          .IF IDN IMM,IMM
                                <MOVE!LDMAR!IMM!<BASE&377>>
001                             .IFF
012154  010017                  <MOVE!LDMAR!IMM!<BASE>>
                                .ENDC
                                MEMINC  IBUS,PORT1              ;READ CSR4
                                MICPC=MICPC+1
000074                          <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
012156  136500
                                MEMINC  IBUS,PORT2              ;READ CSR5
                                MICPC=MICPC+1
000075                          <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
012160  136520
                                MEM     IBUS,PORT4
                                MICPC=MICPC+1
000076                          <MOVE!WRMEM!IBUS!<PORT4>>
012162  122560
                                SP      IBUS,INCON,SP0          ;READ INPUT CONTROL CSR
                                MICPC=MICPC+1
000077                          <MOVE!SPX!IBUS!INCON!SP0>
012164  123000
                                BRWRTE  IMM,100                 ;CLEAR THE BR
                                MICPC=MICPC+1
000100                          <MOVE!WRTEBR!IMM!<100>>
012166  000500
                                OUT     BR,<AANDB!OINCON>       ;CLEAR THE INCONTROL CSR
                                MICPC=MICPC+1
000101                          <MOVE!WROUTX!BR!<AANDB!OINCON>>
012170  061260
                                OUTPUT  IMM,<120!OMODEM>        ;MASK FOR HDX AND DTR
                                MICPC=MICPC+1
000102                          <MOVE!WROUT!IMM!<120!OMODEM>>
012172  002133
                                BRWRTE  MEMX,SELB               ;READ SEL6
                                MICPC=MICPC+1
000103                          <MOVE!WRTEBR!MEMX!<SELB>>
012174  040620
                                BR4     RESUME                  ;IF SET RESUME
                                MICPC=MICPC+1
000104                          <JUMP!BR4CON!<RESUME-INIT&3000*4>!<RESUME-INIT&777/2>>
012176  103113
                                LDMA    IMM,T                   ;LOAD ADDRESS OF TYPE FIELD
                                MICPC=MICPC+1
000105                          .IF IDN IMM,IMM
                                <MOVE!LDMAR!IMM!<T&377>>
001                             .IFF
012200  010151                  <MOVE!LDMAR!IMM!<T>>
```



```
                                .ENDC
                                MEMINC  IMM,6                   ;WRITE START TYPE TO MEMORY
                                MICPC=MICPC+1
000106                          <MOVE!WRMEM!INCMAR!IMM!<6>>
012202  016406
                                MEM     IMM,300                 ;WRITE SELECT AND FINAL TO MEMORY
                                MICPC=MICPC+1
000107                          <MOVE!WRMEM!IMM!<300>>
012204  002700
                                SP      BR,DECA,SP1     ;TURN OFF INIT MODE
                                MICPC=MICPC+1
000110                          <MOVE!SPX!BR!DECA!SP1>
012206  063151
                        BS1:    BRWRTE  IMM,241         ;SET OK TO SEND,STARTMODE AND UNNUM PENDING
                                MICPC=MICPC+1
000111                          <MOVE!WRTEBR!IMM!<241>>
012210  000641
                                ALWAYS  SA3
                                MICPC=MICPC+1
000112                          <JUMP!ALCOND!<SA3-INIT&3000*4>!<SA3-INIT&777/2>>
012212  110737
                        RESUME: SP      IMM,SP4,4               ;SET UP SP4 FOR COUNTING NPRS
                                MICPC=MICPC+1
000113                          <MOVE!SPX!IMM!SP4!4>
012214  003004
                                SP      BR,INCA,SP10            ;SET UNNUMB MESSAGE PENDING TO
                                MICPC=MICPC+1
000114                          <MOVE!SPX!BR!INCA!SP10>
012216  063070
                                                                ;TRICK TRANSMITTER CODE
012220                          LDMA    IMM,BASE                ;ADDRESS BASE TABLE ADDRESS
                                MICPC=MICPC+1
000115                          .IF IDN IMM,IMM
001                             <MOVE!LDMAR!IMM!<BASE&377>>
012220  010017                  .IFF
                                <MOVE!LDMAR!IMM!<BASE>>
                                .ENDC
012222                          STATE   FUDGE                   ;SET TMTR STATE TO ENTER TABLE UPDATE
                                MICPC=MICPC+1
000116                          <MOVE!WRTEBR!IMM!<FUDGE-INIT&777/2>>
012222  000743
012224                          ALWAYS  TB0                     ;GO SET UP NXT BITS AND ADRESS OF BASE FOR NPRS
                                MICPC=MICPC+1
000117                          <JUMP!ALCOND!<TB0-INIT&3000*4>!<TB0-INIT&777/2>>
012224  110455
                        BS2:    LDMA    IMM,IMG10
                                MICPC=MICPC+1
000120                          .IF IDN IMM,IMM
001                             <MOVE!LDMAR!IMM!<IMG10&377>>
012226  010154                  .IFF
                                <MOVE!LDMAR!IMM!<IMG10>>
                                .ENDC
012230                          SP      MEMX!INCMAR,ADRR,SP10           ;RESTORE BIT 1 OF SP10
                                MICPC=MICPC+1
000121
```

```
(1)  012230  057310                    <MOVE!SPX!MEMX!INCMAR!AORB!SP10>
(1)
616  012232                            SP      MEMX!INCMAR,SELB,SP11          ;RESTORE SP11
(1)          000122                    MICPC=MICPC+1
(1)  012232  057231                    <MOVE!SPX!MEMX!INCMAR!SELB!SP11>
(1)
617  012234                            SP      MEMX!INCMAR,SELB,SP12          ;RESTORE SP12
(1)          000123                    MICPC=MICPC+1
(1)  012234  057232                    <MOVE!SPX!MEMX!INCMAR!SELB!SP12>
(1)
618  012236                            SP      MEMX!INCMAR,SELB,SP14          ;RESTORE SP14
(1)          000124                    MICPC=MICPC+1
(1)  012236  057234                    <MOVE!SPX!MEMX!INCMAR!SELB!SP14>
(1)
619  012240                            SP      MEMX!INCMAR,SELB,SP16          ;RESTORE SP16
(1)          000125                    MICPC=MICPC+1
(1)  012240  057236                    <MOVE!SPX!MEMX!INCMAR!SELB!SP16>
(1)
620  012242                            SP      MEMX,SELB,SP17          ;RESTORE        SP17
(1)          000126                    MICPC=MICPC+1
(1)  012242  043237                    <MOVE!SPX!MEMX!SELB!SP17>
(1)
621  012244                            SP      BR,DECA,SP10            ;TURN OFF UNNUM MESSAGE PENDING AND
(1)          000127                    MICPC=MICPC+1
(1)  012244  063170                    <MOVE!SPX!BR!DECA!SP10>
(1)
622                                                                    ;ZERO THE BRG
623  012246                            SP      BR,DECA,SP1             ;CLEAR INIT MODE
(1)          000130                    MICPC=MICPC+1
(1)  012246  063161                    <MOVE!SPX!BR!DECA!SP1>
(1)
624  012250                            BRWRTE  IMM,200                 ;SET OK TO SEND
(1)          000131                    MICPC=MICPC+1
(1)  012250  000600                    <MOVE!WRTEBR!IMM!<200>>
(1)
625  012252                            ALWAYS  SA3
(1)          000132                    MICPC=MICPC+1
(1)  012252  110737                    <JUMP!ALCOND!<SA3-INIT&3000*4>!<SA3-INIT&777/2>>
(1)
```



```
627                                    .SBTTL  NIDLE2---NO CSR ACTIVITY STATE
628  012254                    NIDLE2: BRWRTE  BR,SELA!SP1             ;READ PORT STATUS WORD
(1)          000133                    MICPC=MICPC+1
(1)  012254  060601                    <MOVE!WRTEBR!BR!<SELA!SP1>>
(1)
629             001                    .IF NDF $LOW
630  012256                            BR4     NIDLE5                  ;INTERRUPT PENDING?---BRANCH
(1)          000134                    MICPC=MICPC+1
(1)  012256  103141                    <JUMP!BR4CON!<NIDLE5-INIT&3000*4>!<NIDLE5-INIT&777/2>>
(1)
631             000                    .ENDC
632             001                    .IF DF  $LOW
633                                    BR4     OUTINT
634             000                    .ENDC
635  012260                            SPBR    IBUS,INCON,SP0          ;READ INPUT CONTROL CSR
(1)          000135                    MICPC=MICPC+1
(1)  012260  123400                    <MOVE!SPBRX!IBUS!INCON!SP0>
(1)
636  012262                            BRSHFT                          ;SHIFT IT RIGHT
(1)          000136                    MICPC=MICPC+1
(1)  012262  001620                    <MOVE!SHFTBR!WRTEBR!SELB>
(1)
637  012264                            BR4     INWAT1                  ;IF RQI SET -- BRANCH
(1)          000137                    MICPC=MICPC+1
(1)  012264  103146                    <JUMP!BR4CON!<INWAT1-INIT&3000*4>!<INWAT1-INIT&777/2>>
(1)
638                                                                    ;TO RE-READ THE IN CNTRL REGISTER TO AVOID
639                                                                    ;A RACE IN MICRO-P READ/UNIBUS WRITE
640             001                    .IF NDF $LOW
641  012266                            ALWAYS  IDLE
(1)          000140                    MICPC=MICPC+1
(1)  012266  100451                    <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
642             000                    .ENDC
643  012270                    NIDLE6:
644             001                    .IF DF  $LOW
645                               10$:  SPBR    IBUS,MODEM,SP0          ;READ MODEM CONTROL CSR
646                                     BRWRTE  BR,AA!SP0               ;SHIFT IT LEFT
647                                     BR4     SETDSR                  ;IF DSR SET, CLEAR FLAG
648                                     BRWRTE  BR,SELA!SP10            ;READ LINE STATUS WORD
649                                     BRSHFT
650                                     BR4     IDLE                    ;START MODE
651                                     BRWRTE  BR,AA!SP1               ;READ PORT STATUS WORD
652                                     BR1     IDLE                    ;INIT MODE
653                                     BR7     IDLE                    ;DISCONNECT ERROR ALREADY SENT
654                                     SPBR    IBUS,MAIN,SP0   ;READ THE MAINT REGISTER
655                                     BRWPTE  BR,ADD!SP0              ;SHIFT LEFT
656                                     BR4     IDLE                    ;LU LOOP -- EXIT
657                                     BRWRTE  IMM,100                 ;WRITE DISCONNECT ERROR
658                                     SP      BR,AORB,SP1             ;FLAG ERROR RECORDED
659                                     ALWAYS  ERRXX                   ;MAKE A CONTROL OUT
660                               SETDSR: BRWRTE  IMM,277                 ;CLEAR DISCONNECT ERROR FLAG
661                                     ALWAYS  CLRIDL
662             000                    .ENDC
663             001                    .IF NDF $LOW
664  012270                    NIDLE5: BSTATE  OUTINT                  ;SET STATE FOR INTERRUPT PROCESSING
```

```
(1)  012270                          MEM     IMM,<<OUTINT-INIT&777/2>>
(2)         000141                   MICPC=MICPC+1
(2)  012270 002614                   <MOVE!WRMEM!IMM!<<OUTINT-INIT&777/2>>>
(2)
665  012272                          ALWAYS  IDLE
(1)         000142                   MICPC=MICPC+1
(1)  012272 100451                   <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
666            000                   .ENDC
```



```
668
669  012274                          .SBTTL INWAIT---WAIT FOR RQI TO CLEAR
INWAIT: SPBR    IBUS,INCON,SP0       ;READ INPUT CONTROL CSR
(1)         000143                   MICPC=MICPC+1
(1)  012274 123400                   <MOVE!SPBRX!IBUS!INCON!SP0>
(1)
670  012276                          BRWRTE  BR,<AA!SP0>          ;SHIFT IT LEFT
(1)         000144                   MICPC=MICPC+1
(1)  012276 060520                   <MOVE!WRTEBR!BR!<AA!SP0>>
(1)
671  012300                          BR7     NIDLE3               ;INTERRUPT ENABLE HAS BEEN SET
(1)         000145                   MICPC=MICPC+1
(1)  012300 103550                   <JUMP!BR7CON!<NIDLE3-INIT&3000*4>!<NIDLE3-INIT&777/2>>
(1)
672
673  012302                          INWAT1: SPBR    IBUS,INCON,SP0       ;READ THE INPUT CONTROL CSR
(1)         000146                   MICPC=MICPC+1
(1)  012302 123400                   <MOVE!SPBRX!IBUS!INCON!SP0>
(1)
674  012304                          BR7     INWAT2               ;READY IN STILL SET
(1)         000147                   MICPC=MICPC+1
(1)  012304 103557                   <JUMP!BR7CON!<INWAT2-INIT&3000*4>!<INWAT2-INIT&777/2>>
(1)
675  012306                          NIDLE3: PSTATE  INWAT1               ;UPDATE STATE TO INPUT
(1)  012306                          MEM     IMM,<<INWAT1-INIT&777/2>>
(2)         000150                   MICPC=MICPC+1
(2)  012306 002546                   <MOVE!WRMEM!IMM!<<INWAT1-INIT&777/2>>>
(2)
676  012310                          BRWRTE  BR,AA!SP0            ;SHIFT CSR LEFT
(1)         000151                   MICPC=MICPC+1
(1)  012310 060520                   <MOVE!WRTEBR!BR!<AA!SP0>>
(1)
677  012312                          BR7     ININT
(1)         000152                   MICPC=MICPC+1
(1)  012312 117460                   <JUMP!BR7CON!<ININT-INIT&3000*4>!<ININT-INIT&777/2>>
(1)
678  012314                          PSTATE  INWAIT               ;UPDATE STATE POINTER TO NO INTERRUPT GENERATED
(1)  012314                          MEM     IMM,<<INWAIT-INIT&777/2>>
(2)         000153                   MICPC=MICPC+1
(2)  012314 002543                   <MOVE!WRMEM!IMM!<<INWAIT-INIT&777/2>>>
(2)
679  012316                          NIDLE4: BRWRTE  IMM,200
(1)         000154                   MICPC=MICPC+1
(1)  012316 000600                   <MOVE!WRTEBR!IMM!<200>>
(1)
680  012320                          OUT     BR,AORB!OINCON       ;SET THE RDYI
(1)         000155                   MICPC=MICPC+1
(1)  012320 061300                   <MOVE!WROUTX!BR!<AORB!OINCON>>
(1)
681  012322                          ALWAYS  IDLE
(1)         000156                   MICPC=MICPC+1
(1)  012322 100451                   <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
682                                 ;
683  012324                          INWAT2: BRSHFT                       ;SHIFT THE BR RIGHT
(1)         000157                   MICPC=MICPC+1
(1)  012324 001620                   <MOVE!SHFTBR!WRTEBR!SELB>
```

```
(1)
684                 001                     .IF DF  $LOW
685                                         BR4     NIDLE6              ;RQI SET--- GO AWAY
686                 000                     .ENDC
687                 001                     .IF NDF $LOW
688  012326                                 BR4     IDLE
(1)             000160                     MICPC=MICPC+1
(1)  012326     103051                     <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
689  012330                                 PSTATE  INSRV               ;SET NEXT STATE TO INPUT SERVICE
(1)  012330                                 MEM     IMM,<<INSRV-INIT&777/2>>
(2)             000161                     MICPC=MICPC+1
(2)  012330     002563                     <MOVE!WRMEM!IMM!<<INSRV-INIT&777/2>>>
(2)
690  012332                                 ALWAYS  IDLE
(1)             000162                     MICPC=MICPC+1
(1)  012332     100451                     <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
691                 000                     .ENDC
692  012334                         INSRV:  SPBR    IBUS,INCON,SP0          ;READ THE INPUT CONTROL CSR
(1)             000163                     MICPC=MICPC+1
(1)  012334     123400                     <MOVE!SPBRX!IBUS!INCON!SP0>
(1)
693  012336                                 BR1     30$                 ;--SENSE OR BASE
(1)             000164                     MICPC=MICPC+1
(1)  012336     102600                     <JUMP!BR1CON!<30$-INIT&3000*4>!<30$-INIT&777/2>>
(1)
694  012340                                 BR0     10$                 ;CNTL I
(1)             000165                     MICPC=MICPC+1
(1)  012340     102172                     <JUMP!BR0CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
(1)
695  012342                                 BRSHFT                      ;MUST BE BA/CC=SHIFT FOR IN OR OUT
(1)             000166                     MICPC=MICPC+1
(1)  012342     001620                     <MOVE!SHFTBR!WRTEBR!SELB>
(1)
696  012344                                 BR1     15$
(1)             000167                     MICPC=MICPC+1
(1)  012344     102574                     <JUMP!BR1CON!<15$-INIT&3000*4>!<15$-INIT&777/2>>
(1)
697  012346                                 PSTATE  TBASRV              ;TRANSMITTER
(1)  012346                                 MEM     IMM,<<TBASRV-INIT&777/2>>
(2)             000170                     MICPC=MICPC+1
(2)  012346     002700                     <MOVE!WRMEM!IMM!<<TBASRV-INIT&777/2>>>
(2)
698  012350                                 ALWAYS  20$
(1)             000171                     MICPC=MICPC+1
(1)  012350     100575                     <JUMP!ALCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
(1)
699  012352                         10$:    PSTATE  CTLSRV
(1)  012352                                 MEM     IMM,<<CTLSRV-INIT&777/2>>
(2)             000172                     MICPC=MICPC+1
(2)  012352     002657                     <MOVE!WRMEM!IMM!<<CTLSRV-INIT&777/2>>>
(2)
700  012354                                 ALWAYS  20$
(1)             000173                     MICPC=MICPC+1
(1)  012354     100575                     <JUMP!ALCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
```



```
(1)
701  012356                         15$:    PSTATE  RBASRV
(1)  012356                                 MEM     IMM,<<RBASRV-INIT&777/2>>
(2)             000174                     MICPC=MICPC+1
(2)  012356     002721                     <MOVE!WRMEM!IMM!<<RBASRV-INIT&777/2>>>
(2)
702  012360                         20$:    BRWRTE  BR,SELA!SP1             ;INIT MODE
(1)             000175                     MICPC=MICPC+1
(1)  012360     060601                     <MOVE!WRTEBR!BR!<SELA!SP1>>
(1)
703  012362                                 BR0     PROCER                  ;IF INIT MODE--ERROR
(1)             000176                     MICPC=MICPC+1
(1)  012362     102201                     <JUMP!BR0CON!<PROCER-INIT&3000*4>!<PROCER-INIT&777/2>>
(1)
704  012364                                 ALWAYS  IDLE
(1)             000177                     MICPC=MICPC+1
(1)  012364     100451                     <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
705  012366                         30$:    BR0     INSRV1                          ;IF BASE---PROCESS
(1)             000200                     MICPC=MICPC+1
(1)  012366     102211                     <JUMP!BR0CON!<INSRV1-INIT&3000*4>!<INSRV1-INIT&777/2>>
(1)
706  012370                         PROCER: PSTATE  NIDLE2                  ;RESET PORT STATUS
(1)  012370                                 MEM     IMM,<<NIDLE2-INIT&777/2>>
(2)             000201                     MICPC=MICPC+1
(2)  012370     002533                     <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
(2)
707  012372                                 BRWRTE  IMM,100                 ;CLEAR INPUT CONTROL CSR
(1)             000202                     MICPC=MICPC+1
(1)  012372     000500                     <MOVE!WRTEBR!IMM!<100>>
(1)
708  012374                                 OUT     BR,AANDB!OINCON         ;;
(1)             000203                     MICPC=MICPC+1
(1)  012374     061260                     <MOVE!WROUTX!BR!<AANDB!OINCON>>
(1)
709  012376                                 LDMA    IMM,<<RTHRS+3>>         ;ADDRESS ERROR LINK
(1)             000204                     MICPC=MICPC+1
(1)                 001                     .IF IDN IMM,IMM
(1)  012376     010177                     <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
(1)                                         .IFF
(1)                                         <MOVE!LDMAR!IMM!<<RTHRS+3>>>
(1)                 000                     .ENDC
(1)
710  012400                                 MEMINC  IMM,2
(1)             000205                     MICPC=MICPC+1
(1)  012400     016402                     <MOVE!WRMEM!INCMAR!IMM!<2>>
(1)
711  012402                                 MEM     IMM,0
(1)             000206                     MICPC=MICPC+1
(1)  012402     002400                     <MOVE!WRMEM!IMM!<0>>
(1)
712  012404                                 OUTPUT  MEMX,SELB!OMODEM        ;CLEAR DATA TERMINAL READY
(1)             000207                     MICPC=MICPC+1
(1)  012404     042233                     <MOVE!WROUT!MEMX!<SELB!OMODEM>>
(1)
713  012406                                 ALWAYS  RCERX                   ;POST THE ERROR - FATAL
```

```
(1)          000210                        MICPC=MICPC+1
(1)  012406  114524                        <JUMP!ALCOND!<RCEXX-INIT&3000*4>!<RCEXX-INIT&777/2>>
(1)
714  012410                        INSRV1: BRWRTE  BR,SELA!SP1                     ;INIT MODE?
(1)          000211                        MICPC=MICPC+1
(1)  012410  060601                        <MOVE!WRTEBR!BR!<SELA!SP1>>
(1)
715  012412                                BRO     BASSRV
(1)          000212                        MICPC=MICPC+1
(1)  012412  102072                        <JUMP!BROCON!<BASSRV-INIT&3000*4>!<BASSRV-INIT&777/2>>
(1)
716  012414                                ALWAYS  PROCER                  ;NO = PROCEDURE ERROR
(1)          000213                        MICPC=MICPC+1
(1)  012414  100601                        <JUMP!ALCOND!<PROCER-INIT&3000*4>!<PROCER-INIT&777/2>>
(1)
```



```
718                                     .SBTTL  OUTINT---SET UP OUTPUT INTERRUPT (RDYO)
719  012416                        OUTINT:
720             001                     .IF NDF $LOW
721  012416                             PSTATE  PINT2
(1)  012416                             MEM     IMM,<<PINT2-INIT&777/2>>
(2)          000214                        MICPC=MICPC+1
(2)  012416  002631                        <MOVE!WRMEM!IMM!<<PINT2-INIT&777/2>>>
(2)
722             000                     .ENDC
723             001                     .IF DF  $LOW
724                                     PSTATE  OUTWAIT         ;PORT STATUS TO WAITING FOR OUT
725             000                     .ENDC
726                                                             ;COMPLETION
727  012420                             LDMA    IMM,NXTINT              ;ADDRESS OF NEXT INTERRUPT POINTER
(1)          000215                        MICPC=MICPC+1
(1)             001                     .IF IDN IMM,IMM
(1)  012420  010240                        <MOVE!LDMAR!IMM!<NXTINT&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<NXTINT>>
(1)             000                     .ENDC
(1)
728  012422                             LDMA    MEMX,SELB               ;NEXT INTERRUPT
(1)          000216                        MICPC=MICPC+1
(1)             001                     .IF IDN MEMX,IMM
(1)                                     <MOVE!LDMAR!IMM!<SELB&377>>
(1)                                     .IFF
(1)  012422  050220                     <MOVE!LDMAR!MEMX!<SELB>>
(1)             000                     .ENDC
(1)
729  012424                             SP      IBUS,OCON,SP0           ;READ THE OUTPUT CONTROL CSR
(1)          000217                        MICPC=MICPC+1
(1)  012424  123040                        <MOVE!SPX!IBUS!OCON!SP0>
(1)
730  012426                             OUT     <MEMX!INCMAR>,<AORB!OOCON>      ;WRITE THE OUT CONTROL CSR
(1)          000220                        MICPC=MICPC+1
(1)  012426  055302                        <MOVE!WROUTX!MEMX!INCMAR!<AORB!OOCON>>
(1)
731  012430                             LDMA    MEMX,SELB                       ;ADDRESS LINK
(1)          000221                        MICPC=MICPC+1
(1)             001                     .IF IDN MEMX,IMM
(1)                                     <MOVE!LDMAR!IMM!<SELB&377>>
(1)                                     .IFF
(1)  012430  050220                     <MOVE!LDMAR!MEMX!<SELB>>
(1)             000                     .ENDC
(1)
732  012432                             BRWRTE  <BR!INCMAR>,<AA!SP0>            ;KICK PAST LINK STATUS BYTE
(1)          000222                        MICPC=MICPC+1
(1)  012432  074520                        <MOVE!WRTEBR!BR!INCMAR!<AA!SP0>>
(1)
733                                                             ;SHIFT CSR0 IMAGE LEFT
734                                                             ;***DO NOT CHANGE BR UNTIL BR7***
735  012434                             OUT     <MEMX!INCMAR>,<SELB!OPORT1>     ;WRITE LOW BYTE OF BA TO CSR
(1)          000223                        MICPC=MICPC+1
(1)  012434  055224                        <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT1>>
(1)
736  012436                             OUT     <MEMX!INCMAR>,<SELB!OPORT2>     ;WRITE HIGH BYTE OF BA TO CSR
```

```
(1)          000224                     MICPC=MICPC+1
(1)  012436  055225                     <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT2>>
(1)
737  012440                              OUT     <MEMX!INCMAR>,<SELB!OPORT4>     ;WRITE HIGH BYTE OF COUNT TO CSR
(1)          000225                     MICPC=MICPC+1
(1)  012440  055227                     <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT4>>
(1)
738  012442                              OUT     <MEMX!INCMAR>,<SELB!OPORT3>     ;WRITE THE LOW BYTE OF COUNT
(1)          000226                     MICPC=MICPC+1
(1)  012442  055226                     <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT3>>
(1)
739                                                                 ;***HERE IS BR7***
740  012444                              BR7     PE1                 ;INTERRUPT ENABLE IS SET
(1)          000227                     MICPC=MICPC+1
(1)  012444  103757                     <JUMP!BR7CON!<PE1-INIT&3000*4>!<PE1-INIT&777/2>>
(1)
741                                                                 ;GENERATE AN INTERRUPT
742             001                     .IF NDF $LOW
743  012446                             ALWAYS  IDLE
(1)          000230                     MICPC=MICPC+1
(1)  012446  100451                     <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
744  012450                         PINT2:  PSTATE  OUTWAIT
(1)  012450                             MEM     IMM,<<OUTWAIT-INIT&777/2>>
(2)          000231                     MICPC=MICPC+1
(2)  012450  002652                     <MOVE!WRMEM!IMM!<<OUTWAIT-INIT&777/2>>>
(2)
745             000                     .ENDC
746             001                     .IF DF  $LOW
747                                 PINT2:
748             000                     .ENDC
749  012452                             LDMA    IMM,NXTINT              ;ADDRESS NEXT INTERRUPT QUEUE
(1)          000232                     MICPC=MICPC+1
(1)             001                     .IF IDN IMM,IMM
(1)  012452  010240                     <MOVE!LDMAR!IMM!<NXTINT&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<NXTINT>>
(1)             000                     .ENDC
(1)
750  012454                             SP      MEMX,SELB,SP0           ;COPY ADDRESS FOR NEXT INT TO SP0
(1)          000233                     MICPC=MICPC+1
(1)  012454  043220                     <MOVE!SPX!MEMX!SELB!SP0>
(1)
751  012456                             MEM     IMM,INTSTK              ;ASSUME WRAP AROUND CASE
(1)          000234                     MICPC=MICPC+1
(1)  012456  002642                     <MOVE!WRMEM!IMM!<INTSTK>>
(1)
752  012460                             BRWRTE  IMM,<<MMEND-2>>         ;ADDRESS OF LAST INT IN STACK
(1)          000235                     MICPC=MICPC+1
(1)  012460  000776                     <MOVE!WRTEBR!IMM!<<MMEND-2>>>
(1)
753  012462                             CMP     BR,SP0                  ;SHOULD WE WRAP
(1)          000236                     MICPC=MICPC+1
(1)  012462  060360                     <SUBTC!BR!SP0>
(1)
754  012464                             Z       5$                      ;YES--BRANCH
```



```
(1)          000237                     MICPC=MICPC+1
(1)  012464  101642                     <JUMP!ZCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
(1)
755  012466                             BRWRTE  IMM,2                   ;OFFSET FOR NEXT POINTER
(1)          000240                     MICPC=MICPC+1
(1)  012466  000402                     <MOVE!WRTEBR!IMM!<2>>
(1)
756  012470                             MEM     BR,ADD!SP0              ;UPDATE POINTER
(1)          000241                     MICPC=MICPC+1
(1)  012470  062400                     <MOVE!WRMEM!BR!<ADD!SP0>>
(1)
757  012472                         5$:     SP      MEMX,SELB,SP0           ;COPY POINTER TO SP0
(1)          000242                     MICPC=MICPC+1
(1)  012472  043220                     <MOVE!SPX!MEMX!SELB!SP0>
(1)
758  012474                             LDMA    IMM,NXTSP               ;PICK UP START OF IN QUEUE
(1)          000243                     MICPC=MICPC+1
(1)             001                     .IF IDN IMM,IMM
(1)  012474  010241                     <MOVE!LDMAR!IMM!<NXTSP&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<NXTSP>>
(1)             000                     .ENDC
(1)
759  012476                             CMP     MEMX,SP0                ;COMPARE TO END
(1)          000244                     MICPC=MICPC+1
(1)  012476  040360                     <SUBTC!MEMX!SP0>
(1)
760  012500                             Z       10$                     ;IF EQUAL--CLEAR INT PENDING
(1)          000245                     MICPC=MICPC+1
(1)  012500  101647                     <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
(1)
761  012502                             ALWAYS  IDLE
(1)          000246                     MICPC=MICPC+1
(1)  012502  100451                     <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
762  012504                         10$:    BRWRTE  IMM,357                 ;MASK TO CLEAR INT PENDING
(1)          000247                     MICPC=MICPC+1
(1)  012504  000757                     <MOVE!WRTEBR!IMM!<357>>
(1)
763  012506                         CLRIDL: SP      BR,AANDB,SP1
(1)          000250                     MICPC=MICPC+1
(1)  012506  063261                     <MOVE!SPX!BR!AANDB!SP1>
(1)
764  012510                             ALWAYS  IDLE
(1)          000251                     MICPC=MICPC+1
(1)  012510  100451                     <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
```

```
                                        .SBTTL  OUTWAI--WAIT FOR RDYO TO GO AWAY
012512                          OUTWAI: SPBR    IBUS,OCON,SP0           ;READ OUTPUT CONTROL CSR
        000252                          MICPC=MICPC+1
012512  123440                          <MOVE!SPBRX!IBUS!OCON!SP0>

           001                          .IF DF  $LOW
                                        BR7     NIDLE6                  ;RDYO SET --GET OUT
           000                          .ENDC
           001                          .IF NDF $LOW
012514                                  BR7     IDLE
        000253                          MICPC=MICPC+1
012514  103451                          <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>

           000                          .ENDC
012516                                  BRWRTE  IMM,100         ;CLEAR CONTROL BITS
        000254                          MICPC=MICPC+1
012516  000500                          <MOVE!WRTEBR!IMM!<100>>

012520                                  OUT     BR,OOCON!AANDB
        000255                          MICPC=MICPC+1
012520  061262                          <MOVE!WROUTX!BR!<OOCON!AANDB>>

012522                                  ALWAYS  INS13
        000256                          MICPC=MICPC+1
012522  100671                          <JUMP!ALCOND!<INS13-INIT&3000*4>!<INS13-INIT&777/2>>
```



```
                                        .SBTTL  CTLSRV--CNTL I SERVICE
012524                          CTLSRV: SPBR    IBUS,PORT4,SP0          ;TO SP0
        000257                          MICPC=MICPC+1
012524  123560                          <MOVE!SPBRX!IBUS!PORT4!SP0>

012526                                  BRSHFT
        000260                          MICPC=MICPC+1
012526  001620                          <MOVE!SHFTBR!WRTEBR!SELB>

012530                                  BR1     HDSEL                   ;IF SET IS HALF DUPLEX
        000261                          MICPC=MICPC+1
012530  102754                          <JUMP!BR1CON!<HDSEL-INIT&3000*4>!<HDSEL-INIT&777/2>>

012532                                  OUTPUT  IMM,<100!OMODEM>        ;MASK DTR, TURN OFF HDX
        000262                          MICPC=MICPC+1
012532  002113                          <MOVE!WROUT!IMM!<100!OMODEM>>

012534                          INS11:  BRWRTE  DP,<SELA!SP0>           ;RESTORE THE CNTL WORD
        000263                          MICPC=MICPC+1
012534  060600                          <MOVE!WRTEBR!DP!<SELA!SP0>>

012536                                  BR0     CBOOT                   ;IF SET IS BOOT
        000264                          MICPC=MICPC+1
012536  102273                          <JUMP!BR0CON!<CBOOT-INIT&3000*4>!<CBOOT-INIT&777/2>>

012540                          INS12:  SP      IBUS,INCON,SP0          ;READ THE INPUT CONTROL CSR
        000265                          MICPC=MICPC+1
012540  123000                          <MOVE!SPX!IBUS!INCON!SP0>

012542                                  BRWRTE  IMM,100                 ;ZERO THE BR REGISTER EXCEPT INT ENABLE
        000266                          MICPC=MICPC+1
012542  000500                          <MOVE!WRTEBR!IMM!<100>>

012544                                  OUT     BR,<AANDB!OINCON>       ;CLEAR IN CONTROL CSR
        000267                          MICPC=MICPC+1
012544  061260                          <MOVE!WROUTX!BR!<AANDB!OINCON>>

012546                                  LDMA    IMM,PRTST               ;ADDRESS PORT STATE
        000270                          MICPC=MICPC+1
           001                          .IF IDN IMM,IMM
012546  010211                          <MOVE!LDMAR!IMM!<PRTST&377>>
                                        .IFF
                                        <MOVE!LDMAR!IMM!<PRTST>>
           000                          .ENDC

012550                          INS13:  BSTATE  NIDLE2
012550                                  MEM     IMM,<<NIDLE2-INIT&777/2>>
        000271                          MICPC=MICPC+1
012550  002533                          <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>

012552                                  ALWAYS  IDLE
        000272                          MICPC=MICPC+1
012552  100451                          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>

012554                          CBOOT:  BRWRTE  IMM,200                 ;MASK FOR BOOT MODE
```

```
(1)          000273                     MICPC=MICPC+1
(1)  012554  000600                     <MOVE!WRTEBR!IMM!<200>>
(1)
793  012556                              SP      BR,AORB,SP1             ;IN PORT STATUS WORD
(1)          000274                     MICPC=MICPC+1
(1)  012556  063301                     <MOVE!SPX!BR!AORB!SP1>
(1)
794  012560                              BRWRTE  IMM,204                 ;MASK FOR OK TO SEND AND LINE IDLE
(1)          000275                     MICPC=MICPC+1
(1)  012560  000604                     <MOVE!WRTEBR!IMM!<204>>
(1)
795  012562                              SP      BR,SELB,SP10            ;IN LINE STATUS
(1)          000276                     MICPC=MICPC+1
(1)  012562  063230                     <MOVE!SPX!BR!SELB!SP10>
(1)
796  012564                              ALWAYS  INS12
(1)          000277                     MICPC=MICPC+1
(1)  012564  100665                     <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
(1)
```



```
798                                      .SBTTL  TBASRV--TRANSMITTER BUFFER ADDRESS SERVICE
799  012566                     TBASRV: LDMA    IMM,ETC                 ;GET POINTER TO END OF TMT CHAIN
(1)          000300                     MICPC=MICPC+1
(1)              001                    .IF IDN IMM,IMM
(1)  012566  010070                     <MOVE!LDMAR!IMM!<ETC&377>>
(1)                                      .IFF
(1)                                      <MOVE!LDMAR!IMM!<ETC>>
(1)              000                    .ENDC
(1)
800  012570                              LDMA    MEMX,<SELB!SPX!SP0>     ;FIND THE LINK
(1)          000301                     MICPC=MICPC+1
(1)              001                    .IF IDN MEMX,IMM
(1)                                      <MOVE!LDMAR!IMM!<SELB!SPX!SP0&377>>
(1)                                      .IFF
(1)  012570  053220                     <MOVE!LDMAR!MEMX!<SELB!SPX!SP0>>
(1)              000                    .ENDC
(1)
801  012572                              MEMINC  IMM,1                   ;BUFFER ASSIGNED IN IN LINK FLAGS
(1)          000302                     MICPC=MICPC+1
(1)  012572  016401                     <MOVE!WRMEM!INCMAR!IMM!<1>>
(1)
802  012574                              BRWRTE  <IMM!INCMAR>,TML8               ;POINT PAST NUMBER FIELD
(1)          000303                     MICPC=MICPC+1
(1)  012574  014543                     <MOVE!WRTEBR!IMM!INCMAR!<TML8>>
(1)
803                                                                          ;SET BR FOR ADDITION TO SP0
804  012576                              MEMINC  IBUS,PORT1
(1)          000304                     MICPC=MICPC+1
(1)  012576  136500                     <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
(1)
805  012600                              MEMINC  IBUS,PORT2
(1)          000305                     MICPC=MICPC+1
(1)  012600  136520                     <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
(1)
806  012602                              MEMINC  IBUS,PORT4
(1)          000306                     MICPC=MICPC+1
(1)  012602  136560                     <MOVE!WRMEM!INCMAR!IBUS!<PORT4>>
(1)
807  012604                              MEMINC  IBUS,PORT3
(1)          000307                     MICPC=MICPC+1
(1)  012604  136540                     <MOVE!WRMEM!INCMAR!IBUS!<PORT3>>
(1)
808  012606                              LDMA    IMM,ETC
(1)          000310                     MICPC=MICPC+1
(1)              001                    .IF IDN IMM,IMM
(1)  012606  010070                     <MOVE!LDMAR!IMM!<ETC&377>>
(1)                                      .IFF
(1)                                      <MOVE!LDMAR!IMM!<ETC>>
(1)              000                    .ENDC
(1)
809  012610                              MEM     IMM,TML1                ;ASSUME QUEUE WRAP AROUND
(1)          000311                     MICPC=MICPC+1
(1)  012610  002471                     <MOVE!WRMEM!IMM!<TML1>>
(1)
810  012612                              CMP     BR,SP0                  ;END OF CHAIN?
(1)          000312                     MICPC=MICPC+1
```

```
(1)  012612  060360                        <SUBTC!BR!SP0>
(1)
811  012614                                 Z       10$                              ;IF YES--BRANCH
(1)          000313                        MICPC=MICPC+1
(1)  012614  101716                        <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
(1)
812  012616                                 BRWRTE  IMM,6                            ;QUEUE ENTRY LENGTH
(1)          000314                        MICPC=MICPC+1
(1)  012616  000406                        <MOVE!WRTEBR!IMM!<6>>
(1)
813  012620                                 MEM     BR,ADD!SP0                       ;UPDATE THE END POINTER IN MEMORY
(1)          000315                        MICPC=MICPC+1
(1)  012620  062400                        <MOVE!WRMEM!BR!<ADD!SP0>>
(1)
814  012622                           10$:  BRWRTE  IMM,2                    ;NUMBERED MSG PENDING MASK
(1)          000316                        MICPC=MICPC+1
(1)  012622  000402                        <MOVE!WRTEBR!IMM!<2>>
(1)
815  012624                                 SP      BR,AORB,SP10             ;UPDATE LINE STATUS
(1)          000317                        MICPC=MICPC+1
(1)  012624  063310                        <MOVE!SPX!BR!AORB!SP10>
(1)
816  012626                                 ALWAYS  INS12
(1)          000320                        MICPC=MICPC+1
(1)  012626  100665                        <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
(1)
```



```
818                                        .SBTTL  RBASRV--RECEIVE BUFFER ADDRESS SERVICE
819  012630                           RBASRV: LDMA    IMM,ERC                          ;ADDRES END OF RECEIVE CHAIN
(1)          000321                        MICPC=MICPC+1
(1)             001                        .IF IDN IMM,IMM
(1)  012630  010023                        <MOVE!LDMAR!IMM!<ERC&377>>
(1)                                        .IFF
(1)                                        <MOVE!LDMAR!IMM!<ERC>>
(1)             000                        .ENDC
(1)
820  012632                                 LDMA    MEMX,<SELB!SPX!SP0>              ;GET THE POINTER TO LINK
(1)          000322                        MICPC=MICPC+1
(1)             001                        .IF IDN MEMX,IMM
(1)                                        <MOVE!LDMAR!IMM!<SELB!SPX!SP0&377>>
(1)                                        .IFF
(1)  012632  053220                        <MOVE!LDMAR!MEMX!<SELB!SPX!SP0>>
(1)             000                        .ENDC
(1)
821  012634                                 MEMINC  IMM,1
(1)          000323                        MICPC=MICPC+1
(1)  012634  016401                        <MOVE!WRMEM!INCMAR!IMM!<1>>
(1)
822  012636                                 MEMINC  IBUS,PORT1
(1)          000324                        MICPC=MICPC+1
(1)  012636  136500                        <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
(1)
823  012640                                 MEMINC  IBUS,PORT2
(1)          000325                        MICPC=MICPC+1
(1)  012640  136520                        <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
(1)
824  012642                                 MEMINC  IBUS,PORT4
(1)          000326                        MICPC=MICPC+1
(1)  012642  136560                        <MOVE!WRMEM!INCMAR!IBUS!<PORT4>>
(1)
825  012644                                 MEMINC  IBUS,PORT3
(1)          000327                        MICPC=MICPC+1
(1)  012644  136540                        <MOVE!WRMEM!INCMAR!IBUS!<PORT3>>
(1)
826                                        ;;;;NOTE INVERTED ORDER OF PORT 3 AND PORT4
827  012646                                 LDMA    IMM,ERC
(1)          000330                        MICPC=MICPC+1
(1)             001                        .IF IDN IMM,IMM
(1)  012646  010023                        <MOVE!LDMAR!IMM!<ERC&377>>
(1)                                        .IFF
(1)                                        <MOVE!LDMAR!IMM!<ERC>>
(1)             000                        .ENDC
(1)
828  012650                                 MEM     IMM,RCL1                 ;ASSUME WRAP AROUND CASE
(1)          000331                        MICPC=MICPC+1
(1)  012650  002424                        <MOVE!WRMEM!IMM!<RCL1>>
(1)
829  012652                                 BRWRTE  IMM,RCL7                 ;GET ADDRESS OF END OF CHAIN AREA
(1)          000332                        MICPC=MICPC+1
(1)  012652  000142                        <MOVE!WRTEBR!IMM!<RCL7>>
(1)
830  012654                                 CMP     BR,SP0                           ;CHECK FOR END
(1)          000333                        MICPC=MICPC+1
```

```
(1)  012654  060360                    <SUBTC!BR!SP0>
(1)
831  012656                            Z       INS12                   ;IF EQUAL BRANCH
(1)          000334                    MICPC=MICPC+1
(1)  012656  101665                    <JUMP!ZCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
(1)
832  012660                            BRWRTE  IMM,5                   ;CALCULATE ADDRESS OF NEXT LINK
(1)          000335                    MICPC=MICPC+1
(1)  012660  000405                    <MOVE!WRTEBR!IMM!<5>>
(1)
833  012662                            MEM     BR,ADD!SP0              ;..
(1)          000336                    MICPC=MICPC+1
(1)  012662  062400                    <MOVE!WRMEM!BR!<ADD!SP0>>
(1)
834  012664                            ALWAYS  INS12                   ;EXIT
(1)          000337                    MICPC=MICPC+1
(1)  012664  100665                    <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
(1)
835  012666                    RA1:    BRWRTE  IMM,317                 ;MASK TO CLEAR START MODE AND CLR ACTIVE
(1)          000340                    MICPC=MICPC+1
(1)  012666  000717                    <MOVE!WRTEBR!IMM!<317>>
(1)
836  012670                            SPBR    BR,AANDB,SP10           ;CLEAR BIT IN LINE STATUS WORD
(1)          000341                    MICPC=MICPC+1
(1)  012670  063670                    <MOVE!SPBRX!BR!AANDB!SP10>
(1)
837  012672                    RA3:    BRWRTE  IMM,0                   ;CLEAR BR
(1)          000342                    MICPC=MICPC+1
(1)  012672  000400                    <MOVE!WRTEBR!IMM!<0>>
(1)
838  012674                            SP      BR,SELB,SP13            ;SET NUMB MESSAGE TYPE IN SP13
(1)          000343                    MICPC=MICPC+1
(1)  012674  063233                    <MOVE!SPX!BR!SELB!SP13>
(1)
839  012676                            STATE   RCVB                    ;CHANGE RECEIVE STATE POINTER TO STATE B
(1)          000344                    MICPC=MICPC+1
(1)  012676  000424                    <MOVE!WRTEBR!IMM!<RCVB-INIT&777/2>>
840  012700                            ALWAYS  REXIT
(1)          000345                    MICPC=MICPC+1
(1)  012700  100450                    <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
(1)
841                                     ;
842             001                     .IF NDF $LOW
843  012702                    ACK:    BRWRTE  BR,AA!SP10              ;READ LINE STATUS SHIFTING LEFT
(1)          000346                    MICPC=MICPC+1
(1)  012702  060530                    <MOVE!WRTEBR!BR!<AA!SP10>>
(1)
844  012704                            BR4     5$                      ;IF START RECD--CLEAR START MODE
(1)          000347                    MICPC=MICPC+1
(1)  012704  103351                    <JUMP!BR4CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
(1)
845  012706                            ALWAYS  IDLE
(1)          000350                    MICPC=MICPC+1
(1)  012706  100451                    <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
846  012710                    5$:     BRWRTE  IMM,327                 ;CLEAR START MODE
```



```
(1)          000351                    MICPC=MICPC+1
(1)  012710  000727                    <MOVE!WRTEBR!IMM!<327>>
(1)
847  012712                            SP      BR,AANDB,SP10           ;IN LINE STATUS
(1)          000352                    MICPC=MICPC+1
(1)  012712  063270                    <MOVE!SPX!BR!AANDB!SP10>
(1)
848  012714                            ALWAYS  RD5
(1)          000353                    MICPC=MICPC+1
(1)  012714  104507                    <JUMP!ALCOND!<RD5-INIT&3000*4>!<RD5-INIT&777/2>>
(1)
849             000                     .ENDC
```

```
851  012716                      HDSEL:  BRWRTE  IMM,100                 ;HD MASK TO BR
(1)          000354                      MICPC=MICPC+1
(1)  012716  000500                      <MOVE!WRTEBR!IMM!<100>>
(1)
852  012720                              SP      BR,AORB,SP10             ;UPDATE PORT STATUS WORD
(1)          000355                      MICPC=MICPC+1
(1)  012720  063310                      <MOVE!SPX!BR!AORB!SP10>
(1)
853  012722                              ALWAYS  INS11
(1)          000356                      MICPC=MICPC+1
(1)  012722  100663                      <JUMP!ALCOND!<INS11-INIT&3000*4>!<INS11-INIT&777/2>>
(1)
854                                      ;
855  012724                      PE1:    BRWRTE  IMM,300                  ;MASK FOR INTERUPT AND VECTOR THROUGH X04
(1)          000357                      MICPC=MICPC+1
(1)  012724  000700                      <MOVE!WRTEBR!IMM!<300>>
(1)
856  012726                              SP      IBUS,UBBR,SP0            ;READ BR CONTROL REG
(1)          000360                      MICPC=MICPC+1
(1)  012726  123220                      <MOVE!SPX!IBUS!UBBR!SP0>
(1)
857  012730                              OUT     BR,<AORB!OBR>            ;INTERRUPT
(1)          000361                      MICPC=MICPC+1
(1)  012730  061311                      <MOVE!WROUTX!BR!<AORB!OBR>>
(1)
858             001                      .IF NDF $LOW
859  012732                              ALWAYS  IDLE
(1)          000362                      MICPC=MICPC+1
(1)  012732  100451                      <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
860             000                      .ENDC
861             001                      .IF DF  $LOW
862                                      ALWAYS  PINT2
863             000                      .ENDC
864                                      ;
865  012734                      HALTED: MEMADR  EM6
(1)          000363                      MICPC=MICPC+1
(1)             001                      .IF B
(1)  012734  002722                      <MOVE!WRMEM!<EM6-INIT&777/2>>
(1)                                      .IFF
(1)                                      <MOVE!WRMEM!!<EM6-INIT&777/2>>
(1)             000                      .ENDC
866                                                                       ;FALL INTO ACLOW
867  012736                      ACLOW:  BRWRTE  IMM,2                    ;CAUSE AN AC LOW
(1)          000364                      MICPC=MICPC+1
(1)  012736  000402                      <MOVE!WRTEBR!IMM!<2>>
(1)
868  012740                              OUT     BR,<SELB!OBR>
(1)          000365                      MICPC=MICPC+1
(1)  012740  061231                      <MOVE!WROUTX!BR!<SELB!OBR>>
(1)
869  012742                      5$:     BRWRTE  IBUS,UBBR                ;WAIT FOR IT TO COMPLETE
(1)          000366                      MICPC=MICPC+1
(1)  012742  120620                      <MOVE!WRTEBR!IBUS!<UBBR>>
(1)
870  012744                              BR1     5$
```



```
(1)          000367                      MICPC=MICPC+1
(1)  012744  102766                      <JUMP!BR1CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
(1)
871  012746                              .ALWAY  MEMX,SELB,PAGE3
(1)          000370                      MICPC=MICPC+1
(1)  012746  154620                      <JUMP!ALCOND!MEMX!SELB!PAGE3>
(1)
872  012750                      CKTIME: BRWRTE  IBUS,UBBR                ;READ BR CONTROL REG
(1)          000371                      MICPC=MICPC+1
(1)  012750  120620                      <MOVE!WRTEBR!IBUS!<UBBR>>
(1)
873  012752                              BR4     HALTED
(1)          000372                      MICPC=MICPC+1
(1)  012752  103363                      <JUMP!BR4CON!<HALTED-INIT&3000*4>!<HALTED-INIT&777/2>>
(1)
874  012754                              ALWAYS  EM1
(1)          000373                      MICPC=MICPC+1
(1)  012754  114725                      <JUMP!ALCOND!<EM1-INIT&3000*4>!<EM1-INIT&777/2>>
(1)
875                                      ;
876  012756                      TBU1:   BRWRTE  IBUS,NPR
(1)          000374                      MICPC=MICPC+1
(1)  012756  120600                      <MOVE!WRTEBR!IBUS!<NPR>>
(1)
877  012760                              BR0     IDLE
(1)          000375                      MICPC=MICPC+1
(1)  012760  102051                      <JUMP!BR0CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
878  012762                              ALWAYS  EC2
(1)          000376                      MICPC=MICPC+1
(1)  012762  114752                      <JUMP!ALCOND!<EC2-INIT&3000*4>!<EC2-INIT&777/2>>
(1)
879  012764                              $ZERO
(1)          000377                      MICPC=MICPC+1
(1)  012764  000000                      000000
(1)
880
```

```
                012766                  .=INIT+1000
                000377                  MICPC=377
                                        .SBTTL  RCVA--ROUTINE TO HANDLE FIRST DDCMP CHARACTER
                                        ;ENTERED FROM IDLE LOOP
                                        ;DETERMINES IF MESSAGE TYPE IS NUMBERED,UNNUMBERED OR BOOT
                                        ;SETS UP APROPRIATE STATES FOR REST OF MESSAGE.
012766                          RCVA:   SP      IBUS,RCVDAT,SP0         ;READ RECEIVE CHARACTER TO SP0
(1)             000400                  MICPC=MICPC+1
(1)  012766     023200                  <MOVE!SPX!IBUS!RCVDAT!SP0>
(1)
012770                                  BRWRTE  BR,SELA!SP1             ;READ PORT STATUS WORD
(1)             000401                  MICPC=MICPC+1
(1)  012770     060501                  <MOVE!WRTEBR!BR!<SELA!SP1>>
(1)
012772                                  BR0     5$                      ;IF INIT MODE---ONLY BOOT OK
(1)             000402                  MICPC=MICPC+1
(1)  012772     106612                  <JUMP!BR0CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
(1)
012774                                  BR7     5$                      ;IF BOOT MODE---ONLY BOOT OK
(1)             000403                  MICPC=MICPC+1
(1)  012774     107412                  <JUMP!BR7CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
(1)
012776                                  BRWRTE  IMM,201                 ;SOH TO BR
(1)             000404                  MICPC=MICPC+1
(1)  012776     000601                  <MOVE!WRTEBR!IMM!<201>>
(1)
013000                                  CMP     BR,SP0                  ;COMPARE BR TO SP0
(1)             000405                  MICPC=MICPC+1
(1)  013000     060360                  <SUBTC!BR!SP0>
(1)
013002                                  Z       RA1                     ;IF EQUAL=IS NUMBERED MESSAGE
(1)             000406                  MICPC=MICPC+1
(1)  013002     101740                  <JUMP!ZCOND!<RA1-INIT&3000*4>!<RA1-INIT&777/2>>
(1)
013004                                  BRWRTE  IMM,5                   ;ENQ TO BR
(1)             000407                  MICPC=MICPC+1
(1)  013004     000405                  <MOVE!WRTEBR!IMM!<5>>
(1)
013006                                  CMP     BR,SP0                  ;COMPARE ENQ TO SP0
(1)             000410                  MICPC=MICPC+1
(1)  013006     060360                  <SUBTC!BR!SP0>
(1)
013010                                  Z       RA2                     ;IF EQUAL=IS UNNUMBERED MESSAGE
(1)             000411                  MICPC=MICPC+1
(1)  013010     105422                  <JUMP!ZCOND!<RA2-INIT&3000*4>!<RA2-INIT&777/2>>
(1)
013012                          5$:     BRWRTE  IMM,220                 ;DLE TO BR
(1)             000412                  MICPC=MICPC+1
(1)  013012     000620                  <MOVE!WRTEBR!IMM!<220>>
(1)
013014                                  CMP     BR,SP0                  ;COMPARE DLE TO SP0
(1)             000413                  MICPC=MICPC+1
(1)  013014     060360                  <SUBTC!BR!SP0>
(1)
013016                                  Z       BOOT                    ;IF EQUAL IS BOOT
(1)             000414                  MICPC=MICPC+1
```



```
(1)  013016     105756                  <JUMP!ZCOND!<BOOT-INIT&3000*4>!<BOOT-INIT&777/2>>
(1)
013020                          FLUSH:  OUTPUT  IMM,<200!ORCVCO>        ;FLUSH INPUT SILO
(1)             000415                  MICPC=MICPC+1
(1)  013020     002212                  <MOVE!WROUT!IMM!<200!ORCVCO>>
(1)
                                                                        ;(LOW ORDER BITS READ ONLY)
013022                                  BRWRTE  IMM,357                 ;MASK TO CLEAR--CLEAR ACTIVE
(1)             000416                  MICPC=MICPC+1
(1)  013022     000757                  <MOVE!WRTEBR!IMM!<357>>
(1)
013024                                  SP      BR,AANDB,SP10           ;IN LINE STATUS WORD
(1)             000417                  MICPC=MICPC+1
(1)  013024     063270                  <MOVE!SPX!BR!AANDB!SP10>
(1)
                   001                  .IF DF  $LOW
                                        ALWAYS  RM1                     ;SET STATE TO RCVA AND RETURN TO IDLE
                   000                  .ENDC
                   001                  .IF NDF $LOW
013026                          RM1:    STATE   RCVA
(1)             000420                  MICPC=MICPC+1
(1)  013026     000400                  <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
013030                                  ALWAYS  REXIT
(1)             000421                  MICPC=MICPC+1
(1)  013030     100450                  <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
(1)
                   000                  .ENDC
013032                          RA2:    STATE   RCVI                    ;CHANGE RECEIVE STATE TO I
(1)             000422                  MICPC=MICPC+1
(1)  013032     000665                  <MOVE!WRTEBR!IMM!<RCVI-INIT&777/2>>
                   001                  .IF NDF $LOW
013034                                  ALWAYS  REXIT
(1)             000423                  MICPC=MICPC+1
(1)  013034     100450                  <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
(1)
                   000                  .ENDC
                   001                  .IF DF  $LOW
013036                          REXIT:  SP      BR,SELB,SP3
                                        ALWAYS  IDLE
                   000                  .ENDC
```

```
                                        .SBTTL  RCVB--ROUTINE TO HANDLE FIRST CHARACTER OF COUNT FIELD
                                        ;ENTERED FROM IDLE LOOP
                                        ;STORES COUNT FIELD AND SETS UP RCVC AS NEXT STATE
                                        ;
013036                          RCVB:
013036                                  SP      IBUS,RCVDAT,SP4         ;READ CHARACTER TO SP4
(1)     000424                          MICPC=MICPC+1
(1)  013036  023204                     <MOVE!SPX!IBUS!RCVDAT!SP4>
(1)
013040                                  LDMA    BR,<SELA!SP14>                  ;LOAD MAR WITH ADDRESS OF CURRENT BA
(1)     000425                          MICPC=MICPC+1
(1)        001                          .IF IDN BR,IMM
(1)                                     <MOVE!LDMAR!IMM!<SELA!SP14&377>>
(1)                                     .IFF
(1)  013040  070214                     <MOVE!LDMAR!BR!<SELA!SP14>>
(1)        000                          .ENDC
(1)
013042                                  BRWRTE  MEMX,INCMAR!SELB                ;READ FLAGS BYTE
(1)     000426                          MICPC=MICPC+1
(1)  013042  054620                     <MOVE!WRTEBR!MEMX!<INCMAR!SELB>>
(1)
013044                                  BR0     RB1                             ;RECV BUFFER ASSIGNED---CONTINUE
(1)     000427                          MICPC=MICPC+1
(1)  013044  106041                     <JUMP!BR0CON!<RB1-INIT&3000*4>!<RB1-INIT&777/2>>
(1)
013046                                  BRWRTF  BR,SELA!SP1                     ;READ STATUS BYTE
(1)     000430                          MICPC=MICPC+1
(1)  013046  060601                     <MOVE!WRTEBR!BR!<SELA!SP1>>
(1)
013050                                  BR7     RB3                             ;MAINT MODE
(1)     000431                          MICPC=MICPC+1
(1)  013050  107437                     <JUMP!BR7CON!<RB3-INIT&3000*4>!<RB3-INIT&777/2>>
(1)
013052                                  LDMA    IMM,T                   ;ERROR--LOAD TYPE FIELD ADDRESS IN MAR
(1)     000432                          MICPC=MICPC+1
(1)        001                          .IF IDN IMM,IMM
(1)  013052  010151                     <MOVE!LDMAR!IMM!<T&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<T>>
(1)        000                          .ENDC
(1)
013054                                  MEMINC  IMM,2                   ;LOAD NAK TYPE
(1)     000433                          MICPC=MICPC+1
(1)  013054  016402                     <MOVE!WRMEM!INCMAR!IMM!<2>>
(1)
013056                                  MEM     IMM,310                 ;LOAD SUB-TYPE NO BUFFERS
(1)     000434                          MICPC=MICPC+1
(1)  013056  002710                     <MOVE!WRMEM!IMM!<310>>
(1)
013060                                  LDMA    IMM,NTLS
(1)     000435                          MICPC=MICPC+1
(1)        001                          .IF IDN IMM,IMM
(1)  013060  010012                     <MOVE!LDMAR!IMM!<NTLS&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<NTLS>>
(1)        000                          .ENDC
```



```
(1)
013062                                  ALWAYS  RH5                     ;BRANCH TO SEND NAK ROUTINE
(1)     000436                          MICPC=MICPC+1
(1)  013062  104552                     <JUMP!ALCOND!<RH5-INIT&3000*4>!<RH5-INIT&777/2>>
(1)
013064                          RB3:    BRWRTE  IMM,4                   ;MASK FOR NO BUFFER AVAILABLE
(1)     000437                          MICPC=MICPC+1
(1)  013064  000404                     <MOVE!WRTEBR!IMM!<4>>
(1)
013066                                  SP      BR,AORB,SP1             ;SET THE FLAG
(1)     000440                          MICPC=MICPC+1
(1)  013066  063301                     <MOVE!SPX!BR!AORB!SP1>
(1)
013070                          RB1:    STATE   RCVC
(1)     000441                          MICPC=MICPC+1
(1)  013070  000461                     <MOVE!WRTEBR!IMM!<RCVC-INIT&777/2>>
013072                          RB0:    SP      BR,SELB,SP3
(1)     000442                          MICPC=MICPC+1
(1)  013072  063223                     <MOVE!SPX!BR!SELB!SP3>
(1)
013074                                  OUTPUT  <MEMX!INCMAR>,<SELB!OBA1>       ;OUTPUT LOW ORDER BYTE OF ADDRESS
(1)     000443                          MICPC=MICPC+1
(1)  013074  056226                     <MOVE!WROUT!MEMX!INCMAR!<SELB!OBA1>>
(1)
013076                                  OUTPUT  MEMX!INCMAR,<SELB!OBA2> ;OUTPUT HIGH BYTE OF ADDRESS
(1)     000444                          MICPC=MICPC+1
(1)  013076  056227                     <MOVE!WROUT!MEMX!INCMAR!<SELB!OBA2>>
(1)
013100                                  SP      IBUS,UBBR,SP0           ;READ THE BUS REQ REGISTER
(1)     000445                          MICPC=MICPC+1
(1)  013100  123220                     <MOVE!SPX!IBUS!UBBR!SP0>
(1)
013102                                  BRWRTE  IMM,101                 ;MASK OFF ALL BUT NXM AND VEC4 BITS
(1)     000446                          MICPC=MICPC+1
(1)  013102  000501                     <MOVE!WRTEBR!IMM!<101>>
(1)
013104                                  SP      BR,AANDB,SP0            ;AND SAVE IN SP0
(1)     000447                          MICPC=MICPC+1
(1)  013104  063260                     <MOVE!SPX!BR!AANDB!SP0>
(1)
013106                                  SP      IMM,300,SP5             ;MASK TO ISOLATE EX, MEM BITS
(1)     000450                          MICPC=MICPC+1
(1)  013106  003305                     <MOVE!SPX!IMM!300!SP5>
(1)
                                                                        ;NOTE THIS REALLY WRITES A 305 BUT THE
                                                                        ;5 GETS SHIFTED OUT
013110                                  BRWRTF  MEMX,AANDB!SP5          ;MASK ALL BUT EX, MEM BITS
(1)     000451                          MICPC=MICPC+1
(1)  013110  040665                     <MOVE!WRTFBR!MEMX!<AANDB!SP5>>
(1)
013112                                  BRSHFT                          ;SHIFT THEM INTO THE CORRECT POSITION
(1)     000452                          MICPC=MICPC+1
(1)  013112  001620                     <MOVE!SHFTBR!WRTFBR!SFLG>
(1)
013114                                  BRSHFT
(1)     000453                          MICPC=MICPC+1
```

```
(1)  013114  001620                        <MOVE!SHFTBR!WRTEBR!SELB>
(1)
952  013116                                BRSHFT
(1)          000454                        MICPC=MICPC+1
(1)  013116  001620                        <MOVE!SHFTBR!WRTEBR!SELB>
(1)
953  013120                                BRSHFT
(1)          000455                        MICPC=MICPC+1
(1)  013120  001620                        <MOVE!SHFTBR!WRTEBR!SELB>
(1)
954  013122                                OUT     BR,AORB!OBR                     ;WRITE EX MEM BITS OUT
(1)          000456                        MICPC=MICPC+1
(1)  013122  061311                        <MOVE!WROUTX!BR!<AORB!OBR>>
(1)
955  013124                                ALWAYS  IDLE
(1)          000457                        MICPC=MICPC+1
(1)  013124  100451                        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)
956  013126                           RB2:  ALWAYS  I2
(1)          000460                        MICPC=MICPC+1
(1)  013126  100456                        <JUMP!ALCOND!<I2-INIT&3000*4>!<I2-INIT&777/2>>
(1)
```



```
958                                       .SBTTL  RCVC--ROUTINE TO HANDLE SECOND CHARACTER OF COUNT FIELD, SELECT AND FINA
959                                       ;ENTERED FROM IDLE LOOP
960                                       ;INTERPRETS SELECT AND FINAL
961                                       ;CHECKS FOR COUNT TOO LARGE
962                                       ;
963  013130                           RCVC:
964              001                      .IF DF  $LOW
965                                       ALWAYS  SELQSY                  ;"CALL" SELECT/QSYNC SUBROUTINE
966              000                      .ENDC
967              001                      .IF NDF $LOW
968  013130                               SP      IBUS,RCVDAT,SP5         ;GET CHARACTER
(1)          000461                       MICPC=MICPC+1
(1)  013130  023205                       <MOVE!SPX!IBUS!RCVDAT!SP5>
(1)
969  013132                               BRWRTE  IMM,200                 ;SEPARATE SELECT BIT FROM COUNT
(1)          000462                       MICPC=MICPC+1
(1)  013132  000600                       <MOVE!WRTEBR!IMM!<200>>
(1)
970  013134                               BRWRTE  BR,AANDB!SP5
(1)          000463                       MICPC=MICPC+1
(1)  013134  060665                       <MOVE!WRTEBR!BR!<AANDB!SP5>>
(1)
971  013136                               SP      BR,AORB,SP10
(1)          000464                       MICPC=MICPC+1
(1)  013136  063310                       <MOVE!SPX!BR!AORB!SP10>
(1)
972  013140                               LDMA    IMM,BC                  ;LOAD MAR TO BYTE COUNT
(1)          000465                       MICPC=MICPC+1
(1)              002                      .IF IDN IMM,IMM
(1)  013140  010167                       <MOVE!LDMAR!IMM!<BC&377>>
(1)                                       .IFF
(1)                                       <MOVE!LDMAR!IMM!<BC>>
(1)              001                      .ENDC
(1)
973  013142                               MEMINC  BR,SELA!SP4             ;SAVE LOW BYTE
(1)          000466                       MICPC=MICPC+1
(1)  013142  076604                       <MOVE!WRMEM!INCMAR!BR!<SELA!SP4>>
(1)
974  013144                               MEMINC  BR,SELA!SP5             ;AND NOW HIGH BYTE
(1)          000467                       MICPC=MICPC+1
(1)  013144  076605                       <MOVE!WRMEM!INCMAR!BR!<SELA!SP5>>
(1)
975              000                      .ENDC
976  013146                           RC5:  STATE   RCVD                    ;SET NEXT STATE TO D
(1)          000470                       MICPC=MICPC+1
(1)  013146  000472                       <MOVE!WRTEBR!IMM!<RCVD-INIT&777/2>>
977  013150                               ALWAYS  REXIT
(1)          000471                       MICPC=MICPC+1
(1)  013150  100450                       <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
(1)
```

```
                                        .SBTTL  RCVD--ROUTINE TO HANDLE RESPONSE FIELD FOR NUMBERED MESSAGES
                                        ;
013152                          RCVD:   STATE   RCVE
        000472                          MICPC=MICPC+1
013152  000513                          <MOVE!WRTEBR!IMM!<RCVE-INIT&777/2>>
013154                          RD2:    SP      BR,SELB,SP3             ;SAVE THE STATE
        000473                          MICPC=MICPC+1
013154  063223                          <MOVE!SPX!BR!SELB!SP3>
013156                                  SPBR    IBUS,RCVDAT,SP0         ;INPUT THE CHARACTER
        000474                          MICPC=MICPC+1
013156  023600                          <MOVE!SPBRX!IBUS!RCVDAT!SP0>
013160                                  BRWRTE  BR,SUB!SP17             ;COMPARE NEW R TO LAST R
        000475                          MICPC=MICPC+1
013160  060757                          <MOVE!WRTEBR!BR!<SUB!SP17>>
013162                                  BR7     10$                     ;IF NEW IS GREATER---PROCESS
        000476                          MICPC=MICPC+1
013162  107500                          <JUMP!BR7CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
013164                                  ALWAYS  IDLE
        000477                          MICPC=MICPC+1
013164  100451                          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
013166                          10$:    BRWRTE  BR,SELA!SP1             ;READ STATUS BYTE
        000500                          MICPC=MICPC+1
013166  060601                          <MOVE!WRTEBR!BR!<SELA!SP1>>
013170                                  BR7     IDLE                    ;MAINT. MODE = GET OUT
        000501                          MICPC=MICPC+1
013170  103451                          <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
013172                                  BRWRTE  BR,SELA!SP10
        000502                          MICPC=MICPC+1
013172  060610                          <MOVE!WRTEBR!BR!<SELA!SP10>>
013174                                  BRSHFT
        000503                          MICPC=MICPC+1
013174  001620                          <MOVE!SHFTBR!WRTEBR!SELB>
013176                                  BR4     IDLE
        000504                          MICPC=MICPC+1
013176  103051                          <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
013200                                  LDMA    IMM,ISP17               ;ADDRESS LAST ACKED IMAGE
        000505                          MICPC=MICPC+1
           001                          .IF IDN IMM,IMM
013200  010153                          <MOVE!LDMAR!IMM!<ISP17&377>>
                                        .IFF
                                        <MOVE!LDMAR!IMM!<ISP17>>
000                                     .ENDC
013202                                  MEM     BR,SELA!SP0             ;COPY THE CHAR
        000506                          MICPC=MICPC+1
013202  062600                          <MOVE!WRMEM!BR!<SELA!SP0>>
```



```
013204                          RD5:    BRWRTE  IMM!LDMAR,REPST         ;SET UP COUNT FOR TIMER
        000507                          MICPC=MICPC+1
013204  010403                          <MOVE!WRTEBR!IMM!LDMAR!<REPST>>
                                                                        ;****DEPENDENT ON REPST = 2
013206                                  MEM     IMM,1                   ;RESET REP THRESHOLD
        000510                          MICPC=MICPC+1
013206  002401                          <MOVE!WRMEM!IMM!<1>>
013210                                  SP      BR,SELB,SP15            ;RESET THE COUNT
        000511                          MICPC=MICPC+1
013210  063235                          <MOVE!SPX!BR!SELB!SP15>
013212                                  ALWAYS  IDLE
        000512                          MICPC=MICPC+1
013212  100451                          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
1000                                        .SBTTL  RCVE--ROUTINE TO HANDLE N FIELD OF NUMBERED MESSAGE
1001                                        ;
1002  013214                        RCVE:   BRWRTE  BR,SELA!SP1             ;READ THE STATUS BYTE
 (1)          000513                        MICPC=MICPC+1
 (1)  013214  060601                        <MOVE!WRTEBR!BR!<SELA!SP1>>
 (1)
1003  013216                                BR7     RCVQ
 (1)          000514                        MICPC=MICPC+1
 (1)  013216  107703                        <JUMP!BR7CON!<RCVQ-INIT&3000*4>!<RCVQ-INIT&777/2>>
 (1)
1004  013220                                BRWRTE  IBUS,RCVDAT             ;INPUT THE CHARACTER
 (1)          000515                        MICPC=MICPC+1
 (1)  013220  020600                        <MOVE!WRTEBR!IBUS!<RCVDAT>>
 (1)
1005  013222                                CMP     BR,SP11
 (1)          000516                        MICPC=MICPC+1
 (1)  013222  060371                        <SUBTC!BR!SP11>
 (1)
1006  013224                                Z       5$
 (1)          000517                        MICPC=MICPC+1
 (1)  013224  105522                        <JUMP!ZCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
 (1)
1007  013226                                SP      BR,DECA,SP13            ;FORCE MSG TYPE TO -1
 (1)          000520                        MICPC=MICPC+1
 (1)  013226  063173                        <MOVE!SPX!BR!DECA!SP13>
 (1)
1008  013230                                ALWAYS  RE2
 (1)          000521                        MICPC=MICPC+1
 (1)  013230  104523                        <JUMP!ALCOND!<RE2-INIT&3000*4>!<RE2-INIT&777/2>>
 (1)
1009  013232                        5$:     SP      BR,INCA,SP11            ;UPDATE R FIELD
 (1)          000522                        MICPC=MICPC+1
 (1)  013232  063071                        <MOVE!SPX!BR!INCA!SP11>
 (1)
1010  013234                        RE2:    STATE   RCVF                    ;NEXT RECEIVE STATE IS F
 (1)          000523                        MICPC=MICPC+1
 (1)  013234  000525                        <MOVE!WRTEBR!IMM!<RCVF-INIT&777/2>>
1011  013236                                ALWAYS  REXIT
 (1)          000524                        MICPC=MICPC+1
 (1)  013236  100450                        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
 (1)
```



```
1013                                        .SBTTL  RCVF--ROUTINE TO IGNORE ADDRESS
1014  013240                        RCVF:   SP      BR,DECA,SP4             ;DECREMENTLOW BYTE OF COUNT
 (1)          000525                        MICPC=MICPC+1
 (1)  013240  063164                        <MOVE!SPX!BR!DECA!SP4>
 (1)
1015  013242                                C       RCVF0                   ;NO OVERFLOW
 (1)          000526                        MICPC=MICPC+1
 (1)  013242  105130                        <JUMP!CCOND!<RCVF0-INIT&3000*4>!<RCVF0-INIT&777/2>>
 (1)
1016  013244                                SP      BR,DECA,SP5             ;OVERFLOW = DECREMENT HIGH BYTE
 (1)          000527                        MICPC=MICPC+1
 (1)  013244  063165                        <MOVE!SPX!BR!DECA!SP5>
 (1)
1017  013246                        RCVF0:  STATE   RCVG
 (1)          000530                        MICPC=MICPC+1
 (1)  013246  000533                        <MOVE!WRTEBR!IMM!<RCVG-INIT&777/2>>
1018  013250                        RCVF1:  NOP     IBUS,RCVDAT,0           ;INPUT CHARACTER - AND DISCARD
 (1)          000531                        MICPC=MICPC+1
 (1)  013250  020200                        <IBUS!RCVDAT!0>
 (1)
1019  013252                                ALWAYS  REXIT
 (1)          000532                        MICPC=MICPC+1
 (1)  013252  100450                        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
 (1)
1020                                        ;
1021                                        .SBTTL  RCVG--ROUTINE TO IGNORE CRC1
1022                                        ;
1023  013254                        RCVG:   STATE   RCVH                    ;NEXT STATE IS RCVH
 (1)          000533                        MICPC=MICPC+1
 (1)  013254  000535                        <MOVE!WRTEBR!IMM!<RCVH-INIT&777/2>>
1024  013256                                ALWAYS  RCVF1
 (1)          000534                        MICPC=MICPC+1
 (1)  013256  104531                        <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
 (1)
```

```
1026                                    ,SBTTL  RCVH--ROUTINE TO HANDLE CRC2 AND TO DISPATCH NUMBERED AND UNNUMBERED TYP
1027                                    ;
1028  013260                    RCVH:
1029  013260                            SP      IBUS,RCVDAT,SP0                 ;GET CHAR IN SP0
 (1)          000535                    MICPC=MICPC+1
 (1)  013260  023200                    <MOVE!SPX!IBUS!RCVDAT!SP0>
 (1)
1030  013262                            BRWRTE  IBUS,RCVCON                     ;READ RECVR CONTROL REGISTER
 (1)          000536                    MICPC=MICPC+1
 (1)  013262  020640                    <MOVE!WRTEBR!IBUS!<RCVCON>>
 (1)
1031  013264                            BR0     TDON1                           ;IF BCC MATCH SET CRC IS GOOD
 (1)          000537                    MICPC=MICPC+1
 (1)  013264  116165                    <JUMP!BR0CON!<TDON1-INIT&3000*4>!<TDON1-INIT&777/2>>
 (1)
1032  013266                            BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
 (1)          000540                    MICPC=MICPC+1
 (1)  013266  060601                    <MOVE!WRTEBR!BR!<SELA!SP1>>
 (1)
1033  013270                            BR7     RHX                             ;MAINT MODE
 (1)          000541                    MICPC=MICPC+1
 (1)  013270  107740                    <JUMP!BR7CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
 (1)
1034  013272                            BRWRTE  DP,<SELA!SP10>          ;READ PORT STATUS WORD TO BR
 (1)          000542                    MICPC=MICPC+1
 (1)  013272  060610                    <MOVE!WRTEBR!DP!<SELA!SP10>>
 (1)
1035  013274                            BRSHFT
 (1)          000543                    MICPC=MICPC+1
 (1)  013274  001620                    <MOVE!SHFTBR!WRTEBR!SELB>
 (1)
1036  013276                            BR4     SNAK1                   ;IF START MODE--PROCEED TO RESEND START
 (1)          000544                    MICPC=MICPC+1
 (1)  013276  117307                    <JUMP!BR4CON!<SNAK1-INIT&3000*4>!<SNAK1-INIT&777/2>>
 (1)
1037  013300                            LDMA    IMM,T                           ;ELSE BCC ERROR--LOAD ADDRESS OF TYPE FI
 (1)          000545                    MICPC=MICPC+1
 (1)             001                    .IF IDN IMM,IMM
 (1)  013300  010151                    <MOVE!LDMAR!IMM!<T&377>>
 (1)                                    .IFF
 (1)                                    <MOVE!LDMAR!IMM!<T>>
 (1)             000                    .ENDC
 (1)
1038  013302                            MEMINC  IMM,2                           ;WRITE NAK TYPE
 (1)          000546                    MICPC=MICPC+1
 (1)  013302  016402                    <MOVE!WRMEM!INCMAR!IMM!<2>>
 (1)
1039  013304                            MEMINC  IMM,301                         ;WRITE HEADER BCC ERROR SUBTYPE
 (1)          000547                    MICPC=MICPC+1
 (1)  013304  016701                    <MOVE!WRMEM!INCMAR!IMM!<301>>
 (1)
1040  013306                            MEM     BR,SELA!SP17            ;RESTORE LAST ACKED  IMAGE
 (1)          000550                    MICPC=MICPC+1
 (1)  013306  062617                    <MOVE!WRMEM!BR!<SELA!SP17>>
 (1)
1041  013310                            LDMA    IMM,NHDS                ;ADDRESS CUM ERROR COUNTER
```



```
 (1)          000551                    MICPC=MICPC+1
 (1)             001                    .IF IDN IMM,IMM
 (1)  013310  010013                    <MOVE!LDMAR!IMM!<NHDS&377>>
 (1)                                    .IFF
 (1)                                    <MOVE!LDMAR!IMM!<NHDS>>
 (1)             000                    .ENDC
 (1)
1042  013312                    RH5:    SP      MEMX,SELB,SP0           ;WRITE IT TO SP0
 (1)          000552                    MICPC=MICPC+1
 (1)  013312  043220                    <MOVE!SPX!MEMX!SELB!SP0>
 (1)
1043  013314                            MEM     BR,INCA!SP0             ;INCREMENT IT
 (1)          000553                    MICPC=MICPC+1
 (1)  013314  062460                    <MOVE!WRMEM!BR!<INCA!SP0>>
 (1)
1044  013316                            LDMA    IMM,NAKST               ;ADDRESS NAKS TMTED DYNAMIC
 (1)          000554                    MICPC=MICPC+1
 (1)             001                    .IF IDN IMM,IMM
 (1)  013316  010001                    <MOVE!LDMAR!IMM!<NAKST&377>>
 (1)                                    .IFF
 (1)                                    <MOVE!LDMAR!IMM!<NAKST>>
 (1)             000                    .ENDC
 (1)
1045  013320                            BRWRTE  MEMX,SELB               ;WRITE IT TO BR
 (1)          000555                    MICPC=MICPC+1
 (1)  013320  040620                    <MOVE!WRTEBR!MEMX!<SELB>>
 (1)
1046  013322                            BSHFTB                          ;SHIFT IT RIGHT
 (1)          000556                    MICPC=MICPC+1
 (1)  013322  061620                    <MOVE!SHFTBR!SELB!BR>
 (1)
1047  013324                            MEM     BR,SELB                 ;UPDATE IT
 (1)          000557                    MICPC=MICPC+1
 (1)  013324  062620                    <MOVE!WRMEM!BR!<SELB>>
 (1)
1048  013326                            BR0     NTHRES                  ;BRANCH IF THRESHOLD EXCEEDED
 (1)          000560                    MICPC=MICPC+1
 (1)  013326  116256                    <JUMP!BR0CON!<NTHRES-INIT&3000*4>!<NTHRES-INIT&777/2>>
 (1)
1049  013330                            ALWAYS  SNAK
 (1)          000561                    MICPC=MICPC+1
 (1)  013330  114704                    <JUMP!ALCOND!<SNAK-INIT&3000*4>!<SNAK-INIT&777/2>>
 (1)
1050  013332                    RH3:    BRWRTE  DP,<DECA!SP13>                  ;LOAD TYPE RECEIVED--DECREMENTING
 (1)          000562                    MICPC=MICPC+1
 (1)  013332  060573                    <MOVE!WRTEBR!DP!<DECA!SP13>>
 (1)
1051  013334                            Z       RH1                             ;IF ALUOUT IS ALL ONES IS NUMBERED MSG
 (1)          000563                    MICPC=MICPC+1
 (1)  013334  115447                    <JUMP!ZCOND!<RH1-INIT&3000*4>!<RH1-INIT&777/2>>
 (1)
1052  013336                            RSTATE  RCVA
 (1)          000564                    MICPC=MICPC+1
 (1)  013336  000[illegible]             <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
 (1)          000565                    MICPC=MICPC+1
 (1)  013340  063223                    <MOVE!SPX!BR!SELB!SP3>
```

```
 1053  013342                         BRWRTE  DP,<SELA!SP10>                ;LOAD LINE STATUS WORD IN BR
  (1)          000566                 MICPC=MICPC+1
  (1)  013342  060610                 <MOVE!WRTEBR!DP!<SELA!SP10>>
  (1)
 1054              001                ,IF DF  $LOW
 1055                                 BR4     FLUSH1
 1056                           CG1:
 1057              000                ,ENDC
 1058              001                ,IF NDF $LOW
 1059  013344                         OUTPUT  IMM,<200!ORCVCO>
  (1)          000567                 MICPC=MICPC+1
  (1)  013344  002212                 <MOVE!WROUT!IMM!<200!ORCVCO>>
  (1)
 1060              000                ,ENDC
 1061  013346                         BRSHFT                                  ;SHIFT RIGHT
  (1)          000570                 MICPC=MICPC+1
  (1)  013346  001620                 <MOVE!SHFTBR!WRTEBR!SELB>
  (1)
 1062  013350                         BR4     10$
  (1)          000571                 MICPC=MICPC+1
  (1)  013350  107177                 <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
  (1)
 1063  013352                         LDMA    IMM,TYPTAB                      ;ADDRESS TYPE TABLE
  (1)          000572                 MICPC=MICPC+1
  (1)              001                ,IF IDN IMM,IMM
  (1)  013352  010162                 <MOVE!LDMAR!IMM!<TYPTAB&377>>
  (1)                                 ,IFF
  (1)                                 <MOVE!LDMAR!IMM!<TYPTAB>>
  (1)              000                ,ENDC
  (1)
 1064  013354                         CMP     <MEMX!INCMAR>,SP13
  (1)          000573                 MICPC=MICPC+1
  (1)  013354  054373                 <SUBTC!MEMX!INCMAR!SP13>
  (1)
 1065  013356                         Z       REP
  (1)          000574                 MICPC=MICPC+1
  (1)  013356  115411                 <JUMP!ZCOND!<REP-INIT&3000*4>!<REP-INIT&777/2>>
  (1)
 1066  013360                         CMP     <MEMX!INCMAR>,SP13
  (1)          000575                 MICPC=MICPC+1
  (1)  013360  054373                 <SUBTC!MEMX!INCMAR!SP13>
  (1)
 1067  013362                         Z       NAK
  (1)          000576                 MICPC=MICPC+1
  (1)  013362  115445                 <JUMP!ZCOND!<NAK-INIT&3000*4>!<NAK-INIT&777/2>>
  (1)
 1068  013364                   10$:  LDMA    IMM,TYPSTT              ;SET POINTER TO STAPT TYPE
  (1)          000577                 MICPC=MICPC+1
  (1)              001                ,IF IDN IMM,IMM
  (1)  013364  010164                 <MOVE!LDMAR!IMM!<TYPSTT&377>>
  (1)                                 ,IFF
  (1)                                 <MOVE!LDMAR!IMM!<TYPSTT>>
  (1)              000                ,ENDC
  (1)
 1069  013366                         CMP     <MEMX!INCMAR>,SP13
  (1)          000600                 MICPC=MICPC+1
```



```
  (1)  013366  054373                 <SUBTC!MEMX!INCMAR!SP13>
  (1)
 1070  013370                         Z       START
  (1)          000601                 MICPC=MICPC+1
  (1)  013370  115420                 <JUMP!ZCOND!<START-INIT&3000*4>!<START-INIT&777/2>>
  (1)
 1071                                                                         ;STACK TYPE
 1072  013372                         CMP     <MEMX!INCMAR>,SP13
  (1)          000602                 MICPC=MICPC+1
  (1)  013372  054373                 <SUBTC!MEMX!INCMAR!SP13>
  (1)
 1073  013374                         Z       STACK
  (1)          000603                 MICPC=MICPC+1
  (1)  013374  115432                 <JUMP!ZCOND!<STACK-INIT&3000*4>!<STACK-INIT&777/2>>
  (1)
 1074  013376                         CMP     <MEMX!INCMAR>,SP13      ;ACK TYPE
  (1)          000604                 MICPC=MICPC+1
  (1)  013376  054373                 <SUBTC!MEMX!INCMAR!SP13>
  (1)
 1075  013400                         Z       ACK
  (1)          000605                 MICPC=MICPC+1
  (1)  013400  101746                 <JUMP!ZCOND!<ACK-INIT&3000*4>!<ACK-INIT&777/2>>
  (1)
 1076  013402                         ALWAYS  IDLE                            ;OTHERWISE IGNORE--MUST BE OBS MSG
  (1)          000606                 MICPC=MICPC+1
  (1)  013402  100451                 <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
  (1)
 1077              001                ,IF DF  $LOW
 1078                           RCVCK: SPBR    IBUS,RCVCON,SP0                 ;READ RCVR CONTROL CSR
 1079                                  BRWRTE  BR,ADD!SP0                      ;SHIFT LEFT
 1080                                  BR7     I1
 1081                                  ALWAYS  TA1
 1082                           ACK:   BRWRTE  BR,AA!SP10                      ;READ LINE STATUS=SHIFTING LEFT
 1083                                  BR4     5$                              ;IF START RECD -- CLEAR START MODE
 1084                                  ALWAYS  IDLE
 1085                           5$:    BRWRTE  IMM,327                         ;CLEAR START MODE
 1086                                  SP      BR,AANDB,SP10                   ;IN LINE STATUS
 1087                                  ALWAYS  RD5
 1088              000                ,ENDC
```

```
1090                                    ;*****************TIME CRITICAL CODE-- CHANGE WITH GREAT CARE*********************
1091                                            .SBTTL  RCVKO1--ROUTINE TO HANDLE FIRST BYTE ODD RECEIVE
1092  013404                            RCVKO1: SPBR    IBUS,NPR,SP0                    ;READ NPR REGISTER
 (1)          000607                            MICPC=MICPC+1
 (1)  013404  123600                            <MOVE!SPBRX!IBUS!NPR!SP0>
 (1)
1093  013406                                    BR0     IDLE
 (1)          000610                            MICPC=MICPC+1
 (1)  013406  102051                            <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 (1)
1094  013410                                    BRWRTE  IMM,200                         ;MASK FOR C0(BYTE TRANSFER)
 (1)          000611                            MICPC=MICPC+1
 (1)  013410  000600                            <MOVE!WRTEBR!IMM!<200>>
 (1)
1095             001                            .IF NDF $LOW
1096  013412                                    SP      BR,AORB,SP0
 (1)          000612                            MICPC=MICPC+1
 (1)  013412  063300                            <MOVE!SPX!BR!AORB!SP0>
 (1)
1097             000                            .ENDC
1098             001                            .IF DF  $LOW
1099                                            OUT     BR,<AORB!ONPR>                  ;TURN ON C0
1100             000                            .ENDC
1101  013414                                    STATE   RKE1
 (1)          000613                            MICPC=MICPC+1
 (1)  013414  000653                            <MOVE!WRTEBR!IMM!<RKE1-INIT&777/2>>
1102  013416                                    ALWAYS  RCVKO2
 (1)          000614                            MICPC=MICPC+1
 (1)  013416  104620                            <JUMP!ALCOND!<RCVKO2-INIT&3000*4>!<RCVKO2-INIT&777/2>>
 (1)
1103
1104  013420                                    .SBTTL  RCVKO--PROCESS ODD CHARACTER
                                        RCVKO:  SPBR    IBUS,NPR,SP0                    ;IS AN NPR GOING
 (1)          000615                            MICPC=MICPC+1
 (1)  013420  123600                            <MOVE!SPBRX!IBUS!NPR!SP0>
 (1)
1105             001                            .IF NDF $LOW
1106  013422                                    BR0     RK66                            ;IF SO, REITERATE ODD AND EXIT
 (1)          000616                            MICPC=MICPC+1
 (1)  013422  106247                            <JUMP!BROCON!<RK66-INIT&3000*4>!<RK66-INIT&777/2>>
 (1)
1107             000                            .ENDC
1108             001                            .IF DF  $LOW
1109                                            BR0     IDLE                            ;IF SO, GO BACK TO IDLE LOOP
1110             000                            .ENDC
1111  013424                                    STATE   RCVKE
 (1)          000617                            MICPC=MICPC+1
 (1)  013424  000625                            <MOVE!WRTEBR!IMM!<RCVKE-INIT&777/2>>
1112  013426                            RCVKO2: SP      BR,SELB,SP3                     ;SET STATE
 (1)          000620                            MICPC=MICPC+1
 (1)  013426  063223                            <MOVE!SPX!BR!SELB!SP3>
 (1)
1113  013430                                    OUTPUT  IBUS,RCVDAT!OUTDA2              ;OUTPUT A CHAR
 (1)          000621                            MICPC=MICPC+1
 (1)  013430  022203                            <MOVE!WROUT!IBUS!<RCVDAT!OUTDA2>>
 (1)
1114  013432                            RK8:    BRWRTE  IMM,21                          ;SET OUT NPR (C1) AND NPR REQ
```



```
 (1)          000622                            MICPC=MICPC+1
 (1)  013432  000421                            <MOVE!WRTEBR!IMM!<21>>
 (1)
1115             001                            .IF DF  $LOW
1116                                            SP      IBUS,NPR,SP0                    ;READ NPR REGISTER
1117             000                            .ENDC
1118  013434                            RK7:    OUT     BR,<AORB!ONPR>                  ;WRITE NPR REGISTER
 (1)          000623                            MICPC=MICPC+1
 (1)  013434  061310                            <MOVE!WROUTX!BR!<AORB!ONPR>>
 (1)
1119  013436                                    ALWAYS  IDLE
 (1)          000624                            MICPC=MICPC+1
 (1)  013436  100451                            <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 (1)
```

```
                                        .SBTTL  RCVKE--HANDLE EVEN BYTES
013440                          RCVKE:  BRWRTE  IBUS,NPR                ;READ NPR CONTROL REGISTER
        000625                          MICPC=MICPC+1
013440  120600                          <MOVE!WRTEBR!IBUS!<NPR>>

           001                          .IF NDF $LOW
013442                                  BR4     RK4                     ;IF  RECV NPR--BRANCH
        000626                          MICPC=MICPC+1
013442  107251                          <JUMP!BR4CON!<RK4-INIT&3000*4>!<RK4-INIT&777/2>>

           000                          .ENDC
           001                          .IF DF  $LOW
                                        BR0     IDLE
           000                          .ENDC
013444                          RK5:    SP      IBUS,IOBA1,SP0          ;READ LOW BYTE OF BA TO SP
        000627                          MICPC=MICPC+1
013444  023140                          <MOVE!SPX!IBUS!IOBA1!SP0>

013446                                  OUTPUT  DP,<INCA!OBA1>          ;WRITE INCREMENTED BA
        000630                          MICPC=MICPC+1
013446  062066                          <MOVE!WROUT!DP!<INCA!OBA1>>

013450                          RK50:   SP      BR,DECA,SP4             ;DECREMENT CHARACTER COUNT
        000631                          MICPC=MICPC+1
013450  063164                          <MOVE!SPX!BR!DECA!SP4>

013452                                  C       10$                     ;NO OVERFLOW
        000632                          MICPC=MICPC+1
013452  105235                          <JUMP!CCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>

013454                                  SP      BR,DECA,SP5             ;OVERFLOW = DECREMENT HIGH BYTE
        000633                          MICPC=MICPC+1
013454  063165                          <MOVE!SPX!BR!DECA!SP5>

013456                                  Z       RL3                     ;BYTE COUNT ZERO
        000634                          MICPC=MICPC+1
013456  105711                          <JUMP!ZCOND!<RL3-INIT&3000*4>!<RL3-INIT&777/2>>

013460                          10$:    OUTPUT  IBUS,<RCVDAT!OUTDA1>    ;READ CHARACTER AND WRITE IT
        000635                          MICPC=MICPC+1
013460  022202                          <MOVE!WROUT!IBUS!<RCVDAT!OUTDA1>>

013462                                  SP      IBUS,IOBA1,SP0          ;READ INCREMENTED BA
        000636                          MICPC=MICPC+1
013462  023140                          <MOVE!SPX!IBUS!IOBA1!SP0>

013464                                  OUTPUT  DP,<INCA!OBA1>          ;WRITE INCREMENTED BA
        000637                          MICPC=MICPC+1
013464  062066                          <MOVE!WROUT!DP!<INCA!OBA1>>

013466                                  C       ICBA22                  ;IF CARRY INC BA HIGH
        000640                          MICPC=MICPC+1
013466  115035                          <JUMP!CCOND!<ICBA22-INIT&3000*4>!<ICBA22-INIT&777/2>>

013470                          RK3:    SP      BR,DECA,SP4             ;DECREMENT THE COUNT OF BYTES
        000641                          MICPC=MICPC+1
```



```
013470  063164                          <MOVE!SPX!BR!DECA!SP4>

013472                                  C       RK6                     ;NO OVERFLOW
        000642                          MICPC=MICPC+1
013472  105245                          <JUMP!CCOND!<RK6-INIT&3000*4>!<RK6-INIT&777/2>>

013474                                  SP      BR,DECA,SP5             ;DECREMENT HIGH BYTE OF COUNT
        000643                          MICPC=MICPC+1
013474  063165                          <MOVE!SPX!BR!DECA!SP5>

013476                                  Z       RL4                     ;BYTE COUNT ZERO
        000644                          MICPC=MICPC+1
013476  111772                          <JUMP!ZCOND!<RL4-INIT&3000*4>!<RL4-INIT&777/2>>

           001                          .IF NDF $LOW
013500                          RK6:    BRWRTE  IBUS,RCVCON             ;READ RECEIVER CONTROL REGISTER
        000645                          MICPC=MICPC+1
013500  020640                          <MOVE!WRTEBR!IBUS!<RCVCON>>

013502                                  BR4     RCVKO                   ;IF ANOTHER CHARACTER--PROCESS
        000646                          MICPC=MICPC+1
013502  107215                          <JUMP!BR4CON!<RCVKO-INIT&3000*4>!<RCVKO-INIT&777/2>>

013504                          RK66:   STATE   RCVKO
        000647                          MICPC=MICPC+1
013504  000615                          <MOVE!WRTEBR!IMM!<RCVKO-INIT&777/2>>
013506                                  ALWAYS  REXIT
        000650                          MICPC=MICPC+1
013506  100450                          <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>

013510                          RK4:    BR0     IDLE
        000651                          MICPC=MICPC+1
013510  102051                          <JUMP!BR0CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>

013512                                  ALWAYS  RK5                     ;IF NO NPR --PROCESS
        000652                          MICPC=MICPC+1
013512  104627                          <JUMP!ALCOND!<RK5-INIT&3000*4>!<RK5-INIT&777/2>>

           000                          .ENDC
           001                          .IF DF  $LOW
013514                          RK6:    STATE   RCVKO
                                        ALWAYS  REXIT
           000                          .ENDC
                                        ;
013514                          RKE1:   SP      IBUS,NPR,SP0            ;READ NPR REGISTER
        000653                          MICPC=MICPC+1
013514  123200                          <MOVE!SPX!IBUS!NPR!SP0>

           001                          .IF NDF $LOW
013516                                  BR0     IDLE                    ;NPR STILL IN PROGRESS
        000654                          MICPC=MICPC+1
013516  102051                          <JUMP!BR0CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>

           000                          .ENDC
013520                                  BRWRTE  IMM,177                 ;MASK FOR ALL BUT C0
        000655                          MICPC=MICPC+1
```

```
  (1)  013520  000577                        <MOVE!WRTEBR!IMM!<177>>
  (1)
1161   013522                                OUT     BR,<AANDB!ONPR>                 ;TURN OFF ALL BUT C0
  (1)          000656                        MICPC=MICPC+1
  (1)  013522  061270                        <MOVE!WROUTX!BR!<AANDB!ONPR>>
  (1)
1162   013524                                ALWAYS  RK50
  (1)          000657                        MICPC=MICPC+1
  (1)  013524  104631                        <JUMP!ALCOND!<RK50-INIT&3000*4>!<RK50-INIT&777/2>>
  (1)
1163                                    ;*****************END OF TIME CRITICAL PATH************************************
1164
1165   013526                           RCVKE0: SP      IBUS,RCVDAT,SP0          ;READ CHARACTER AND SAVE IN SP0
  (1)          000660                        MICPC=MICPC+1
  (1)  013526  023200                        <MOVE!SPX!IBUS!RCVDAT!SP0>
  (1)
1166   013530                                OUTPUT  BR,<SELA!OUTDA1>         ;SEND NONSENSE CHARACTER
  (1)          000661                        MICPC=MICPC+1
  (1)  013530  062202                        <MOVE!WROUT!BR!<SELA!OUTDA1>>
  (1)
1167   013532                                BRWRTE  BR,SELA!SP1              ;READ STATUS BYTE
  (1)          000662                        MICPC=MICPC+1
  (1)  013532  060601                        <MOVE!WRTEBR!BR!<SELA!SP1>>
  (1)
1168   013534                                BR7     PASWRD                   ;MAINT MODE - SEE IF RLD MESSAGE
  (1)          000663                        MICPC=MICPC+1
  (1)  013534  117576                        <JUMP!BR7CON!<PASWRD-INIT&3000*4>!<PASWRD-INIT&777/2>>
  (1)
1169   013536                                ALWAYS  RK3                      ;OTHERWISE PROCESS NORMALLY
  (1)          000664                        MICPC=MICPC+1
  (1)  013536  104641                        <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
  (1)
```



```
1171                                        .SBTTL  RCVI--STORE UNNUMBERED MESSAGE TYPE
1172   013540                           RCVI:   SP      IBUS,RCVDAT,SP13         ;STORE UNNUMBERED TYPE
  (1)          000665                        MICPC=MICPC+1
  (1)  013540  023213                        <MOVE!SPX!IBUS!RCVDAT!SP13>
  (1)
1173   013542                                STATE   RCVJ                     ;NEXT STATE IS J
  (1)          000666                        MICPC=MICPC+1
  (1)  013542  000670                        <MOVE!WRTEBR!IMM!<RCVJ-INIT&777/2>>
1174   013544                                ALWAYS  REXIT
  (1)          000667                        MICPC=MICPC+1
  (1)  013544  100450                        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
  (1)
1175                                        ;
```

```
1177                                    .SBTTL  RCVJ--ROUTINE TO HANDLE SUBTYPE FIELD,SELECT AND FINAL
1178  013546                            RCVJ:
1179              001                   .IF DF  $LOW
1180                                    ALWAYS  SELQSY          ;"CALL" SELECT AND QSYNC SUBROUTINE
1181              000                   .ENDC
1182              001                   .IF NDF $LOW
1183  013546                            SP      IBUS,RCVDAT,SP5         ;GET CHARACTER
 (1)          000670                   MICPC=MICPC+1
 (1)  013546  023205                   <MOVE!SPX!IBUS!RCVDAT!SP5>
 (1)
1184  013550                            BRWRTE  IMM,200                 ;CONDITIONALLY SET BIT
 (1)          000671                   MICPC=MICPC+1
 (1)  013550  000600                   <MOVE!WRTEBR!IMM!<200>>
 (1)
1185  013552                            BRWRTE  BR,AANDB!SP5
 (1)          000672                   MICPC=MICPC+1
 (1)  013552  060665                   <MOVE!WRTEBR!BR!<AANDB!SP5>>
 (1)
1186  013554                            SP      BR,AORB,SP10
 (1)          000673                   MICPC=MICPC+1
 (1)  013554  063310                   <MOVE!SPX!BR!AORB!SP10>
 (1)
1187              000                   .ENDC
1188  013556                            STATE   RCVR                    ;NEXT STATE IS N
 (1)          000674                   MICPC=MICPC+1
 (1)  013556  000676                   <MOVE!WRTEBR!IMM!<RCVR-INIT&777/2>>
1189  013560                            ALWAYS  REXIT
 (1)          000675                   MICPC=MICPC+1
 (1)  013560  100450                   <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
 (1)
```



```
1191                                    .SBTTL  RCVR--UNNUMBERED MESSAGE RESPONSE FIELD
1192                                    ;ENTERED FROM IDLE LOOP
1193                                    ;
1194  013562                            RCVR:   BRWRTE  IMM,3                   ;REP MESSAGE TYPE TO BR
 (1)          000676                   MICPC=MICPC+1
 (1)  013562  000403                   <MOVE!WRTEBR!IMM!<3>>
 (1)
1195  013564                            NOP     BR,SUB,SP13             ;IS TYPE ACK OR NAK
 (1)          000677                   MICPC=MICPC+1
 (1)  013564  060353                   <BR!SUB!SP13>
 (1)
1196  013566                            STATE   RCVQ                    ;NEXT STATE IS RCVQ
 (1)          000700                   MICPC=MICPC+1
 (1)  013566  000703                   <MOVE!WRTEBR!IMM!<RCVQ-INIT&777/2>>
1197                                                                    ;***NOTE THIS INSTR DOES NOT CLOCK "C"
1198  013570                            C       RCVF1                   ;IF NOT IGNORE
 (1)          000701                   MICPC=MICPC+1
 (1)  013570  105131                   <JUMP!CCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
 (1)
1199  013572                            ALWAYS  RD2                     ;DO RANGE CHECKS
 (1)          000702                   MICPC=MICPC+1
 (1)  013572  104473                   <JUMP!ALCOND!<RD2-INIT&3000*4>!<RD2-INIT&777/2>>
 (1)
```

```
1201                                    ,SBTTL  RCVQ--UNNUMBERED MESSAGE--NUMBER FIELD
1202                                    ;ENTER  FROM IDLE
1203                                    ;
1204  013574                    RCVQ:   STATE   RCVF                    ;NEXT STATE IS ADDRESS
 (1)          000703                    MICPC=MICPC+1
 (1)  013574  000525                    <MOVE!WRTEBR!IMM!<RCVF-INIT&777/2>>
1205  013576                            ALWAYS  RCVF1
 (1)          000704                    MICPC=MICPC+1
 (1)  013576  104531                    <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
 (1)
```



```
1207                                    ,SBTTL  RCVL--PROCESS CRC3
1208                                    ;ENTERED FROM  IDLE LOOP
1209  013600                    RCVL:   SPBR    IBUS,NPR,SP0                    ;READ NPR CONTROL
 (1)          000705                    MICPC=MICPC+1
 (1)  013600  123600                    <MOVE!SPBRX!IBUS!NPR!SP0>
 (1)
1210             001                    ,IF NDF $LOW
1211  013602                            BR4     RL1                             ;RCV NPR BRANCH
 (1)          000706                    MICPC=MICPC+1
 (1)  013602  107314                    <JUMP!BR4CON!<RL1-INIT&3000*4>!<RL1-INIT&777/2>>
 (1)
1212             000                    ,ENDC
1213             001                    ,IF DF  $LOW
1214                                    BR0     IDLE
1215             000                    ,ENDC
1216  013604                    RL2:    BRWRTE  IMM,176                         ;MASK TO TURN OFF C0
 (1)          000707                    MICPC=MICPC+1
 (1)  013604  000576                    <MOVE!WRTEBR!IMM!<176>>
 (1)
1217  013606                            OUT     BR,AANDB!ONPR
 (1)          000710                    MICPC=MICPC+1
 (1)  013606  061270                    <MOVE!WROUTX!BR!<AANDB!ONPR>>
 (1)
1218                                    ;
1219  013610                    RL3:    NOP     IBUS,RCVDAT,0           ;INPUT CHARACTER AND DISCARD
 (1)          000711                    MICPC=MICPC+1
 (1)  013610  020200                    <IBUS!RCVDAT!0>
 (1)
1220  013612                            STATE   RCVM
 (1)          000712                    MICPC=MICPC+1
 (1)  013612  000716                    <MOVE!WRTEBR!IMM!<RCVM-INIT&777/2>>
1221  013614                            ALWAYS  REXIT
 (1)          000713                    MICPC=MICPC+1
 (1)  013614  100450                    <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
 (1)
1222                                    ;
1223             001                    ,IF NDF $LOW
1224  013616                    RL1:    BR0     IDLE                    ;NPR GOING --GET OUT
 (1)          000714                    MICPC=MICPC+1
 (1)  013616  102051                    <JUMP!BR0CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 (1)
1225  013620                            ALWAYS  RL2
 (1)          000715                    MICPC=MICPC+1
 (1)  013620  104707                    <JUMP!ALCOND!<RL2-INIT&3000*4>!<RL2-INIT&777/2>>
 (1)
1226             000                    ,ENDC
1227                                    ;
```

```
                                        ;SBTTL  RCVM--PROCESS CRC4--END OF DATA MESSAGE
                                        ;ENTERED FROM IDLE LOOP
                                        ;IF CRC CORRECT -- QUEUE INTERUPT AND UPDATE RESPONSE
                                        ;
                                        ;IF CRC WRONG SEND NAK
013622                          RCVM:   BRWRTE  IBUS,UBBR                       ;READ UNIBUS BR REGISTER
        000716                          MICPC=MICPC+1
013622  120620                          <MOVE!WRTEBR!IBUS!<UBBR>>

013624                                  BR0     NXMERR                          ;NON-EXISTANT MEMORY
        000717                          MICPC=MICPC+1
013624  106351                          <JUMP!BR0CON!<NXMERR-INIT&3000*4>!<NXMERR-INIT&777/2>>

013626                                  SP      IBUS,RCVDAT,SP0                 ;READ CRC CHARACTER
        000720                          MICPC=MICPC+1
013626  023200                          <MOVE!SPX!IBUS!RCVDAT!SP0>

013630                                  BRWRTE  IBUS,RCVCON                     ;READ RECEIVER CONTROL REGISTER
        000721                          MICPC=MICPC+1
013630  020640                          <MOVE!WRTEBR!IBUS!<RCVCON>>

013632                                  BR0     RCVM1                           ;IF CRC GOOD -- PROCESS
        000722                          MICPC=MICPC+1
013632  116214                          <JUMP!BR0CON!<RCVM1-INIT&3000*4>!<RCVM1-INIT&777/2>>

013634                                  BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
        000723                          MICPC=MICPC+1
013634  060601                          <MOVE!WRTEBR!BR!<SELA!SP1>>

013636                                  BR7     RHX                             ;CRC ERROR IN BOOT MODE - FLUSH
        000724                          MICPC=MICPC+1
013636  107740                          <JUMP!BR7CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>

013640                                  LDMA    IMM,T                           ;ELSE SEND NAK --DATA ERROR
        000725                          MICPC=MICPC+1
           001                          .IF IDN IMM,IMM
013640  010151                          <MOVE!LDMAR!IMM!<T&377>>
                                        .IFF
                                        <MOVE!LDMAR!IMM!<T>>
              000                       .ENDC

013642                                  MEMINC  IMM,2                           ;NAK TYPE
        000726                          MICPC=MICPC+1
013642  016102                          <MOVE!WRMEM!INCMAR!IMM!<2>>

013644                                  MEMINC  IMM,302                         ;DATA ERROR SUBTYPE
        000727                          MICPC=MICPC+1
013644  016702                          <MOVE!WRMEM!INCMAR!IMM!<302>>

013646                                  LDMA    IMM,NDATS
        000730                          MICPC=MICPC+1
           001                          .IF IDN IMM,IMM
013646  010014                          <MOVE!LDMAR!IMM!<NDATS&377>>
                                        .IFF
                                        <MOVE!LDMAR!IMM!<NDATS>>
              000                       .ENDC
```



```

013650                                  ALWAYS  RH5                             ;SEND NAK
        000731                          MICPC=MICPC+1
013650  104552                          <JUMP!ALCOND!<RH5-INIT&3000*4>!<RH5-INIT&777/2>>


013652                          RCVM0:  LDMA    IMM,<<RTHRS+3>>                 ;POINT TO ERROR WORD
        000732                          MICPC=MICPC+1
           001                          .IF IDN IMM,IMM
013652  010177                          <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
                                        .IFF
                                        <MOVE!LDMAR!IMM!<<RTHRS+3>>>
              000                       .ENDC

013654                                  BRWRTE  IMM,10                          ;MAINT MESSAGE ERROR
        000733                          MICPC=MICPC+1
013654  000410                          <MOVE!WRTEBR!IMM!<10>>

013656                                  ALWAYS RCEXY                            ;GIVE FATAL ERROR
        000734                          MICPC=MICPC+1
013656  114522                          <JUMP!ALCOND!<RCEXY-INIT&3000*4>!<RCEXY-INIT&777/2>>

```

```
1251                                    .SBTTL  EM2--PROCESS RLD MESSAGE
1252                                    ;ENTERED FROM IDLE LOOP
1253                                    ;IF RLD PASSWORD CHECKS TRIGGER THE BOOT ROM
1254
1255  013660                    EM2:    BRWRTE  IBUS,RCVDAT             ;READ THE CHAR
 (1)          000735                    MICPC=MICPC+1
 (1)  013660  020600                    <MOVE!WRTEBR!IBUS!<RCVDAT>>
 (1)
1256  013662                            CMP     BR,SP13                 ;IS IT A MATCH
 (1)          000736                    MICPC=MICPC+1
 (1)  013662  060373                    <SUBTC!BR!SP13>
 (1)
1257  013664                            Z       EM3
 (1)          000737                    MICPC=MICPC+1
 (1)  013664  105746                    <JUMP!ZCOND!<EM3-INIT&3000*4>!<EM3-INIT&777/2>>
 (1)
1258                                                                    ;FALL INTO RHX
1259  013666                    RHX:    BRWRTE  BR,AA!SP1               ;READ STATUS BYTE SHIFTED LEFT
 (1)          000740                    MICPC=MICPC+1
 (1)  013666  060521                    <MOVE!WRTEBR!BR!<AA!SP1>>
 (1)
1260  013670                            BR4     10$                     ;DLE RECEIVED IN NORMAL MODE
 (1)          000741                    MICPC=MICPC+1
 (1)  013670  107343                    <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
 (1)
1261  013672                            ALWAYS  FLUSH                   ;ALREADY IN MAINT MODE
 (1)          000742                    MICPC=MICPC+1
 (1)  013672  104415                    <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
 (1)
1262  013674                    10$:    BRWRTE  IMM,163                 ;MASK TO CLEAR ALL MAINT RELATED BITS
 (1)          000743                    MICPC=MICPC+1
 (1)  013674  000563                    <MOVE!WRTEBR!IMM!<163>>
 (1)
1263  013676                            SP      BR,AANDB,SP1            ;CLEAR THEM
 (1)          000744                    MICPC=MICPC+1
 (1)  013676  063261                    <MOVE!SPX!BR!AANDB!SP1>
 (1)
1264  013700                            ALWAYS  FLUSH
 (1)          000745                    MICPC=MICPC+1
 (1)  013700  104415                    <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
 (1)
1265
1266  013702                    EM3:    SP      BR,DECA,SP4             ;DECREMENT CHARACTER COUNT BY ONE
 (1)          000746                    MICPC=MICPC+1
 (1)  013702  063164                    <MOVE!SPX!BR!DECA!SP4>
 (1)
1267  013704                            Z       EMTRIG                  ;TRIGGER AC LOW
 (1)          000747                    MICPC=MICPC+1
 (1)  013704  115712                    <JUMP!ZCOND!<EMTRIG-INIT&3000*4>!<EMTRIG-INIT&777/2>>
 (1)
1268  013706                            ALWAYS  IDLE
 (1)          000750                    MICPC=MICPC+1
 (1)  013706  100451                    <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 (1)
```



```
1270                  001               .IF NDF $LOW
1271                                    .SBTTL NXMERR ---NON EXISTANT MEMORY HANDLER
1272  013710                    NXMERR: LDMA    IMM,<<RTHRS+3>>         ;ADDRESS ERROR LINK
 (1)          000751                    MICPC=MICPC+1
 (1)                002               .IF IDN IMM,IMM
 (1)  013710  010177                    <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
 (1)                                    .IFF
 (1)                                    <MOVE!LDMAR!IMM!<<RTHRS+3>>>
 (1)                001               .ENDC
 (1)
1273  013712                            MEMINC  IMM,1
 (1)          000752                    MICPC=MICPC+1
 (1)  013712  016401                    <MOVE!WRMEM!INCMAR!IMM!<1>>
 (1)
1274  013714                            MEM     IMM,0                   ;NXM ERROR BIT
 (1)          000753                    MICPC=MICPC+1
 (1)  013714  002400                    <MOVE!WRMEM!IMM!<0>>
 (1)
1275  013716                            SP      MEMX,SELB,SP10          ;CLEAR STATUS
 (1)          000754                    MICPC=MICPC+1
 (1)  013716  043230                    <MOVE!SPX!MEMX!SELB!SP10>
 (1)
1276  013720                            ALWAYS  RCEXX
 (1)          000755                    MICPC=MICPC+1
 (1)  013720  114524                    <JUMP!ALCOND!<RCEXX-INIT&3000*4>!<RCEXX-INIT&777/2>>
 (1)
```

```
                        000                     .ENDC
                        001                     .IF DF  $LOW
                                                .SBTTL  SELQSY--ROUTINETOCHECK SELECT AND QSYNC AND DIDDLE LINE STATUS WORD
                                                ;USES SP5, ALWAYS CALLED BY FIRST INSTR IN A RSTATE
                                        SELQSY: SPBR    IBUS,RCVDAT,SP5         ;READCHARACTERINTO  SP5 AND THE BR
                                                BR7     15$                     ;SELECT SET?--BRANCH
                                        5$:     BRWRTE  BR,AA!SP5               ;SHIFTBR LEFT
                                                BR7     20$                     ;FINAL SET?
                                        10$:    BRWRTE  IMM,77                  ;MASK   TO BR
                                                SP      BR,AANDB,SP5            ;TURN OFF SELECTANDFINAL
                                                .ALWAY  BR,INCA,SP3!PAGE1
                                                ;
                                        15$:    BRWRTE  IMM,200                 ;SET OK TO SEND
                                                SP      BR,AORB,SP10            ;IN LINE STATUS WORD
                                                ALWAYS  5$
                                        20$:    BRWRTE  IMM,20                  ;SETCLEARACTIVE
                                                SP      BR,AORB,SP10            ;IN LINE STATUS WORD
                                                ALWAYS  10$
                                                .PAGE
                        000                     .ENDC
                                                ;
013722                                  BOOT:   BRWRTE  BR,SELA!SP1             ;SEE IF IN MAINT. MODE
(1)             000756                          MICPC=MICPC+1
(1)     013722  060601                          <MOVE!WRTEBR!BR!<SELA!SP1>>
(1)
013724                                          BR7     RA3                     ;BRANCH IF SO AND TREAT DLE LIKE NUM. MSG.
(1)             000757                          MICPC=MICPC+1
(1)     013724  103742                          <JUMP!BR7CON!<RA3-INIT&3000*4>!<RA3-INIT&777/2>>
(1)
013726                                          BRWRTE  IMM,210                 ;MASK TO SET MAINT MODE AND DLE RECV'D
(1)             000760                          MICPC=MICPC+1
(1)     013726  000610                          <MOVE!WRTEBR!IMM!<210>>
(1)
013730                                          SP      BR,AORB,SP1             ;SET THE BITS
(1)             000761                          MICPC=MICPC+1
(1)     013730  063301                          <MOVE!SPX!BR!AORB!SP1>
(1)
013732                                          ALWAYS  RA3                     ;TREAT LIKE NUMBERED MESSAGE
(1)             000762                          MICPC=MICPC+1
(1)     013732  100742                          <JUMP!ALCOND!<RA3-INIT&3000*4>!<RA3-INIT&777/2>>
(1)
013734                                  RESEXT: BRWRTE  IMM,4                   ;ADD TO NXT BITS
(1)             000763                          MICPC=MICPC+1
(1)     013734  000404                          <MOVE!WRTEBR!IMM!<4>>
(1)
013736                                          SP      BR,ADD,SP0
(1)             000764                          MICPC=MICPC+1
(1)     013736  063000                          <MOVE!SPX!BR!ADD!SP0>
(1)
013740                                          ALWAYS  TH3X
(1)             000765                          MICPC=MICPC+1
(1)     013740  110601                          <JUMP!ALCOND!<TH3X-INIT&3000*4>!<TH3X-INIT&777/2>>
(1)
013742                                  TABMXT: BRWRTE  IMM,4                   ;INCREMENT NXT
(1)             000766                          MICPC=MICPC+1
(1)     013742  000404                          <MOVE!WRTEBR!IMM!<4>>
```



```
(1)
013744                                          SP      IBUS,UBBR,SP0           ;READ BR CONTROL
(1)             000767                          MICPC=MICPC+1
(1)     013744  123220                          <MOVE!SPX!IBUS!UBBR!SP0>
(1)
013746                                          OUT     BR,ADD!OBR
(1)             000770                          MICPC=MICPC+1
(1)     013746  061011                          <MOVE!WROUTX!BR!<ADD!OBR>>
(1)
013750                                          ALWAYS  ECX
(1)             000771                          MICPC=MICPC+1
(1)     013750  114761                          <JUMP!ALCOND!<ECX-INIT&3000*4>!<ECX-INIT&777/2>>
(1)
                                                ;
013752                                  RTHRES: BRWRTE  IMM,2
(1)             000772                          MICPC=MICPC+1
(1)     013752  000402                          <MOVE!WRTEBR!IMM!<2>>
(1)
013754                                          ALWAYS  ERRXX
(1)             000773                          MICPC=MICPC+1
(1)     013754  114663                          <JUMP!ALCOND!<ERRXX-INIT&3000*4>!<ERRXX-INIT&777/2>>
(1)
                        000004                  .REPT   4
                                                $ZERO
                                                .ENDR
(1)     013756                                  $ZERO
(2)             000774                          MICPC=MICPC+1
(2)     013756  000000                          000000
(2)
(1)     013760                                  $ZERO
(2)             000775                          MICPC=MICPC+1
(2)     013760  000000                          000000
(2)
(1)     013762                                  $ZERO
(2)             000776                          MICPC=MICPC+1
(2)     013762  000000                          000000
(2)
(1)     013764                                  $ZERO
(2)             000777                          MICPC=MICPC+1
(2)     013764  000000                          000000
(2)
```

```
1318          013766                        .=INIT+2000
1319          000777                        MICPC=777
1320                                        .SBTTL  TMTDA--TRANSMITTER DISPATCH ROUTINE
1321                                        ;
1322  013766                        TMTDA:  BRWRTE  IBUS,TMTCON                     ;READ TRANMITTER CONTROL REGISTER
 (1)          001000                        MICPC=MICPC+1
 (1)  013766  020620                        <MOVE!WRTEBR!IBUS!<TMTCON>>
 (1)
1323  013770                                .BR4    DP,SELA,<2!PAGE2>               ;IF READY PROCEED
 (1)          001001                        MICPC=MICPC+1
 (1)  013770  173202                        <JUMP! BR4CON!DP!SELA!2!PAGE2>
 (1)
1324  013772                                ALWAYS  I1                              ;ELSE IDLE
 (1)          001002                        MICPC=MICPC+1
 (1)  013772  100454                        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
```



```
1326                                        .SBTTL  TMTA--FIRST CHARACTER OF HEADER
1327                                        ;
1328  013774                        TMTA:
1329             001                        .IF DF  $LOW
1330                                        BRWRTE  BR,AA!SP10                      ;SHIFT LEFT
1331                                        BP7     RCVCK
1332                                  TA1:
1333             000                        .ENDC
1334  013774                                BRWRTE  BR,SELA!SP10                    ;REPEAD STATUS
 (1)          001003                        MICPC=MICPC+1
 (1)  013774  060610                        <MOVE!WRTEBR!BR!<SELA!SP10>>
 (1)
1335  013776                                BP0     NUMSYN                          ;IF UNNUMBPENDING  -- SEND IT
 (1)          001004                        MICPC=MICPC+1
 (1)  013776  112007                        <JUMP!BROCON!<NUMSYN-INIT&3000*4>!<NUMSYN-INIT&777/2>>
 (1)
1336  014000                                BRSHFT
 (1)          001005                        MICPC=MICPC+1
 (1)  014000  001620                        <MOVE!SHFTBR!WRTEBR!SELB>
 (1)
1337  014002                                BR4     IDLE0                           ;IF START MODE--EXIT
 (1)          001006                        MICPC=MICPC+1
 (1)  014002  103063                        <JUMP!BR4CON!<IDLE0-INIT&3000*4>!<IDLE0-INIT&777/2>>
 (1)
1338             001                        .IF DF  $LOW
1339                                        BP1     NUMSYN                  ;IF LINE HAS GONE IDLE SEND SYN
1340                                                                        ;ELSE--START TO SEND MESSAGE
1341             000                        .ENDC
1342             001                        .IF NDF $LOW
1343  014004                        NUMSYN: BRWRTE  BR,<SELA!SP10>                  ;READ LINE STATUS WORD
 (1)          001007                        MICPC=MICPC+1
 (1)  014004  060610                        <MOVE!WRTEBR!BR!<SELA!SP10>>
 (1)
1344  014006                                BR7     5$                              ;IF OK TO SEND--PROCEED
 (1)          001010                        MICPC=MICPC+1
 (1)  014006  113412                        <JUMP!BR7CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
 (1)
1345  014010                                ALWAYS  I1                              ;ELSE--IDLE
 (1)          001011                        MICPC=MICPC+1
 (1)  014010  100454                        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1346  014012                          5$:   BRWRTE  IBUS,MODEM                      ;ARE WE STILL SENDING?
 (1)          001012                        MICPC=MICPC+1
 (1)  014012  020660                        <MOVE!WRTEBR!IBUS!<MODEM>>
 (1)
1347  014014                                BRSHFT
 (1)          001013                        MICPC=MICPC+1
 (1)  014014  001620                        <MOVE!SHFTBR!WRTEBR!SELB>
 (1)
1348  014016                                BR4     I1                              ;RTS SET? IF SO WE ARE--STALL
 (1)          001014                        MICPC=MICPC+1
 (1)  014016  103051                        <JUMP!BR4CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1349  014020                                BRWRTE  IMM,373                 ;ASK TO TURN OFFLINE IDLE
 (1)          001015                        MICPC=MICPC+1
 (1)  014020  000373                        <MOVE!WRTEBR!IMM!<373>>
```

```
 (1)
1350  014022                          SP      BR,AANDB,SP10           ;IN LINE STATUS WORD
 (1)          001016                  MICPC=MICPC+1
 (1)  014022  063270                  <MOVE!SPX!BR!AANDB!SP10>
 (1)
1351  014024                          TSTATE  TMTA1
 (1)          001017                  MICPC=MICPC+1
 (1)  014024  000424                  <MOVE!WRTEBR!IMM!<TMTA1-INIT&777/2>>
 (1)          001020                  MICPC=MICPC+1
 (1)  014026  063222                  <MOVE!SPX!BR!SELB!SP2>
1352  014030                          BRWRTE  IMM,12
 (1)          001021                  MICPC=MICPC+1
 (1)  014030  000412                  <MOVE!WRTEBR!IMM!<12>>
 (1)
1353  014032                          SP      BR,SELB,SP6                     ;STORE IN SP6
 (1)          001022                  MICPC=MICPC+1
 (1)  014032  063226                  <MOVE!SPX!BR!SELB!SP6>
 (1)
1354  014034                          ALWAYS  I1                              ;BACK TO IDLE LOOP
 (1)          001023                  MICPC=MICPC+1
 (1)  014034  100454                  <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1355  014036                    TMTA1:  SP      BR,DECA,SP6                     ;DECREMENT SYN COUNT
 (1)          001024                  MICPC=MICPC+1
 (1)  014036  063166                  <MOVE!SPX!BR!DECA!SP6>
 (1)
1356  014040                          Z       TMTEXT
 (1)          001025                  MICPC=MICPC+1
 (1)  014040  111432                  <JUMP!ZCOND!<TMTEXT-INIT&3000*4>!<TMTEXT-INIT&777/2>>
 (1)
1357  014042                          OUTPUT  IMM,<1!OTMTCO>                  ;WRITE SOH TO TMTR CONTRL
 (1)          001026                  MICPC=MICPC+1
 (1)  014042  002011                  <MOVE!WROUT!IMM!<1!OTMTCO>>
 (1)
1358  014044                          BRWRTE  IMM,226                         ;SYNC CHAR
 (1)          001027                  MICPC=MICPC+1
 (1)  014044  000626                  <MOVE!WRTEBR!IMM!<226>>
 (1)
1359  014046                          OUTPUT  BR,<SELB!TMTDAT>                ;SEND THE CHARACTER
 (1)          001030                  MICPC=MICPC+1
 (1)  014046  062230                  <MOVE!WROUT!BR!<SELB!TMTDAT>>
 (1)
1360  014050                          ALWAYS  I1
 (1)          001031                  MICPC=MICPC+1
 (1)  014050  100454                  <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1361                                  .ENDC
1362  014052                    TMTEXT: BRWRTE  BR,<SELA!SP10>                  ;UNNUMB MESSGE?
 (1)          001032                  MICPC=MICPC+1
 (1)  014052  060610                  <MOVE!WRTEBR!BR!<SELA!SP10>>
 (1)
1363  014054                          BRC     TMTUN                           ;IF SO --BRANCH
 (1)          001033                  MICPC=MICPC+1
 (1)  014054  112043                  <JUMP!BRCCON!<TMTUN-INIT&3000*4>!<TMTUN-INIT&777/2>>
 (1)
1364  014056                          TSTATE  TMTB
```



```
 (1)          001034                  MICPC=MICPC+1
 (1)  014056  000451                  <MOVE!WRTEBR!IMM!<TMTB-INIT&777/2>>
 (1)          001035                  MICPC=MICPC+1
 (1)  014060  063222                  <MOVE!SPX!BR!SELB!SP2>
1365  014062                          BRWRTE  BR,SELA!SP1                     ;ARE WE IN BOOT MODE
 (1)          001036                  MICPC=MICPC+1
 (1)  014062  060601                  <MOVE!WRTEBR!BR!<SELA!SP1>>
 (1)
1366  014064                          BR7     TMTBT                   ;IF SO SEND DLE
 (1)          001037                  MICPC=MICPC+1
 (1)  014064  113447                  <JUMP!BR7CON!<TMTBT-INIT&3000*4>!<TMTBT-INIT&777/2>>
 (1)
1367  014066                          BRWRTE  IMM,201                         ;ELSE STORE SOH
 (1)          001040                  MICPC=MICPC+1
 (1)  014066  000601                  <MOVE!WRTEBR!IMM!<201>>
 (1)
1368  014070                    TMTA5:  OUTPUT  BR,<SELB!TMTDAT>                ;IN TMT SILO
 (1)          001041                  MICPC=MICPC+1
 (1)  014070  062230                  <MOVE!WROUT!BR!<SELB!TMTDAT>>
 (1)
1369  014072                          ALWAYS  I1
 (1)          001042                  MICPC=MICPC+1
 (1)  014072  100454                  <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1370  014074                    TMTUN:  TSTATE  TMTI
 (1)          001043                  MICPC=MICPC+1
 (1)  014074  000610                  <MOVE!WRTEBR!IMM!<TMTI-INIT&777/2>>
 (1)          001044                  MICPC=MICPC+1
 (1)  014076  063222                  <MOVE!SPX!BR!SELB!SP2>
1371  014100                          BRWRTE  IMM,5                           ;ENQ TO BR
 (1)          001045                  MICPC=MICPC+1
 (1)  014100  000405                  <MOVE!WRTEBR!IMM!<5>>
 (1)
1372  014102                          ALWAYS  TMTA5
 (1)          001046                  MICPC=MICPC+1
 (1)  014102  110441                  <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
 (1)
1373  014104                    TMTBT:  BRWRTE  IMM,220                 ;WRITE A DLE TO BR
 (1)          001047                  MICPC=MICPC+1
 (1)  014104  000620                  <MOVE!WRTEBR!IMM!<220>>
 (1)
1374  014106                          ALWAYS  TMTA5                   ;SEND IT
 (1)          001050                  MICPC=MICPC+1
 (1)  014106  110441                  <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
 (1)
1375             001                  .IF DF  $LOW
1376                                  ;
1377                                  NUMSYN: BRWRTE  BR,<SELA!SP10>                  ;READ LINE STATUS WORD
1378                                          BR7     5$                              ;IF OK TO SEND--PROCEED
1379                                          ALWAYS  I1                              ;ELSE--IDLE
1380                                  5$:     BRWRTE  IBUS,MODEM              ;ARE WE STILL SENDING?
1381                                          BRSHFT
```

```
1383                                    BR4     I1                      ;RTS SET? IF SO WE ARE--STALL
1384                                    BRWRTE  IMM,373                 ;MASK TO TURN OFFLINE IDLE
1385                                    SP      BR,AANDB,SP10           ;IN LINE STATUS WORD
1386                                    TSTATE  TMTA1
1387                                    BRWRTE  IMM,10
```



```
1388                                    SP      BR,SELB,SP6             ;STORE IN SP6
1390                            TMTA1:  SP      BR,DECA,SP6             ;DECREMENT SYN COUNT
1391                                    Z       TMTFXT
1392                                    OUTPUT  IMM,<1!OTMTCO>          ;WRITE SOM TO TMTR CONTRL
1393                                    BRWRTE  IMM,226                 ;SYNC CHAR
1394                                    ALWAYS  TMTA5
1395                    000             .ENDC
```

```
1397                                        .SBTTL  TMTB--OUTPUT FIRST CHAR OF COUNT
1398                                        ;
1399  014110                        TMTB:   LDMA    BR,SELA!SP16                    ;GETPOINTER TO NEXT TMT LINK
 (1)          001051                        MICPC=MICPC+1
 (1)             001                        .IF IDN BR,TMM
 (1)                                        <MOVE!LDMAR!IMM!<SELA!SP16&377>>
 (1)                                        .IFF
 (1)  014110  070216                        <MOVE!LDMAR!BR!<SELA!SP16>>
 (1)             000                        .ENDC
 (1)
1400  014112                                MEMINC  IMM,3                           ;WRITE MSG TMTED TO FLAGS
 (1)          001052                        MICPC=MICPC+1
 (1)  014112  016403                        <MOVE!WRMEM!INCMAR!IMM!<3>>
 (1)
1401  014114                                MEMINC  BR,SELA!SP12                    ;PICK UP MSGNO
 (1)          001053                        MICPC=MICPC+1
 (1)  014114  076612                        <MOVE!WRMEM!INCMAR!BR!<SELA!SP12>>
 (1)
1402  014116                                STATE   TMTC                            ;ADDRESS TMTR STATE
 (1)          001054                        MICPC=MICPC+1
 (1)  014116  000476                        <MOVE!WRTEBR!IMM!<TMTC-INIT&777/2>>
1403  014120                        TB0:    SP      BR,SELB,SP2                     ;UPDATE IT
 (1)          001055                        MICPC=MICPC+1
 (1)  014120  063222                        <MOVE!SPX!BR!SELB!SP2>
 (1)
1404  014122                                OUTPUT  <MEMX!INCMAR>,SELB!IBA1         ;WRITELOWBYTEOFADDRESS
 (1)          001056                        MICPC=MICPC+1
 (1)  014122  056224                        <MOVE!WROUT!MEMX!INCMAR!<SELB!IBA1>>
 (1)
1405  014124                                OUTPUT  <MEMX!INCMAR>,SELB!IBA2         ;WRITE HIGH BYTE OF ADDRESS
 (1)          001057                        MICPC=MICPC+1
 (1)  014124  056225                        <MOVE!WROUT!MEMX!INCMAR!<SELB!IBA2>>
 (1)
1406  014126                                SP      MEMX,SELB,SP7   ;HIGH BYTE OF COUNT TO SP7
 (1)          001060                        MICPC=MICPC+1
 (1)  014126  043227                        <MOVE!SPX!MEMX!SELB!SP7>
 (1)
1407                                                                ;WAIT TO MASK OFF MEM EXT. BITS
1408  014130                                SP      IBUS,NPR,SP0
 (1)          001061                        MICPC=MICPC+1
 (1)  014130  123200                        <MOVE!SPX!IBUS!NPR!SP0>
 (1)
1409  014132                                BRWRTE  IMM,220
 (1)          001062                        MICPC=MICPC+1
 (1)  014132  000620                        <MOVE!WRTEBR!IMM!<220>>
 (1)
1410  014134                                SP      BR,AANDB,SP0
 (1)          001063                        MICPC=MICPC+1
 (1)  014134  063260                        <MOVE!SPX!BR!AANDB!SP0>
 (1)
1411  014136                                SP      IMM,300,SP6                     ;MASK FOR MXT
 (1)          001064                        MICPC=MICPC+1
 (1)  014136  003306                        <MOVE!SPX!IMM!300!SP6>
 (1)
1412  014140                                BRWRTE  MEMX!INCMAR,AANDB!SP6           ;TURN OFF CC2
 (1)          001065                        MICPC=MICPC+1
```



```
 (1)  014140  054666                        <MOVE!WRTEBR!MEMX!INCMAR!<AANDB!SP6>>
 (1)
1413  014142                                OUTPUT  MEMX,SELB!TMTDAT                ;ALSO WRITE COUNT TO TMTR SILO
 (1)          001066                        MICPC=MICPC+1
 (1)  014142  042230                        <MOVE!WROUT!MEMX!<SELB!TMTDAT>>
 (1)
1414  014144                                BRSHFT                                  ;SHIFT BITS INTO CORRECT POSITION
 (1)          001067                        MICPC=MICPC+1
 (1)  014144  001620                        <MOVE!SHFTBR!WRTEBR!SELB>
 (1)
1415  014146                                BRSHFT
 (1)          001070                        MICPC=MICPC+1
 (1)  014146  001620                        <MOVE!SHFTBR!WRTEBR!SELB>
 (1)
1416  014150                                BRSHFT
 (1)          001071                        MICPC=MICPC+1
 (1)  014150  001620                        <MOVE!SHFTBR!WRTEBR!SELB>
 (1)
1417  014152                                BRSHFT
 (1)          001072                        MICPC=MICPC+1
 (1)  014152  001620                        <MOVE!SHFTBR!WRTEBR!SELB>
 (1)
1418  014154                                OUT     BR,AORB!ONPR
 (1)          001073                        MICPC=MICPC+1
 (1)  014154  061310                        <MOVE!WROUTX!BR!<AORB!ONPR>>
 (1)
1419  014156                                SPBR    MEMX,SELB,SP6           ;LOWBYTE OF COUNT TO SP6
 (1)          001074                        MICPC=MICPC+1
 (1)  014156  043626                        <MOVE!SPBRX!MEMX!SELB!SP6>
 (1)
1420             001                        .IF DF  $LOW    ;*****10/21/76
1421                                        ALWAYS  IDLE
1422             000                        .ENDC
1423             001                        .IF NDF $LOW    ;*****10/21/76
1424  014160                                ALWAYS  I1
 (1)          001075                        MICPC=MICPC+1
 (1)  014160  100454                        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1425             000                        .ENDC
1426                                        ;
```

```
1428                                    .SBTTL  TMTC--OUTPUT SECOND CHAR OF COUNT
1429                                    ;
1430  014162                    TMTC: BRWRTE  IMM,77                          ;MASK TO CLEAR MXT BITS
 (1)          001076                 MICPC=MICPC+1
 (1)  014162  000477                 <MOVE!WRTEBR!IMM!<77>>
 (1)
1431  014164                          SPBR    BR,AANDB,SP7                    ;CLEAR THEM
 (1)          001077                 MICPC=MICPC+1
 (1)  014164  063667                 <MOVE!SPBRX!BR!AANDB!SP7>
 (1)
1432  014166                          OUTPUT  DP,<SELB!TMTDAT>                ;WRITE TO TMT SILO
 (1)          001100                 MICPC=MICPC+1
 (1)  014166  062230                 <MOVE!WROUT!DP!<SELB!TMTDAT>>
 (1)
1433  014170                          BRWRTE  IMM,TML8                                ;GET WRAPAROUND ADDRESS
 (1)          001101                 MICPC=MICPC+1
 (1)  014170  000543                 <MOVE!WRTEBR!IMM!<TML8>>
 (1)
1434  014172                          CMP     BR,SP16                         ;WRAPAORUND
 (1)          001102                 MICPC=MICPC+1
 (1)  014172  060376                 <SUBTC!BR!SP16>
 (1)
1435  014174                          Z       10$
 (1)          001103                 MICPC=MICPC+1
 (1)  014174  111511                 <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
 (1)
1436  014176                          BRWRTE  IMM,6                           ;OFFSET TO NEXT LINK
 (1)          001104                 MICPC=MICPC+1
 (1)  014176  000406                 <MOVE!WRTEBR!IMM!<6>>
 (1)
1437  014200                          SP      BR,ADD,SP16                     ;UPDATE THE POINTER
 (1)          001105                 MICPC=MICPC+1
 (1)  014200  063016                 <MOVE!SPX!BR!ADD!SP16>
 (1)
1438  014202                    5$:
1439             001                  .IF DF  $LOW
1440                                   STATE   TMTD
1441                                   ALWAYS  XEXIT
1442             000                  .ENDC
1443             001                  .IF NDF $LOW
1444  014202                           TSTATE  TMTD
 (1)          001106                 MICPC=MICPC+1
 (1)  014202  000514                 <MOVE!WRTEBR!IMM!<TMTD-INIT&777/2>>
 (1)          001107                 MICPC=MICPC+1
 (1)  014204  063227                 <MOVE!SPX!BR!SELB!SP2>
1445  014206                          ALWAYS  I1              ;****OCTOBER 29, 1976
 (1)          001110                 MICPC=MICPC+1
 (1)  014206  100454                 <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1446             000                  .ENDC
1447  014210                    10$:  BRWRTE  IMM,TML1                        ;GO BACK TO FIRST LINK
 (1)          001111                 MICPC=MICPC+1
 (1)  014210  000471                 <MOVE!WRTEBR!IMM!<TML1>>
 (1)
1448  014212                          SP      BR,SELB,SP16
 (1)          001112                 MICPC=MICPC+1
```



```
 (1)  014212  063236                 <MOVE!SPX!BR!SELB!SP16>
 (1)
1449  014214                          ALWAYS  5$
 (1)          001113                 MICPC=MICPC+1
 (1)  014214  110506                 <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
 (1)
1450                                    ;
```

```
1452                                    .SBTTL  TMTD--RESPONSE FIELD-NUMBERED MESSAGE
1453  014216                            TMTD:   STATE   TMTE
 (1)          001114                            MICPC=MICPC+1
 (1)  014216  000524                            <MOVE!WRTEBR!IMM!<TMTE-INIT&777/2>>
1454  014220                                    SP      BR,DECA,SP6                     ;ADJUSRT COUNT FOR  TWO'S  COMPLEMENT
 (1)          001115                            MICPC=MICPC+1
 (1)  014220  063166                            <MOVE!SPX!BR!DECA!SP6>
 (1)
1455  014222                                    C       TD2                             ;NO OVERFLOW
 (1)          001116                            MICPC=MICPC+1
 (1)  014222  111120                            <JUMP!CCOND!<TD2-INIT&3000*4>!<TD2-INIT&777/2>>
 (1)
1456  014224                                    SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF COUNT
 (1)          001117                            MICPC=MICPC+1
 (1)  014224  063167                            <MOVE!SPX!BR!DECA!SP7>
 (1)
1457  014226                            TD2:    LDMA    IMM,ISP11                       ;RESP FIELD ADDR TO MAR
 (1)          001120                            MICPC=MICPC+1
 (1)            001                             .IF IDN IMM,IMM
 (1)  014226  010171                            <MOVE!LDMAR!IMM!<ISP11&377>>
 (1)                                            .IFF
 (1)                                            <MOVE!LDMAP!IMM!<ISP11>>
 (1)              000                           .ENDC
 (1)
1458  014230                            TD3:    OUTPUT  MEMX,SELB!TMTDAT                ;WRITE IT TO SILO
 (1)          001121                            MICPC=MICPC+1
 (1)  014230  042230                            <MOVE!WROUT!MEMX!<SELB!TMTDAT>>
 (1)
1459  014232                            XEXIT2: SP      BR,SELB,SP2
 (1)          001122                            MICPC=MICPC+1
 (1)  014232  063222                            <MOVE!SPX!BR!SELB!SP2>
 (1)
1460  014234                                    ALWAYS  I1
 (1)          001123                            MICPC=MICPC+1
 (1)  014234  100454                            <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
```



```
1462                                    .SBTTL  TMTE--NUMBER FIELD--NUMBERED MESSAGE
1463  014236                            TMTE:
1464  014236                                    SPBR    IBUS,NPR,SP0            ;READ NPR CONTROL REGISTER
 (1)          001124                            MICPC=MICPC+1
 (1)  014236  123600                            <MOVE!SPBRX!IBUS!NPR!SP0>
 (1)
1465  014240                                    BR0     I1                      ;BUSY - GET OUT
 (1)          001125                            MICPC=MICPC+1
 (1)  014240  102054                            <JUMP!BR0CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1466  014242                                    BRWRTE  BR,SELA!SP12
 (1)          001126                            MICPC=MICPC+1
 (1)  014242  060612                            <MOVE!WRTEBR!BR!<SELA!SP12>>
 (1)
1467  014244                                    OUTPUT  BR,<SELB!TMTDAT>                ;WRITE IT TO THE SILO
 (1)          001127                            MICPC=MICPC+1
 (1)  014244  062230                            <MOVE!WROUT!BR!<SELB!TMTDAT>>
 (1)
1468  014246                                    STATE   TMTF
 (1)          001130                            MICPC=MICPC+1
 (1)  014246  000532                            <MOVE!WRTEBR!IMM!<TMTF-INIT&777/2>>
1469  014250                                    ALWAYS  TH3
 (1)          001131                            MICPC=MICPC+1
 (1)  014250  110600                            <JUMP!ALCOND!<TH3-INIT&3000*4>!<TH3-INIT&777/2>>
 (1)
```

```
1471                                    .SBTTL  TMTF--NUMBERED MSG ADDRESS FIELD
1472                                    ;
1473  014252                    TMTF1:  STATE   TF1
 (1)          001132                    MICPC=MICPC+1
 (1)  014252  000537                    <MOVE!WRTEBR!IMM!<TF1-INIT&777/2>>
1474  014254                    TF2:    SP      BR,SELB,SP2
 (1)          001133                    MICPC=MICPC+1
 (1)  014254  063222                    <MOVE!SPX!BR!SELB!SP2>
 (1)
1475  014256                            BRWRTE  IMM,1                   ;LOAD ADDRESS
 (1)          001134                    MICPC=MICPC+1
 (1)  014256  000401                    <MOVE!WRTEBR!IMM!<1>>
 (1)
1476             001                    .IF NDF $LOW
1477  014260                    TF3:    OUTPUT  BR,<SELB!TMTDAT>
 (1)          001135                    MICPC=MICPC+1
 (1)  014260  062230                    <MOVE!WROUT!BR!<SELB!TMTDAT>>
 (1)
1478  014262                            ALWAYS  I1
 (1)          001136                    MICPC=MICPC+1
 (1)  014262  100454                    <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1479             000                    .ENDC
1480             001                    .IF DF  $LOW
1481                                    ALWAYS  TMT45
1482             000                    .ENDC
```



```
1484                                    .SBTTL  TF1-NUMBERED MSG HEADER EOM
1485  014264                    TF1:    BRWRTE  IMM,2                   ;EOM  MASK TO BR
 (1)          001137                    MICPC=MICPC+1
 (1)  014264  000402                    <MOVE!WRTEBR!IMM!<2>>
 (1)
1486  014266                            OUTPUT  BR,<SELB!OTMTCO>        ;UPDATE TMTP CONTROL REGISTER
 (1)          001140                    MICPC=MICPC+1
 (1)  014266  062231                    <MOVE!WROUT!BR!<SELB!OTMTCO>>
 (1)
1487  014270                            OUTPUT  BR,<SELB!TMTDAT>        ;OUTPUT A GARBAGE CHAR
 (1)          001141                    MICPC=MICPC+1
 (1)  014270  062230                    <MOVE!WROUT!BR!<SELB!TMTDAT>>
 (1)
1488  014272                            BRWRTE  IBUS,IIBA1              ;READ LOW ORDER FROM INBA
 (1)          001142                    MICPC=MICPC+1
 (1)  014272  020500                    <MOVE!WRTEBR!IBUS!<IIBA1>>
 (1)
1489  014274                            BRO     TMTF1                   ;IF ODD BYTE--BRANCH
 (1)          001143                    MICPC=MICPC+1
 (1)  014274  112162                    <JUMP!BROCON!<TMTF1-INIT&3000*4>!<TMTF1-INIT&777/2>>
 (1)
1490  014276                            STATE   TMTH
 (1)          001144                    MICPC=MICPC+1
 (1)  014276  000546                    <MOVE!WRTEBR!IMM!<TMTH-INIT&777/2>>
1491  014300                            ALWAYS  XEXIT
 (1)          001145                    MICPC=MICPC+1
 (1)  014300  110563                    <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
 (1)
```

```
1493                                    ;************TIME CRITICAL PATH--MODIFY WITH GREAT CARE
1494                                            .SBTTL  TMTH--ROUTINE TO OUTPUT DATA CHARACTERS
1495                                            ;
1496  014302                            TMTH:   SPBR    IBUS,NPR,SP0                    ;READ NPR CONTROL
 (1)          001146                            MICPC=MICPC+1
 (1)  014302  123600                            <MOVE!SPBRX!IBUS!NPR!SP0>
 (1)
1497             001                            .IF NDF $LOW
1498  014304                                    BR4     5$                              ;IF RECV NPR --PROCESS
 (1)          001147                            MICPC=MICPC+1
 (1)  014304  113151                            <JUMP!BR4CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
 (1)
1499             000                            .ENDC
1500  014306                                    BR0     I1                              ;IF NPR IN PROGRESS --BRANCH
 (1)          001150                            MICPC=MICPC+1
 (1)  014306  102054                            <JUMP!BR0CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1501  014310                            5$:     OUTPUT  IBUS,<INDAT1!TMTDAT>            ;WRITE THE EVEN CHAR TO TMT SILO
 (1)          001151                            MICPC=MICPC+1
 (1)  014310  022010                            <MOVE!WROUT!IBUS!<INDAT1!TMTDAT>>
 (1)
1502  014312                                    SP      IBUS,IIBA1,SP0                  ;READ LOW BYTE OF BA TO SP
 (1)          001152                            MICPC=MICPC+1
 (1)  014312  023100                            <MOVE!SPX!IBUS!IIBA1!SP0>
 (1)
1503  014314                                    OUTPUT  BR,<INCA!IBA1>                  ;OUTPUT INCREMENTED BA
 (1)          001153                            MICPC=MICPC+1
 (1)  014314  062064                            <MOVE!WROUT!BR!<INCA!IBA1>>
 (1)
1504  014316                                    SP      BR,DECA,SP6                     ;DECREMENT CHARACTER  COUNT
 (1)          001154                            MICPC=MICPC+1
 (1)  014316  063166                            <MOVE!SPX!BR!DECA!SP6>
 (1)
1505  014320                                    C       TH6                             ;NO OVERFLOW
 (1)          001155                            MICPC=MICPC+1
 (1)  014320  111160                            <JUMP!CCOND!<TH6-INIT&3000*4>!<TH6-INIT&777/2>>
 (1)
1506  014322                                    SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF COUNT
 (1)          001156                            MICPC=MICPC+1
 (1)  014322  063167                            <MOVE!SPX!BR!DECA!SP7>
 (1)
1507  014324                                    Z       HEH1                            ;BYTE  COUNT ZERO
 (1)          001157                            MICPC=MICPC+1
 (1)  014324  115407                            <JUMP!ZCOND!<HEH1-INIT&3000*4>!<HEH1-INIT&777/2>>
 (1)
1508  014326                            TH6:
1509             001                            .IF NDF $LOW
1510  014326                                    BRWRTE  IBUS,TMTCON             ;READ TMTR CONTROL CSR
 (1)          001160                            MICPC=MICPC+1
 (1)  014326  020620                            <MOVE!WRTEBR!IBUS!<TMTCON>>
 (1)
1511  014330                                    BR4     TH9                             ;IF MORE ROOM IN SILO--BRANCH
 (1)          001161                            MICPC=MICPC+1
 (1)  014330  113165                            <JUMP!BR4CON!<TH9-INIT&3000*4>!<TH9-INIT&777/2>>
 (1)
1512             000                            .ENDC
```



```
1513  014332                            TMTF1:  STATE   TMTHO
 (1)          001162                            MICPC=MICPC+1
 (1)  014332  000565                            <MOVE!WRTEBR!IMM!<TMTHO-INIT&777/2>>
1514             001                            .IF DF  $LOW
1515                                            ALWAYS  XEXIT
1516             000                            .ENDC
1517             001                            .IF NDF $LOW
1518  014334                            XEXIT:  SP      BR,SELB,SP2                     ;STORE NEW TRANSMIT STATE
 (1)          001163                            MICPC=MICPC+1
 (1)  014334  063222                            <MOVE!SPX!BR!SELB!SP2>
 (1)
1519  014336                                    ALWAYS  I1
 (1)          001164                            MICPC=MICPC+1
 (1)  014336  100454                            <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1520             000                            .ENDC
1521  014340                            TMTHO:
1522             001                            .IF DF  $LOW
1523                                            SPBR    IBUS,NPR,SP0            ;NPR BUSY
1524                                            BR0     I1
1525             000                            .ENDC
1526  014340                            TH9:    OUTPUT  IBUS,<INDAT2!TMTDAT>            ;ODD CHAR TO SILO
 (1)          001165                            MICPC=MICPC+1
 (1)  014340  022030                            <MOVE!WROUT!IBUS!<INDAT2!TMTDAT>>
 (1)
1527  014342                                    SP      IBUS,IIBA1,SP0                  ;READ LOW BYTE TO BA
 (1)          001166                            MICPC=MICPC+1
 (1)  014342  023100                            <MOVE!SPX!IBUS!IIBA1!SP0>
 (1)
1528  014344                                    OUTPUT  BR,<INCA!IBA1>                  ;OUTPUT THE INCREMENTED BA
 (1)          001167                            MICPC=MICPC+1
 (1)  014344  062064                            <MOVE!WROUT!BR!<INCA!IBA1>>
 (1)
1529  014346                                    C       HOINCH
 (1)          001170                            MICPC=MICPC+1
 (1)  014346  111377                            <JUMP!CCOND!<HOINCH-INIT&3000*4>!<HOINCH-INIT&777/2>>
 (1)
1530  014350                            TH8:    SP      BR,DECA,SP6                     ;DECREMENT CHARACTERCOUNT
 (1)          001171                            MICPC=MICPC+1
 (1)  014350  063166                            <MOVE!SPX!BR!DECA!SP6>
 (1)
1531  014352                                    C       TH7                             ;NO OVERFLOW
 (1)          001172                            MICPC=MICPC+1
 (1)  014352  111175                            <JUMP!CCOND!<TH7-INIT&3000*4>!<TH7-INIT&777/2>>
 (1)
1532  014354                                    SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF  COUNT
 (1)          001173                            MICPC=MICPC+1
 (1)  014354  063167                            <MOVE!SPX!BR!DECA!SP7>
 (1)
1533  014356                                    Z       HEH1                            ;BYTE COUNT ZERO
 (1)          001174                            MICPC=MICPC+1
 (1)  014356  115407                            <JUMP!ZCOND!<HEH1-INIT&3000*4>!<HEH1-INIT&777/2>>
 (1)
1534  014360                            TH7:    SPBR    IBUS,NPR,SP0                    ;READ NPR REGISTER
 (1)          001175                            MICPC=MICPC+1
 (1)  014360  123600                            <MOVE!SPBRX!IBUS!NPR!SP0>
```

```
 (1)
1535            001                     .IF NDF $LOW
1536  014362                            BR0     TH2                             ;IF NPR BUSY WAIT TO GO
 (1)          001176                    MICPC=MICPC+1
 (1)  014362  112205                    <JUMP!BROCON!<TH2-INIT&3000*4>!<TH2-INIT&777/2>>
 (1)
1537            000                     .ENDC
1538  014364                            STATE   TMTH
 (1)          001177                    MICPC=MICPC+1
 (1)  014364  000546                    <MOVE!WRTEBR!IMM!<TMTH-INIT&777/2>>
1539  014366                      TH3:  SP      BR,SELB,SP2                     ;SAVE TSTATE
 (1)          001200                    MICPC=MICPC+1
 (1)  014366  063222                    <MOVE!SPX!BR!SELB!SP2>
 (1)
1540  014370                      TH3X: BRWRTE  IMM,156                         ;CLEAR C0 AND C1
 (1)          001201                    MICPC=MICPC+1
 (1)  014370  000556                    <MOVE!WRTEBR!IMM!<156>>
 (1)
1541  014372                            SP      BR,AANDB,SP0                    ;CLEAR THE BITS
 (1)          001202                    MICPC=MICPC+1
 (1)  014372  063260                    <MOVE!SPX!BR!AANDB!SP0>
 (1)
1542  014374                            OUT     BR,<INCA!ONPR>
 (1)          001203                    MICPC=MICPC+1
 (1)  014374  061070                    <MOVE!WROUTX!BR!<INCA!ONPR>>
 (1)
1543  014376                            ALWAYS  I1
 (1)          001204                    MICPC=MICPC+1
 (1)  014376  100454                    <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1544            001                     .IF NDF $LOW
1545  014400                      TH2:  TSTATE  TH7
 (1)          001205                    MICPC=MICPC+1
 (1)  014400  000575                    <MOVE!WRTEBR!IMM!<TH7-INIT&777/2>>
 (1)          001206                    MICPC=MICPC+1
 (1)  014402  063222                    <MOVE!SPX!BR!SELB!SP2>
1546  014404                            ALWAYS  I1
 (1)          001207                    MICPC=MICPC+1
 (1)  014404  100454                    <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1547            000                     .ENDC
1548                            ;***************END TIME CRITICAL PATH*****************************
1549                                    ;
```



```
1551                                    .SBTTL  TMTI--SEND UNNUMBERED TYPE FIELD
1552  014406                      TMTI: LDMA    IMM,T                           ;ADDRESS OF TYPE FIELD TO MAR
 (1)          001210                    MICPC=MICPC+1
 (1)             001                    .IF IDN IMM,IMM
 (1)  014406  010151                    <MOVE!LDMAR!IMM!<T&377>>
 (1)                                    .IFF
 (1)                                    <MOVE!LDMAR!IMM!<T>>
 (1)             000                    .ENDC
 (1)
1553  014410                            SP      MEMX,SELB,SP6                   ;COPY IT TO SP6
 (1)          001211                    MICPC=MICPC+1
 (1)  014410  043226                    <MOVE!SPX!MEMX!SELB!SP6>
 (1)
1554  014412                            STATE   TMTJ
 (1)          001212                    MICPC=MICPC+1
 (1)  014412  000614                    <MOVE!WRTEBR!IMM!<TMTJ-INIT&777/2>>
1555  014414                            ALWAYS  TD3
 (1)          001213                    MICPC=MICPC+1
 (1)  014414  110521                    <JUMP!ALCOND!<TD3-INIT&3000*4>!<TD3-INIT&777/2>>
 (1)
1556                                    ;
1557                                    .SBTTL  TMTJ--SEND SUB-TYPE FIELD
1558  014416                      TMTJ: LDMA    IMM,ST                          ;ADDRESS OF SUB-TYPE FIELD TO MAR
 (1)          001214                    MICPC=MICPC+1
 (1)             001                    .IF IDN IMM,IMM
 (1)  014416  010152                    <MOVE!LDMAR!IMM!<ST&377>>
 (1)                                    .IFF
 (1)                                    <MOVE!LDMAR!IMM!<ST>>
 (1)             000                    .ENDC
 (1)
1559  014420                            STATE   TMTK
 (1)          001215                    MICPC=MICPC+1
 (1)  014420  000617                    <MOVE!WRTEBR!IMM!<TMTK-INIT&777/2>>
1560  014422                            ALWAYS  TD3
 (1)          001216                    MICPC=MICPC+1
 (1)  014422  110521                    <JUMP!ALCOND!<TD3-INIT&3000*4>!<TD3-INIT&777/2>>
 (1)
```

```
1562                                    .SBTTL  TMTK--OUTPUT RESPONSE FIELD (UNNUMB MSG)
1563                                    ;
1564  014424                    TMTK:   BRWRTE  IMM,3                   ;WRITE A 3 TO BR
 (1)          001217                    MICPC=MICPC+1
 (1)  014424  000403                    <MOVE!WRTEBR!IMM!<3>>
 (1)
1565  014426                            NOP     BR,SUB,SP6              ;IF TYPE LESS THAN 3
 (1)          001220                    MICPC=MICPC+1
 (1)  014426  060346                    <BR!SUB!SP6>
 (1)
1566  014430                            TSTATE  TMTL
 (1)          001221                    MICPC=MICPC+1
 (1)  014430  000625                    <MOVE!WRTEBR!IMM!<TMTL-INIT&777/2>>
 (1)          001222                    MICPC=MICPC+1
 (1)  014432  063222                    <MOVE!SPX!BP!SELB!SP2>
1567  014434                            C       TMTL0
 (1)          001223                    MICPC=MICPC+1
 (1)  014434  111232                    <JUMP!CCOND!<TMTL0-INIT&3000*4>!<TMTL0-INIT&777/2>>
 (1)
1568  014436                            ALWAYS  TD2
 (1)          001224                    MICPC=MICPC+1
 (1)  014436  110520                    <JUMP!ALCOND!<TD2-INIT&3000*4>!<TD2-INIT&777/2>>
 (1)
1569                                    ;
1570                                    .SBTTL  TMTL--UNNUMB MSG NUMBER FIELD
1571  014440                    TMTL:   TSTATE  TMTM
 (1)          001225                    MICPC=MICPC+1
 (1)  014440  000637                    <MOVE!WRTEBR!IMM!<TMTM-INIT&777/2>>
 (1)          001226                    MICPC=MICPC+1
 (1)  014442  063222                    <MOVE!SPX!BR!SELB!SP2>
1572  014444                            BRWRTE  IMM,3
 (1)          001227                    MICPC=MICPC+1
 (1)  014444  000403                    <MOVE!WRTEBR!IMM!<3>>
 (1)
1573  014446                            CMP     BR,SP6                  ;IS MESSAGE REP
 (1)          001230                    MICPC=MICPC+1
 (1)  014446  060366                    <SUBTC!BR!SP6>
 (1)
1574  014450                            Z       TMTL1                   ;YES
 (1)          001231                    MICPC=MICPC+1
 (1)  014450  111635                    <JUMP!ZCOND!<TMTL1-INIT&3000*4>!<TMTL1-INIT&777/2>>
 (1)
1575  014452                    TMTL0:  BRWRTE  IMM,0                   ;ADDRESS CONTNAT OF ZERO
 (1)          001232                    MICPC=MICPC+1
 (1)  014452  000400                    <MOVE!WRTEBR!IMM!<0>>
 (1)
1576             001                    .IF DF  $LOW
1577                                    ALWAYS  TMTA5
1578             000                    .ENDC
1579             001                    .IF NDF $LOW
1580  014454                            OUTPUT  BR,<SELB!TMTDAT>                ;SEND IT OUT
 (1)          001233                    MICPC=MICPC+1
 (1)  014454  062230                    <MOVE!WROUT!BR!<SELB!TMTDAT>>
 (1)
1581  014456                            ALWAYS  I1                      ;BACK TO IDLE LOOP
 (1)          001234                    MICPC=MICPC+1
```



```
 (1)  014456  100454                    <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1582             000                    .ENDC
1583                                    ;
1584  014460                    TMTL1:  BRWRTE  BR,DECA!SP12                    ;WRITE A RESPONSE
 (1)          001235                    MICPC=MICPC+1
 (1)  014460  060572                    <MOVE!WRTEBR!BR!<DECA!SP12>>
 (1)
1585  014462                            ALWAYS  TMTA5
 (1)          001236                    MICPC=MICPC+1
 (1)  014462  110441                    <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
 (1)
1586                                    ;
```

```
1588                                    ,SBTTL  TMTM--UNNUMB MSG--STATION ADDRESS
1589  014464                            TMTM:   STATE   TNEOM
 (1)                 001237                    MICPC=MICPC+1
 (1)  014464  000641                           <MOVE!WRTEBR!IMM!<TNEOM-INIT&777/2>>
1590  014466                                    ALWAYS  TF2
 (1)                 001240                    MICPC=MICPC+1
 (1)  014466  110533                           <JUMP!ALCOND!<TF2-INIT&3000*4>!<TF2-INIT&777/2>>
 (1)
1591  014470                            TNEOM:  BRWRTE  IMM,2                           ;END OF MESSAGE TO BR
 (1)                 001241                    MICPC=MICPC+1
 (1)  014470  000402                           <MOVE!WRTEBR!IMM!<2>>
 (1)
1592  014472                                    OUTPUT  BR,<SELB!OTMTCO>
 (1)                 001242                    MICPC=MICPC+1
 (1)  014472  062231                           <MOVE!WROUT!BR!<SELB!OTMTCO>>
 (1)
1593  014474                                    OUTPUT  BR,<SELB!TMTDAT>                ;OUTPUT A GARBAGE CHARACTER
 (1)                 001243                    MICPC=MICPC+1
 (1)  014474  062230                           <MOVE!WROUT!BR!<SELB!TMTDAT>>
 (1)
1594  014476                                    BRWRTE  IMM,4                           ;SET UP LINE HAS GONE IDLE MASK
 (1)                 001244                    MICPC=MICPC+1
 (1)  014476  000404                           <MOVE!WRTEBR!IMM!<4>>
 (1)
1595  014500                                    SPBR    BR,AORB,SP10                    ;UPDATE LINE STATUS WORD
 (1)                 001245                    MICPC=MICPC+1
 (1)  014500  063710                           <MOVE!SPBRX!BR!AORB!SP10>
 (1)
1596  014502                                    BRWRTE  BR,AA!SP10                      ;SHIFT STATUS LEFT
 (1)                 001246                    MICPC=MICPC+1
 (1)  014502  060530                           <MOVE!WRTEBR!BR!<AA!SP10>>
 (1)
1597  014504                                    BR7     10$                             ;IF HDX SET---BRANCH TO CLEAR OK TO SEND
 (1)                 001247                    MICPC=MICPC+1
 (1)  014504  113653                           <JUMP!BR7CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
 (1)
1598  014506                                    BRWRTE  IMM,376                         ;MASK TO TURN OFF UNNUMB PENDDING
 (1)                 001250                    MICPC=MICPC+1
 (1)  014506  000776                           <MOVE!WRTEBR!IMM!<376>>
 (1)
1599  014510                            5$:     SP      BR,AANDB,SP10                   ;MASK TO LINE STATUS WORD
 (1)                 001251                    MICPC=MICPC+1
 (1)  014510  063270                           <MOVE!SPX!BR!AANDB!SP10>
 (1)
1600  014512                                    ALWAYS  TEOM2
 (1)                 001252                    MICPC=MICPC+1
 (1)  014512  110740                           <JUMP!ALCOND!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
 (1)
1601  014514                            10$:    BRWRTE  IMM,176                         ;CLEAR OK TO SEND AND UNNUMB PENPENDING
 (1)                 001253                    MICPC=MICPC+1
 (1)  014514  000576                           <MOVE!WRTEBR!IMM!<176>>
 (1)
1602  014516                                    ALWAYS  5$
 (1)                 001254                    MICPC=MICPC+1
 (1)  014516  110651                           <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
 (1)
```



```
1604                                    ,SBTTL  TIMSRV--TIMEOUT ROUTINE--SENDS  REP
1605                                    ;
1606                                    ,ENABLE LSB
1607  014520                            TIMSRV: BRWRTE  IMM,177                         ;MASK OFF  BR REQ
 (1)                 001255                    MICPC=MICPC+1
 (1)  014520  000577                           <MOVE!WRTEBR!IMM!<177>>
 (1)
1608                                                                                    ;RESET TIMER---SLICK MOVE
1609                                                                                    ;SINCE TIMER IS RESET BY WRITING
1610                                                                                    ;A 1 AND THE EXPIRATION   LOOKS
1611                                                                                    ;LIKE 1--VOILA
1612  014522                                    OUT     BR,<AANDB!OBR>                  ;AND THE BIT ON
 (1)                 001256                    MICPC=MICPC+1
 (1)  014522  061271                           <MOVE!WROUTX!BR!<AANDB!OBR>>
 (1)
1613  014524                                    BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
 (1)                 001257                    MICPC=MICPC+1
 (1)  014524  060601                           <MOVE!WRTEBR!BR!<SELA!SP1>>
 (1)
1614  014526                                    BR0     IDLE
 (1)                 001260                    MICPC=MICPC+1
 (1)  014526  102051                           <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 (1)
1615  014530                                    BR7     IDLE                            ;IF IN MAINT, MODE DISABLE TIMER
 (1)                 001261                    MICPC=MICPC+1
 (1)  014530  103451                           <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 (1)
1616  014532                                    SP      BR,DECA,SP15                    ;DECREMENT THE COUNTER
 (1)                 001262                    MICPC=MICPC+1
 (1)  014532  063175                           <MOVE!SPX!BR!DECA!SP15>
 (1)
1617  014534                                    Z       20$                             ;IF ALL ONES HAS EXPIRED
 (1)                 001263                    MICPC=MICPC+1
 (1)  014534  111670                           <JUMP!ZCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
 (1)
1618  014536                            10$:    BRWRTE  BR,SELA!SP10            ;READ LINE STATUS
 (1)                 001264                    MICPC=MICPC+1
 (1)  014536  060610                           <MOVE!WRTEBR!BR!<SELA!SP10>>
 (1)
1619  014540                                    BR1     TABUPD                          ;NUMBERED MESSAGE IN PROGRESS
 (1)                 001265                    MICPC=MICPC+1
 (1)  014540  116731                           <JUMP!BR1CON!<TABUPD-INIT&3000*4>!<TABUPD-INIT&777/2>>
 (1)
1620  014542                                    BR0     TABUPD                          ;UNNUMBMSGIN PROGRESS
 (1)                 001266                    MICPC=MICPC+1
 (1)  014542  116331                           <JUMP!BROCON!<TABUPD-INIT&3000*4>!<TABUPD-INIT&777/2>>
 (1)
1621  014544                                    ALWAYS  IDLE                            ;ELSE BACK TO IDLE LOOP
 (1)                 001267                    MICPC=MICPC+1
 (1)  014544  100451                           <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 (1)
1622  014546                            TIMF1:!
1623  014546                            20$:    BRWRTE  IMM,2                           ;
 (1)                 001270                    MICPC=MICPC+1
 (1)  014546  000402                           <MOVE!WRTEBR!IMM!<2>>
 (1)
```

```
1624  014550                          SP      BR,SELB,SP15                    ;RESET THE TIMER TICK COUNT
 (1)          001271                 MICPC=MICPC+1
 (1)  014550  063235                 <MOVE!SPX!BR!SELB!SP15>
 (1)
1625              001                 .IF NDF $LOW
1626  014552                          BRWRTE  IMM,201                         ;SET OK TO SEND AND
 (1)          001272                 MICPC=MICPC+1
 (1)  014552  000601                 <MOVE!WRTEBR!IMM!<201>>
 (1)
1627  014554                          SPBR    BR,AORB,SP10                    ;UNNUM MSG PENDING
 (1)          001273                 MICPC=MICPC+1
 (1)  014554  063710                 <MOVE!SPBRX!BR!AORB!SP10>
 (1)
1628              000                 .ENDC
1629              001                 .IF DF  $LOW
1630                                  BRWRTE  DP,<SELA!SP10>                  ;READ LINE STATUS WORD
1631              000                 .ENDC
1632  014556                          BRSHFT
 (1)          001274                 MICPC=MICPC+1
 (1)  014556  001620                 <MOVE!SHFTBR!WRTEBR!SELB>
 (1)
1633  014560                          BR4     BS1                             ;IF IN START MODE--BRANCH
 (1)          001275                 MICPC=MICPC+1
 (1)  014560  103111                 <JUMP!BR4CON!<BS1-INIT&3000*4>!<BS1-INIT&777/2>>
 (1)
1634  014562                          BRWRTE  BR,DECA!SP12                    ;GET LAST NUMBER SENT
 (1)          001276                 MICPC=MICPC+1
 (1)  014562  060572                 <MOVE!WRTEBR!BR!<DECA!SP12>>
 (1)
1635  014564                          CMP     BR,SP17                         ;COMPARE TO LAST ACKED
 (1)          001277                 MICPC=MICPC+1
 (1)  014564  060377                 <SUBTC!BR!SP17>
 (1)
1636  014566                          Z       SNDACK                  ;IF EQ --SEND ACK
 (1)          001300                 MICPC=MICPC+1
 (1)  014566  111733                 <JUMP!ZCOND!<SNDACK-INIT&3000*4>!<SNDACK-INIT&777/2>>
 (1)
1637  014570                    TIME2:LDMA    IMM,T                           ;LOAD ADDRESS OF TYPE FIELD IN UNNUMB SK
 (1)          001301                 MICPC=MICPC+1
 (1)              001                 .IF IDN IMM,IMM
 (1)  014570  010151                 <MOVE!LDMAR!IMM!<T&377>>
 (1)                                 .IFF
 (1)                                 <MOVE!LDMAR!IMM!<T>>
 (1)              000                 .ENDC
 (1)
1638  014572                          MEMINC  IMM,3                           ;LOAD REP TYPE
 (1)          001302                 MICPC=MICPC+1
 (1)  014572  016403                 <MOVE!WRMEM!INCMAR!IMM!<3>>
 (1)
1639  014574                          MEMINC  IMM,300                         ;ZERO THE SUB-TYPE
 (1)          001303                 MICPC=MICPC+1
 (1)  014574  016700                 <MOVE!WRMEM!INCMAR!IMM!<300>>
 (1)
1640  014576                          LDMA    IMM,REPCS                       ;CUMULATIVE REPS RECD
 (1)          001304                 MICPC=MICPC+1
 (1)              001                 .IF IDN IMM,IMM
```



```
 (1)  014576  010015                 <MOVE!LDMAR!IMM!<REPCS&377>>
 (1)                                 .IFF
 (1)                                 <MOVE!LDMAR!IMM!<REPCS>>
 (1)              000                 .ENDC
 (1)
1641  014600                          SP      MEMX,SELB,SP0                   ;COPY IT TO SP0
 (1)          001305                 MICPC=MICPC+1
 (1)  014600  043220                 <MOVE!SPX!MEMX!SELB!SP0>
 (1)
1642  014602                          MEM     BR,INCA!SP0                     ;INCREMENT IT
 (1)          001306                 MICPC=MICPC+1
 (1)  014602  062460                 <MOVE!WRMEM!BR!<INCA!SP0>>
 (1)
1643  014604                          LDMA    IMM,REPST                       ;ADDRESS DYNAMIC REP COUNTER
 (1)          001307                 MICPC=MICPC+1
 (1)              001                 .IF IDN IMM,IMM
 (1)  014604  010003                 <MOVE!LDMAR!IMM!<REPST&377>>
 (1)                                 .IFF
 (1)                                 <MOVE!LDMAR!IMM!<REPST>>
 (1)              000                 .ENDC
 (1)
1644  014606                          BRWRTE  MEMX,SELB                       ;COPY IT TO THE BR
 (1)          001310                 MICPC=MICPC+1
 (1)  014606  040620                 <MOVE!WRTEBR!MEMX!<SELB>>
 (1)
1645  014610                          BSHFTB
 (1)          001311                 MICPC=MICPC+1
 (1)  014610  061620                 <MOVE!SHFTBR!SELB!BR>
 (1)
1646  014612                          MEM     BP,SELB
 (1)          001312                 MICPC=MICPC+1
 (1)  014612  062620                 <MOVE!WRMEM!BP!<SELB>>
 (1)
1647  014614                          BRO     RTHRES
 (1)          001313                 MICPC=MICPC+1
 (1)  014614  106372                 <JUMP!BROCON!<RTHRES-INIT&3000*4>!<RTHRES-INIT&777/2>>
 (1)
1648              001                 .IF DF  $LOW
1649                                  BRWRTE  IMM,201                 ;MASK FOR OK TO SEND
1650                                  SP      BR,AORB,SP10            ;OR IT IN
1651              000                 .ENDC
1652  014616                          ALWAYS  IDLE
 (1)          001314                 MICPC=MICPC+1
 (1)  014616  100451                 <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 (1)
1653                                  .DSABLE LSB
1654                                  ;
```

```
1656  014620                        TEOM:   BRWRTE  IBUS,UBBR
 (1)          001315                        MICPC=MICPC+1
 (1)  014620  120620                        <MOVE!WRTEBR!IBUS!<UBBR>>
 (1)
1657  014622                                BRO     NXMERR                          ;NON-EXISTANT MEMORY
 (1)          001316                        MICPC=MICPC+1
 (1)  014622  106351                        <JUMP!BROCON!<NXMERR-INIT&3000*4>!<NXMERR-INIT&777/2>>
 (1)
1658  014624                                BRWRTE  IMM,2                   ;EOM TO BR
 (1)          001317                        MICPC=MICPC+1
 (1)  014624  000402                        <MOVE!WRTEBR!IMM!<2>>
 (1)
1659  014626                                OUTPUT  BR,<SELB!OTMTCO>        ;WRITE TMTR CONTROL
 (1)          001320                        MICPC=MICPC+1
 (1)  014626  062231                        <MOVE!WROUT!BR!<SELB!OTMTCO>>
 (1)
1660  014630                                OUTPUT  BR,<SELB!TMTDAT>        ;WRITE GARBAGE DATA
 (1)          001321                        MICPC=MICPC+1
 (1)  014630  062230                        <MOVE!WROUT!BR!<SELB!TMTDAT>>
 (1)
1661  014632                                BRWRTE  BR,SELA!SP1             ;CKECK FOR BOOT MODE
 (1)          001322                        MICPC=MICPC+1
 (1)  014632  060601                        <MOVE!WRTEBR!BR!<SELA!SP1>>
 (1)
1662  014634                                BR7     BTEOM                   ;---IF SET IS MAINT MSG
 (1)          001323                        MICPC=MICPC+1
 (1)  014634  113762                        <JUMP!BR7CON!<BTEOM-INIT&3000*4>!<BTEOM-INIT&777/2>>
 (1)
1663  014636                                SP      BR,INCA,SP12            ;INCREMENT THE MESSAGE NUMBER
 (1)          001324                        MICPC=MICPC+1
 (1)  014636  063072                        <MOVE!SPX!BR!INCA!SP12>
 (1)
1664  014640                        TEOM1:  LDMA    BR,SELA!SP16                    ;ADDRESS LAST TMT LINK
 (1)          001325                        MICPC=MICPC+1
 (1)             001                        .IF IDN BR,IMM
 (1)                                        <MOVE!LDMAR!IMM!<SELA!SP16&377>>
 (1)                                        .IFF
 (1)  014640  070216                        <MOVE!LDMAR!BR!<SELA!SP16>>
 (1)             000                        .ENDC
 (1)
1665  014642                                BRWRTE  MEMX,SELB
 (1)          001326                        MICPC=MICPC+1
 (1)  014642  040620                        <MOVE!WRTEBR!MEMX!<SELB>>
 (1)
1666  014644                                BRO     TEOM2
 (1)          001327                        MICPC=MICPC+1
 (1)  014644  112340                        <JUMP!BROCON!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
 (1)
1667  014646                        TEOM3:  BRWRTE  IMM,375                 ;TURN OFF MESSAGE PENDING
 (1)          001330                        MICPC=MICPC+1
 (1)  014646  000775                        <MOVE!WRTEBR!IMM!<375>>
 (1)
1668  014650                                SPBR    BR,AANDB,SP10           ;
 (1)          001331                        MICPC=MICPC+1
 (1)  014650  063670                        <MOVE!SPBRX!BR!AANDB!SP10>
 (1)
```



```
1669  014652                                BRO     TEOM2                   ;IF UNNUMB PENDING--GO AWAY
 (1)          001332                        MICPC=MICPC+1
 (1)  014652  112340                        <JUMP!BROCON!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
 (1)
1670                                        .SBTTL  SNDACK--ROUTINE TO SEND AN ACK
1671  014654                        SNDACK: LDMA    IMM,T
 (1)          001333                        MICPC=MICPC+1
 (1)             001                        .IF IDN IMM,IMM
 (1)  014654  010151                        <MOVE!LDMAR!IMM!<T&377>>
 (1)                                        .IFF
 (1)                                        <MOVE!LDMAR!IMM!<T>>
 (1)             000                        .ENDC
 (1)
1672  014656                                MEMINC  IMM,1
 (1)          001334                        MICPC=MICPC+1
 (1)  014656  016401                        <MOVE!WRMEM!INCMAR!IMM!<1>>
 (1)
1673  014660                                BRWRTE  IMM,5
 (1)          001335                        MICPC=MICPC+1
 (1)  014660  000405                        <MOVE!WRTEBR!IMM!<5>>
 (1)
1674  014662                        SA2:    MEMINC  IMM,300
 (1)          001336                        MICPC=MICPC+1
 (1)  014662  016700                        <MOVE!WRMEM!INCMAR!IMM!<300>>
 (1)
1675  014664                        SA3:    SP      BR,AORB,SP10
 (1)          001337                        MICPC=MICPC+1
 (1)  014664  063310                        <MOVE!SPX!BR!AORB!SP10>
 (1)
1676                                        ;
1677             001                        .IF DF  $LOW
1678                             TEOM2:     STATE   TMTA
1679                                        ALWAYS  XEXIT
1680             000                        .ENDC
1681             001                        .IF NDF $LOW
1682  014666                     TEOM2:     TSTATE  TMTA
 (1)          001340                        MICPC=MICPC+1
 (1)  014666  000403                        <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
 (1)          001341                        MICPC=MICPC+1
 (1)  014670  063222                        <MOVE!SPX!BR!SELB!SP2>
1683  014672                                ALWAYS  I1
 (1)          001342                        MICPC=MICPC+1
 (1)  014672  100454                        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 (1)
1684             000                        .ENDC
1685  014674                        FUDGE:  BRWRTE  IBUS,NPR                ;READ NPR CONTROL
 (1)          001343                        MICPC=MICPC+1
 (1)  014674  120500                        <MOVE!WRTEBR!IBUS!<NPR>>
 (1)
1686  014676                                BRO     IDLE                    ;IF NPR GOING---LEAVE
 (1)          001344                        MICPC=MICPC+1
 (1)  014676  102351                        <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 (1)
1687  014700                                BRWRTE  BR!LDMAR,SELA!SP4       ;LOAD THE MAR
 (1)          001345                        MICPC=MICPC+1
 (1)  014700  070004                        <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
```

```
 (1)
1688  014702                          BR7     BS2                     ;IF SET - READ BACK ALL 200
 (1)          001346                 MICPC=MICPC+1
 (1)  014702  103520                 <JUMP!BR7CON!<BS2-INIT&3000*4>!<BS2-INIT&777/2>>
 (1)
1689  014704                          MEMINC  IBUS,INDAT1             ;OTHERWISE RESTORE TWO BYTES
 (1)          001347                 MICPC=MICPC+1
 (1)  014704  036400                 <MOVE!WRMEM!INCMAR!IBUS!<INDAT1>>
 (1)
1690  014706                          MEMINC  IBUS,INDAT2             ;..
 (1)          001350                 MICPC=MICPC+1
 (1)  014706  036420                 <MOVE!WRMEM!INCMAR!IBUS!<INDAT2>>
 (1)
1691  014710                          BRWRTE  IMM,2                   ;UPDATE---UNIBUS ADDRESS
 (1)          001351                 MICPC=MICPC+1
 (1)  014710  000402                 <MOVE!WRTEBR!IMM!<2>>
 (1)
1692  014712                          SP      BR,ADD,SP4              ;UPDATE NPR COUNTER
 (1)          001352                 MICPC=MICPC+1
 (1)  014712  063004                 <MOVE!SPX!BR!ADD!SP4>
 (1)
1693  014714                          SP      IBUS,IIBA1,SP0          ;UPDATE ADDRESS LOW
 (1)          001353                 MICPC=MICPC+1
 (1)  014714  023100                 <MOVE!SPX!IBUS!IIBA1!SP0>
 (1)
1694  014716                          OUTPUT  BR,ADD!IBA1
 (1)          001354                 MICPC=MICPC+1
 (1)  014716  062004                 <MOVE!WROUT!BR!<ADD!IBA1>>
 (1)
1695  014720                          SP      IBUS,IIBA2,SP0          ;READ HIGH ADDRESS
 (1)          001355                 MICPC=MICPC+1
 (1)  014720  023120                 <MOVE!SPX!IBUS!IIBA2!SP0>
 (1)
1696  014722                          OUTPUT  BR,AC!IBA2              ;UPDATE HIGH
 (1)          001356                 MICPC=MICPC+1
 (1)  014722  062105                 <MOVE!WROUT!BR!<AC!IBA2>>
 (1)
1697  014724                          SP      IBUS,NPR,SP0            ;READ NPR REGISTER
 (1)          001357                 MICPC=MICPC+1
 (1)  014724  123200                 <MOVE!SPX!IBUS!NPR!SP0>
 (1)
1698  014726                          C       RESEXT                  ;IF CARRY---UPDATE MXT
 (1)          001360                 MICPC=MICPC+1
 (1)  014726  105363                 <JUMP!CCOND!<RESEXT-INIT&3000*4>!<RESEXT-INIT&777/2>>
 (1)
1699  014730                          ALWAYS  TH3X                    ;GO DO ANOTHER NPR
 (1)          001361                 MICPC=MICPC+1
 (1)  014730  110601                 <JUMP!ALCOND!<TH3X-INIT&3000*4>!<TH3X-INIT&777/2>>
 (1)
1700  014732                    BTEOM:  BRWRTE  IMM,374                 ;MASK FOR CLEAR MSG PENDING
 (1)          001362                 MICPC=MICPC+1
 (1)  014732  000774                 <MOVE!WRTEBR!IMM!<374>>
 (1)
1701  014734                          SP      BR,AANDB,SP10           ;TURN THEM OFF IN LINE STATUS WORD
 (1)          001363                 MICPC=MICPC+1
 (1)  014734  063270                 <MOVE!SPX!BR!AANDB!SP10>
```



```
 (1)
1702  014736                          SP      BR,SELB,SP13            ;STORE UNRECOGNIZABLE VALUE INTO SP13
 (1)          001364                 MICPC=MICPC+1
 (1)  014736  063233                 <MOVE!SPX!BR!SELB!SP13>
 (1)
1703                                                                 ;SO "RH3" WILL EXIT BACK TO IDLE LOOP
1704  014740                          LDMA    IMM,STC                 ;ADDRESS START OF TMT  CHAIN
 (1)          001365                 MICPC=MICPC+1
 (1)             001                 .IF IDN IMM,IMM
 (1)  014740  010067                 <MOVE!LDMAR!IMM!<STC&377>>
 (1)                                 .IFF
 (1)                                 <MOVE!LDMAR!IMM!<STC>>
 (1)             000                 .ENDC
 (1)
1705  014742                          SP      MEMX,SELB,SP0           ;COPY LINK ADDRESS
 (1)          001366                 MICPC=MICPC+1
 (1)  014742  043220                 <MOVE!SPX!MEMX!SELB!SP0>
 (1)
1706             001                 .IF DF  $LOW
1707                                 TSTATE  NUMSYN                  ;CHANGE XMIT STATE TO LINE IS IDLE
1708             000                 .ENDC
1709             001                 .IF NDF $LOW
1710  014744                         TSTATE  TMTA                    ;CHANGE XMIT STATE TO LINE IS IDLE
 (1)          001367                 MICPC=MICPC+1
 (1)  014744  000403                 <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
 (1)          001370                 MICPC=MICPC+1
 (1)  014746  063222                 <MOVE!SPX!BR!SELB!SP2>
1711             000                 .ENDC
1712  014750                         ALWAYS  TDON2                   ;POST A DONE
 (1)          001371                 MICPC=MICPC+1
 (1)  014750  114532                 <JUMP!ALCOND!<TDON2-INIT&3000*4>!<TDON2-INIT&777/2>>
 (1)
1713  014752                    RL4:  RSTATE  RCVL
 (1)          001372                 MICPC=MICPC+1
 (1)  014752  000705                 <MOVE!WRTEBR!IMM!<RCVL-INIT&777/2>>
 (1)          001373                 MICPC=MICPC+1
 (1)  014754  063223                 <MOVE!SPX!BR!SELB!SP3>
1714  014756                         SP      IBUS,NPR,SP0            ;READ NPR CONTROL REGISTER
 (1)          001374                 MICPC=MICPC+1
 (1)  014756  123200                 <MOVE!SPX!IBUS!NPR!SP0>
 (1)
1715  014760                         BRWRTE  IMM,221
 (1)          001375                 MICPC=MICPC+1
 (1)  014760  000621                 <MOVE!WRTEBR!IMM!<221>>
 (1)
1716  014762                         ALWAYS  RK7
 (1)          001376                 MICPC=MICPC+1
 (1)  014762  104623                 <JUMP!ALCOND!<RK7-INIT&3000*4>!<RK7-INIT&777/2>>
 (1)
1717  014764                    HOINCH: SP      IBUS,IIBA2,SP0
 (1)          001377                 MICPC=MICPC+1
 (1)  014764  023120                 <MOVE!SPX!IBUS!IIBA2!SP0>
 (1)
1718  014766                         OUTPUT  BR,INCA!IBA2                    ;OUTPUT INCREMENTED BA
 (1)          001400                 MICPC=MICPC+1
 (1)  014766  062065                 <MOVE!WROUT!BR!<INCA!IBA2>>
```

```
 (1)
1719  014770                        C       5$                          ;INCREMENT BYTEW COUNT
 (1)          001401                MICPC=MICPC+1
 (1)  014770  115003                <JUMP!CCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
 (1)
1720  014772                        ALWAYS  TH8
 (1)          001402                MICPC=MICPC+1
 (1)  014772  110571                <JUMP!ALCOND!<TH8-INIT&3000*4>!<TH8-INIT&777/2>>
 (1)
1721                                ;INCREMENT MXT BITS
1722  014774                  5$:   SP      IBUS,NPR,SP0              ;READ NPR REG IWTH CURRENT MXT BITS
 (1)          001403                MICPC=MICPC+1
 (1)  014774  123200                <MOVE!SPX!IBUS!NPR!SP0>
 (1)
1723  014776                        BRWRTE  IMM,4                     ;WRITE BIT TO ADD
 (1)          001404                MICPC=MICPC+1
 (1)  014776  000404                <MOVE!WRTEBR!IMM!<4>>
 (1)
1724  015000                        OUT     BR,<ADD!ONPR>             ;TURN ON PROPER MXT BITS
 (1)          001405                MICPC=MICPC+1
 (1)  015000  061010                <MOVE!WROUTX!BR!<ADD!ONPR>>
 (1)
1725  015002                        ALWAYS  TH8
 (1)          001406                MICPC=MICPC+1
 (1)  015002  110571                <JUMP!ALCOND!<TH8-INIT&3000*4>!<TH8-INIT&777/2>>
 (1)
1726                                ;
1727             001                .IF DF  LOW
1728  015004                  HEH1: STATE   TEOM
 (1)          001407                MICPC=MICPC+1
 (1)  015004  000715                <MOVE!WRTEBR!IMM!<TEOM-INIT&777/2>>
1729  015006                        ALWAYS  XFXIT
 (1)          001410                MICPC=MICPC+1
 (1)  015006  110563                <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
 (1)
1730             000                .ENDC
1731                                ;
1732             001                .IF NDF LOW
1733                  HEH1: TSTATE  TEOM
1734                        ALWAYS  I1
1735             000                .ENDC
```



```
1737                                .SBTTL  REP HANDLER
1738  015010                  REP:  LDMA    IMM,REPCR                 ;LOAD MAR ADDRESS WITH POINTER TO REPS RECC
 (1)          001411                MICPC=MICPC+1
 (1)             001                .IF IDN IMM,IMM
 (1)  015010  010016                <MOVE!LDMAR!IMM!<REPCR&377>>
 (1)                                .IFF
 (1)                                <MOVE!LDMAR!IMM!<REPCR>>
 (1)             000                .ENDC
 (1)
1739  015012                        SP      MEMX,SELB,SP0             ;READ NUMBER OF REPS RECD
 (1)          001412                MICPC=MICPC+1
 (1)  015012  043220                <MOVE!SPX!MEMX!SELB!SP0>
 (1)
1740  015014                        MEM     DP,<INCA!SP0>             ;INCREMNT REPS RECD
 (1)          001413                MICPC=MICPC+1
 (1)  015014  062460                <MOVE!WRMEM!DP!<INCA!SP0>>
 (1)
1741  015016                        LDMA    IMM,T                     ;LOAD ADDRESS OF TYPE FIELD
 (1)          001414                MICPC=MICPC+1
 (1)             001                .IF IDN IMM,IMM
 (1)  015016  010151                <MOVE!LDMAR!IMM!<T&377>>
 (1)                                .IFF
 (1)                                <MOVE!LDMAR!IMM!<T>>
 (1)             000                .ENDC
 (1)
1742  015020                        MEMINC  IMM,2                     ; LOAD NAK TYPE
 (1)          001415                MICPC=MICPC+1
 (1)  015020  016402                <MOVE!WRMEM!INCMAR!IMM!<2>>
 (1)
1743  015022                        MEMINC  IMM,303                   ;LOAD REP RESPONSE SUB-TYPE
 (1)          001416                MICPC=MICPC+1
 (1)  015022  016703                <MOVE!WRMEM!INCMAR!IMM!<303>>
 (1)
1744  015024                        ALWAYS  SNAK                      ;SEND AN UNNUMB MSG
 (1)          001417                MICPC=MICPC+1
 (1)  015024  114704                <JUMP!ALCOND!<SNAK-INIT&3000*4>!<SNAK-INIT&777/2>>
 (1)
1745                                ;
1746                                .SBTTL  START HANDLER
1747  015026                  START: BRWRTE  DP,<SELA!SP10>            ;READ LINE STATUS WORD
 (1)          001420                MICPC=MICPC+1
 (1)  015026  060610                <MOVE!WRTEBR!DP!<SELA!SP10>>
 (1)
1748  015030                        BPSHFT                            ;GET START MODE BIT IN TESTABLE POSITION
 (1)          001421                MICPC=MICPC+1
 (1)  015030  001620                <MOVE!SHFTBR!WRTEBR!SELB>
 (1)
1749  015032                        BR4     10$                       ;IF IN START MODE SET STACK
 (1)          001422                MICPC=MICPC+1
 (1)  015032  117026                <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
 (1)
1750                                                                  ;ELSE SET UP START ERROR
1751  015034                        LDMA    IMM,<<RTHRS+3>>
 (1)          001423                MICPC=MICPC+1
 (1)             001                .IF IDN IMM,IMM
 (1)  015034  010177                <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
```

```
                                        .IFF
                                        <MOVE!LDMAR!IMM!<<RTHRS+3>>>
                000                     .ENDC

1752  015036                            BRWRTE  IMM,200
                001424                  MICPC=MICPC+1
      015036    000600                  <MOVE!WRTEBR!IMM!<200>>

1753  015040                            ALWAYS  RCEXY
                001425                  MICPC=MICPC+1
      015040    114522                  <JUMP!ALCOND!<RCEXY-INIT&3000*4>!<RCEXY-INIT&777/2>>

1754  015042                      10$:  LDMA    IMM,T                   ;SET UP ADDRESS OF TYPE FIELD
                001426                  MICPC=MICPC+1
                  001                   .IF IDN IMM,IMM
      015042    010151                  <MOVE!LDMAR!IMM!<T&377>>
                                        .IFF
                                        <MOVE!LDMAR!IMM!<T>>
                000                     .ENDC

1755  015044                            MEMINC  IMM,7                   ;WRITE STACK TYPE
                001427                  MICPC=MICPC+1
      015044    016407                  <MOVE!WRMEM!INCMAR!IMM!<7>>

1756  015046                            BRWRTE  IMM,11                  ;SET START RECD AND UNNUMB PENDING
                001430                  MICPC=MICPC+1
      015046    000411                  <MOVE!WRTEBR!IMM!<11>>

1757  015050                            ALWAYS  SA2                     ;SEND THE UNNUMBERED MESSAGE
                001431                  MICPC=MICPC+1
      015050    110736                  <JUMP!ALCOND!<SA2-INIT&3000*4>!<SA2-INIT&777/2>>

1758                                    ;
1759                                    .SBTTL  STACK HANDLER
1760  015052                      STACK:BRWRTE  IMM,327                 ;MASK TO CLEAR START MODE
                001432                  MICPC=MICPC+1
      015052    000727                  <MOVE!WRTEBR!IMM!<327>>

1761  015054                            SP      BR,AANDB,SP10           ;CLEAR START MODE
                001433                  MICPC=MICPC+1
      015054    063270                  <MOVE!SPX!BR!AANDB!SP10>

1762  015056                            ALWAYS  TIME1                   ;RESET TIMER AND IDLE
                001434                  MICPC=MICPC+1
      015056    110670                  <JUMP!ALCOND!<TIME1-INIT&3000*4>!<TIME1-INIT&777/2>>
```



```
1764  015060                      ICBA22: SP      IBUS,IOBA2,SP0          ;READTHEHIGH ORDERBITS OF BA TO SP0
                001435                  MICPC=MICPC+1
      015060    023160                  <MOVE!SPX!IBUS!IOBA2!SP0>

1765  015062                            OUTPUT  DP,<INCA!OBA2>          ;OUTPUT THE INCREMENTED COUNT
                001436                  MICPC=MICPC+1
      015062    062067                  <MOVE!WROUT!DP!<INCA!OBA2>>

1766  015064                            C       5$                      ;IF CARRY SET INCREMENT THE MXTBITS
                001437                  MICPC=MICPC+1
      015064    115041                  <JUMP!CCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>

1767  015066                            ALWAYS  RK3
                001440                  MICPC=MICPC+1
      015066    104641                  <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>

1768                                    ;
1769  015070                      5$:   SP      IBUS,UBBR,SP0
                001441                  MICPC=MICPC+1
      015070    123220                  <MOVE!SPX!IBUS!UBBR!SP0>

1770  015072                            BRWRTE  IMM,4
                001442                  MICPC=MICPC+1
      015072    000404                  <MOVE!WRTEBR!IMM!<4>>

1771  015074                            OUT     BR,<ADD!OBR>
                001443                  MICPC=MICPC+1
      015074    061011                  <MOVE!WROUTX!BR!<ADD!OBR>>

1772  015076                            ALWAYS  RK3
                001444                  MICPC=MICPC+1
      015076    104641                  <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>

1773                001                 .IF DF  $LOW
1774                                FLUSH1: OUTPUT  IMM,<200!ORCVCO>        ;FLUSH THE RECVR
1775                                    ALWAYS  CG1
1776                000                 .ENDC
1777  015100                      NAK:  LDMA    IMM,NDATR               ;CUMMULATIVE NAK COUNTER
                001445                  MICPC=MICPC+1
                  001                   .IF IDN IMM,IMM
      015100    010011                  <MOVE!LDMAR!IMM!<NDATR&377>>
                                        .IFF
                                        <MOVE!LDMAR!IMM!<NDATR>>
                000                     .ENDC

1778  015102                            SP      MEMX,SELB,SP0           ;READ IT
                001446                  MICPC=MICPC+1
      015102    043220                  <MOVE!SPX!MEMX!SELB!SP0>

1779  015104                            MEM     MEMX,INCA,SP0           ;INCREMENT THE COUNTER
                001447                  MICPC=MICPC+1
      015104    012460                  <MOVE!WRMEM!MEMX!<INCA!SP0>>

1780  015106                            LDMA    IMM,STC                 ;ADDRESS START OF TMT CHAIN
                001450                  MICPC=MICPC+1
                  001                   .IF IDN IMM,IMM
```

```
   (1)  015106  010067                        <MOVE!LDMAR!IMM!<STC&377>>
   (1)                                        .IFF
   (1)                                        <MOVE!LDMAR!IMM!<STC>>
   (1)                000                     .ENDC
   (1)
  1781  015110                                SP      MEMX,SELB,SP16          ;COPY START OF CHAIN TO LAST XMIT POINTER
   (1)          001451                       MICPC=MICPC+1
   (1)  015110  043236                       <MOVE!SPX!MEMX!SELB!SP16>
   (1)
  1782  015112                                BRWRTE  BR,INCA!SP17                    ;GETLASTMESSAGE ACKED
   (1)          001452                       MICPC=MICPC+1
   (1)  015112  060477                       <MOVE!WRTEBR!BR!<INCA!SP17>>
   (1)
  1783  015114                                SP      BR,SELB,SP12                    ;COPY TO CURRENT NUMBER
   (1)          001453                       MICPC=MICPC+1
   (1)  015114  063232                       <MOVE!SPX!BR!SELB!SP12>
   (1)
  1784  015116                                BRWRTE  IMM,6                   ;WRITE NUMBERED MSG PENDING
   (1)          001454                       MICPC=MICPC+1
   (1)  015116  000406                       <MOVE!WRTEBR!IMM!<6>>
   (1)
  1785                                                                      ; AND LINE HAS GONE IDLE
  1786  015120                                SP      BR,AORB,SP10            ;SET IT IN LINE STATUS WORD
   (1)          001455                       MICPC=MICPC+1
   (1)  015120  063310                       <MOVE!SPX!BR!AORB!SP10>
   (1)
  1787  015122                                SP      BR,SELB,SP15            ;RESET TIMER COUNT
   (1)          001456                       MICPC=MICPC+1
   (1)  015122  063235                       <MOVE!SPX!BR!SELB!SP15>
   (1)
  1788  015124                                ALWAYS  TEOM1
   (1)          001457                       MICPC=MICPC+1
   (1)  015124  110725                       <JUMP!ALCOND!<TEOM1-INIT&3000*4>!<TEOM1-INIT&777/2>>
   (1)
  1789  015126                        ININT:  BRWRTE  IMM,15                  ;MASK FOR TURN OFF ALL BUT EXT MEM BITS + NXM
   (1)          001460                       MICPC=MICPC+1
   (1)  015126  000415                       <MOVE!WRTEBR!IMM!<15>>
   (1)
  1790  015130                                SP      IBUS,UBBR,SP0           ;READ BR  CONTROL REGISTER
   (1)          001461                       MICPC=MICPC+1
   (1)  015130  123220                       <MOVE!SPX!IBUS!UBBR!SP0>
   (1)
  1791  015132                                SP      BR,AANDB,SP0            ;MASK OFF VECTOR TO X04
   (1)          001462                       MICPC=MICPC+1
   (1)  015132  063260                       <MOVE!SPX!BR!AANDB!SP0>
   (1)
  1792  015134                                BRWRTE  IMM,200                 ;MASK FOR INTERRUPT
   (1)          001463                       MICPC=MICPC+1
   (1)  015134  000600                       <MOVE!WRTEBR!IMM!<200>>
   (1)
  1793  015136                                OUT     BR,AORB!OBR             ;INTERRUPT
   (1)          001464                       MICPC=MICPC+1
   (1)  015136  061311                       <MOVE!WROUTX!BR!<AORB!OBR>>
   (1)
  1794  015140                                SP      IBUS,INCON,SP0          ;RESTORE INPUT CONTROL CSR
   (1)          001465                       MICPC=MICPC+1
```



```
   (1)  015140  123000                        <MOVE!SPX!IBUS!INCON!SP0>
   (1)
  1795  015142                                ALWAYS  NIDLE4
   (1)          001466                       MICPC=MICPC+1
   (1)  015142  100554                       <JUMP!ALCOND!<NIDLE4-INIT&3000*4>!<NIDLE4-INIT&777/2>>
   (1)
  1796                                        ;
```

```
                001                     .IF DF  SLOW
                                        .SBTTL NXMERR ---NON EXISTANT MEMORY HANDLER
                                NXMERR: LDMA    IMM,<<RTHRS+3>>         ;ADDRESS ERROR LINK
                                        MEMINC  IMM,1
                                        MEM     IMM,0                   ;NXM ERROR BIT
                                        SP      MEMX,SELB,SP10          ;CLEAR STATUS
                                        ALWAYS  RCEXX
                                        .PAGE
                000                     .ENDC
                                        ;FUGITIVE RECEIVE ROUTINES---DON'T FIT IN PAGE
 015144                         PH1:    BRWRTE  IMM,77
  (1)           001467                  MICPC=MICPC+1
  (1)  015144   000477                  <MOVE!WRTEBR!IMM!<77>>
  (1)
 015146                                 SP      BR,AANDB,SP5
  (1)           001470                  MICPC=MICPC+1
  (1)  015146   063265                  <MOVE!SPX!BR!AANDB!SP5>
  (1)
 015150                                 LDMA    BR,<INCA!SP14>          ;LOAD ADDRESS OF CURRENT COUNT
  (1)           001471                  MICPC=MICPC+1
  (1)              001                  .IF IDN BR,IMM
  (1)                                   <MOVE!LDMAR!IMM!<INCA!SP14&377>>
  (1)                                   .IFF
  (1)  015150   070074                  <MOVE!LDMAR!BR!<INCA!SP14>>
  (1)              000                  .ENDC
  (1)
 015152                                 SP      BR!INCMAR,SELB,SP0      ;SAVE MASK
  (1)           001472                  MICPC=MICPC+1
  (1)  015152   077220                  <MOVE!SPX!BR!INCMAR!SELB!SP0>
  (1)
 015154                                 BRWRTE  BR!INCMAR,SELA!SP1      ;READ STATUS BYTE
  (1)           001473                  MICPC=MICPC+1
  (1)  015154   074601                  <MOVE!WRTEBR!BR!INCMAR!<SELA!SP1>>
  (1)
 015156                                 BRSHFT                          ;SHIFT IT RIGHT
  (1)           001474                  MICPC=MICPC+1
  (1)  015156   001620                  <MOVE!SHFTBR!WRTEBR!SELB>
  (1)
 015160                                 BR1     RH2                     ;NO BUFFER ASSIGNED IN MAINT MODE
  (1)           001475                  MICPC=MICPC+1
  (1)  015160   116502                  <JUMP!BR1CON!<RH2-INIT&3000*4>!<RH2-INIT&777/2>>
  (1)
 015162                                 BRWRTE  MEMX!INCMAR,AANDB!SP0   ;GET HIGH BYTE COUNT BITS
  (1)           001476                  MICPC=MICPC+1
  (1)  015162   054660                  <MOVE!WRTEBR!MEMX!INCMAR!<AANDB!SP0>>
  (1)
 015164                                 CMP     BR,SP5          ;COMPARE HIGH ORDER BITS OF COUNT
  (1)           001477                  MICPC=MICPC+1
  (1)  015164   060365                  <SUBTC!BR!SP5>
  (1)
 015166                                 C       RCFATL                  ;IF CARRY--TOO BIG ERROR
  (1)           001500                  MICPC=MICPC+1
  (1)  015166   115113                  <JUMP!CCOND!<RCFATL-INIT&3000*4>!<RCFATL-INIT&777/2>>
  (1)
 015170                                 Z       RCLOW                   ;IF EQUAL COMPARE LOW ORDER BITS OF COUNT
  (1)           001501                  MICPC=MICPC+1
```



```
  (1)  015170   115510                  <JUMP!ZCOND!<RCLOW-INIT&3000*4>!<RCLOW-INIT&777/2>>
  (1)
 015172                         RH2:    BRWRTE  IBUS,IOBA1                      ;READ LOW BYTE OF IN BA
  (1)           001502                  MICPC=MICPC+1
  (1)  015172   020540                  <MOVE!WRTEBR!IBUS!<IOBA1>>
  (1)
 015174                                 BP0     RCVODD                          ;IF SET IS ODD TRANSFER
  (1)           001503                  MICPC=MICPC+1
  (1)  015174   116106                  <JUMP!BR0CON!<RCVODD-INIT&3000*4>!<RCVODD-INIT&777/2>>
  (1)
                001                     .IF NDF LOW
                                        BRWRTE  IBUS,RCVCON                     ;IS THE RECEIVER READY?
                                        BR4     RCVKE0                          ;YES--GO PROCESS
                000                     .ENDC
 015176                                 STATE   RCVKE0
  (1)           001504                  MICPC=MICPC+1
  (1)  015176   000660                  <MOVE!WRTEBR!IMM!<RCVKE0-INIT&777/2>>
 015200                                 ALWAYS  REXIT
  (1)           001505                  MICPC=MICPC+1
  (1)  015200   100450                  <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
  (1)
                                ;
 015202                         RCVODD: STATE   RCVKO1
  (1)           001506                  MICPC=MICPC+1
  (1)  015202   000607                  <MOVE!WRTEBR!IMM!<RCVKO1-INIT&777/2>>
 015204                                 ALWAYS  REXIT
  (1)           001507                  MICPC=MICPC+1
  (1)  015204   100450                  <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
  (1)
                                ;
 015206                         RCLOW:  CMP     MEMX,SP4                        ;COMPARE LOW ORDER BITS OF COUNT
  (1)           001510                  MICPC=MICPC+1
  (1)  015206   040364                  <SUBTC!MEMX!SP4>
  (1)
 015210                                 C       RCFATL                          ;CARRY--TOO BIG
  (1)           001511                  MICPC=MICPC+1
  (1)  015210   115113                  <JUMP!CCOND!<RCFATL-INIT&3000*4>!<RCFATL-INIT&777/2>>
  (1)
 015212                                 ALWAYS  RH2                     ;ELSE CONTINUE
  (1)           001512                  MICPC=MICPC+1
  (1)  015212   114502                  <JUMP!ALCOND!<RH2-INIT&3000*4>!<RH2-INIT&777/2>>
  (1)
 015214                         RCFATL: LDMA    IMM,T
  (1)           001513                  MICPC=MICPC+1
  (1)              001                  .IF IDN IMM,IMM
  (1)  015214   010151                  <MOVE!LDMAR!IMM!<T&377>>
  (1)                                   .IFF
  (1)                                   <MOVE!LDMAP!IMM!<T>>
  (1)              000                  .ENDC
  (1)
 015216                                 MEMINC  IMM,2
  (1)           001514                  MICPC=MICPC+1
  (1)  015216   016402                  <MOVE!WPMEM!INCMAR!IMM!<2>>
  (1)
 015220                                 MEM     IMM,311
  (1)           001515                  MICPC=MICPC+1
```

```
   (1)  015220  002711                        <MOVE!WRMEM!IMM!<311>>
   (1)
1837  015222                                  LDMA    IMM,<<RTHRS+1>>          ;ADDRESS ERROR LINK
   (1)          001516                        MICPC=MICPC+1
   (1)             001                        .IF IDN IMM,IMM
   (1)  015222  010175                        <MOVE!LDMAR!IMM!<<RTHRS+1>&377>>
   (1)                                        .IFF
   (1)                                        <MOVE!LDMAR!IMM!<<RTHRS+1>>>
   (1)             000                        .ENDC
   (1)
1838  015224                                  MEMINC  IBUS,IOBA1
   (1)          001517                        MICPC=MICPC+1
   (1)  015224  036540                        <MOVE!WRMEM!INCMAR!IBUS!<IOBA1>>
   (1)
1839  015226                                  MEMINC  IBUS,IOBA2
   (1)          001520                        MICPC=MICPC+1
   (1)  015226  036560                        <MOVE!WRMEM!INCMAR!IBUS!<IOBA2>>
   (1)
1840  015230                                  BRWRTE  IMM,20
   (1)          001521                        MICPC=MICPC+1
   (1)  015230  000420                        <MOVE!WRTEBR!IMM!<20>>
   (1)
1841  015232                            RCEXY:  MEMINC  IMM,0
   (1)          001522                        MICPC=MICPC+1
   (1)  015232  016400                        <MOVE!WRMEM!INCMAR!IMM!<0>>
   (1)
1842  015234                                  MEM     BR,SELB
   (1)          001523                        MICPC=MICPC+1
   (1)  015234  062620                        <MOVE!WRMEM!BR!<SELB>>
   (1)
1843  015236                            RCEXX:  OUTPUT  IMM,<200!ORCVCO>         ;FLUSH INPUT SILO
   (1)          001524                        MICPC=MICPC+1
   (1)  015236  002212                        <MOVE!WROUT!IMM!<200!ORCVCO>>
   (1)
1844  015240                                  SP      IMM,SP2,2                ;INHIBIT FURTHER TRANSMISSIONS
   (1)          001525                        MICPC=MICPC+1
   (1)  015240  003002                        <MOVE!SPX!IMM!SP2!2>
   (1)
1845  015242                                  SP      IMM,1,SP1        ;SET INIT MODE IN PORT STATUS WORD
   (1)          001526                        MICPC=MICPC+1
   (1)  015242  003001                        <MOVE!SPX!IMM!1!SP1>
   (1)
1846  015244                                  ALWAYS  NTRS1
   (1)          001527                        MICPC=MICPC+1
   (1)  015244  114666                        <JUMP!ALCOND!<NTRS1-INIT&3000*4>!<NTRS1-INIT&777/2>>
   (1)
1847  015246                            TDON3:  BRWRTE  MEMX,SUB!SP17                    ;COMPARE RESPONSE TO MSG NO
   (1)          001530                        MICPC=MICPC+1
   (1)  015246  040757                        <MOVE!WRTEBR!MEMX!<SUB!SP17>>
   (1)
1848  015250                                  BR7     RH3                      ;IF NEGATIVE EXIT
   (1)          001531                        MICPC=MICPC+1
   (1)  015250  107562                        <JUMP!BR7CON!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
   (1)
1849  015252                            TDON2:  LDMA    BR,SELA!SP0              ;ADDRESS THE TRANSMITLINK
   (1)          001532                        MICPC=MICPC+1
```



```
   (1)             001                        .IF IDN BR,IMM
   (1)                                        <MOVE!LDMAR!IMM!<SELA!SP0&377>>
   (1)                                        .IFF
   (1)  015252  070200                        <MOVE!LDMAR!BR!<SELA!SP0>>
   (1)             000                        .ENDC
   (1)
1850  015254                                  MEM     IMM,0                    ;TURN OF ASSIGNEDAND TMTED BITS IN FLAG
   (1)          001533                        MICPC=MICPC+1
   (1)  015254  002400                        <MOVE!WRMEM!IMM!<0>>
   (1)
1851  015256                                  LDMA    IMM,STC
   (1)          001534                        MICPC=MICPC+1
   (1)             001                        .IF IDN IMM,IMM
   (1)  015256  010067                        <MOVE!LDMAR!IMM!<STC&377>>
   (1)                                        .IFF
   (1)                                        <MOVE!LDMAR!IMM!<STC>>
   (1)             000                        .ENDC
   (1)
1852  015260                                  MEM     IMM,TML1                 ;ASSUME WRAPAROUND
   (1)          001535                        MICPC=MICPC+1
   (1)  015260  002471                        <MOVE!WRMEM!IMM!<TML1>>
   (1)
1853  015262                                  BRWRTE  IMM,TML8                 ;WRAPAROUND?
   (1)          001536                        MICPC=MICPC+1
   (1)  015262  000543                        <MOVE!WRTEBR!IMM!<TML8>>
   (1)
1854  015264                                  CMP     BR,SP0
   (1)          001537                        MICPC=MICPC+1
   (1)  015264  060360                        <SUBTC!BR!SP0>
   (1)
1855  015266                                  Z       TDON4                    ;YES
   (1)          001540                        MICPC=MICPC+1
   (1)  015266  115543                        <JUMP!ZCOND!<TDON4-INIT&3000*4>!<TDON4-INIT&777/2>>
   (1)
1856  015270                                  BRWRTE  IMM,6                    ;OFFSET FOR NEXT TMT LINK
   (1)          001541                        MICPC=MICPC+1
   (1)  015270  000406                        <MOVE!WRTEBR!IMM!<6>>
   (1)
1857  015272                                  MEM     BR,ADD!SP0               ;UPDATE THE POINTER
   (1)          001542                        MICPC=MICPC+1
   (1)  015272  062400                        <MOVE!WRMEM!BR!<ADD!SP0>>
   (1)
1858  015274                            TDON4:  LDMA    IMM,NXTSP                ;ADDRESS DONE LINK
   (1)          001543                        MICPC=MICPC+1
   (1)             001                        .IF IDN IMM,IMM
   (1)  015274  010241                        <MOVE!LDMAR!IMM!<NXTSP&377>>
   (1)                                        .IFF
   (1)                                        <MOVE!LDMAR!IMM!<NXTSP>>
   (1)             000                        .ENDC
   (1)
1859  015276                                  LDMA    MEMX,SELB!SPX!SP3        ;ADDRESS THE LINK,COPYING
   (1)          001544                        MICPC=MICPC+1
   (1)             001                        .IF IDN MEMX,IMM
   (1)                                        <MOVE!LDMAR!IMM!<SELB!SPX!SP3&377>>
   (1)                                        .IFF
   (1)  015276  053223                        <MOVE!LDMAR!MEMX!<SELB!SPX!SP3>>
```

```
 (1)              000                     .ENDC
 (1)
1860                                                                    ;ITS ADDRESS TO SP0
1861  015300                             MEMINC  IMM,200                ;WRITE THE INTERRUPT TYPE
 (1)           001545                    MICPC=MICPC+1
 (1)  015300   016600                    <MOVE!WRMEM!INCMAR!IMM!<200>>
 (1)
1862  015302                             MEM     BR,INCA!SP0            ;COPY ACTUAL LINK ADDRESS
 (1)           001546                    MICPC=MICPC+1
 (1)  015302   062460                    <MOVE!WRMEM!BR!<INCA!SP0>>
 (1)
1863  015304                             LDMA    IMM,NXTSP              ;ADDRESS PTR INT STACK
 (1)           001547                    MICPC=MICPC+1
 (1)              001                    .IF IDN IMM,IMM
 (1)  015304   010241                    <MOVE!LDMAR!IMM!<NXTSP&377>>
 (1)                                     .IFF
 (1)                                     <MOVE!LDMAR!IMM!<NXTSP>>
 (1)              000                    .ENDC
 (1)
1864  015306                             MEM     IMM,INTSTK             ;ASSUME WRAP AROUND
 (1)           001550                    MICPC=MICPC+1
 (1)  015306   002642                    <MOVE!WRMEM!IMM!<INTSTK>>
 (1)
1865  015310                             BRWRTE  IMM,<<MMEND-2>>        ;ADDRESS ENDOFINT STACK
 (1)           001551                    MICPC=MICPC+1
 (1)  015310   000776                    <MOVE!WRTEBR!IMM!<<MMEND-2>>>
 (1)
1866  015312                             CMP     BR,SP3                 ;WRAPAROUND?
 (1)           001552                    MICPC=MICPC+1
 (1)  015312   060363                    <SUBTC!BR!SP3>
 (1)
1867  015314                             Z       TDON40                 ;YES===BRANCH
 (1)           001553                    MICPC=MICPC+1
 (1)  015314   115556                    <JUMP!ZCOND!<TDON40-INIT&3000*4>!1<TDON40-INIT&777/2>>
 (1)
1868  015316                             BRWRTE  IMM,2                  ;OFFSET TO NEXT PAIR
 (1)           001554                    MICPC=MICPC+1
 (1)  015316   000402                    <MOVE!WRTEBR!IMM!<2>>
 (1)
1869  015320                             MEM     BR,ADD!SP3             ;UPDATE POINTER
 (1)           001555                    MICPC=MICPC+1
 (1)  015320   062403                    <MOVE!WRMEM!BR!<ADD!SP3>>
 (1)
1870  015322                       TDON40: BRWRTE  IMM,20                 ;WRITE INTERUPT PENDING
 (1)           001556                    MICPC=MICPC+1
 (1)  015322   000420                    <MOVE!WRTEBR!IMM!<20>>
 (1)
1871  015324                             SP      BR,AORB,SP1            ;IN PORT STATUS WORD
 (1)           001557                    MICPC=MICPC+1
 (1)  015324   063301                    <MOVE!SPX!BR!AORB!SP1>
 (1)
1872  015326                             LDMA    IMM,ETC                ;ADDRESS NEXT EMPTY PTR
 (1)           001560                    MICPC=MICPC+1
 (1)              001                    .IF IDN IMM,IMM
 (1)  015326   010070                    <MOVE!LDMAR!IMM!<ETC&377>>
 (1)                                     .IFF
```



```
 (1)                                     <MOVE!LDMAR!IMM!<ETC>>
 (1)              000                    .ENDC
 (1)
1873  015330                             SP      MEMX,SELB,SP0          ;COPY IT TO SP0
 (1)           001561                    MICPC=MICPC+1
 (1)  015330   043220                    <MOVE!SPX!MEMX!SELB!SP0>
 (1)
1874  015332                             LDMA    IMM,STC                ;GET NEXT DONE PTR
 (1)           001562                    MICPC=MICPC+1
 (1)              001                    .IF IDN IMM,IMM
 (1)  015332   010067                    <MOVE!LDMAR!IMM!<STC&377>>
 (1)                                     .IFF
 (1)                                     <MOVE!LDMAR!IMM!<STC>>
 (1)              000                    .ENDC
 (1)
1875  015334                             CMP     MEMX,SP0               ;IDENTICAL?
 (1)           001563                    MICPC=MICPC+1
 (1)  015334   040360                    <SUBTC!MEMX!SP0>
 (1)
1876  015336                             Z       RH3                    ;FINISH PROCESSING HEADER
 (1)           001564                    MICPC=MICPC+1
 (1)  015336   105562                    <JUMP!ZCOND!<RH3-INIT&3000*4>!1<RH3-INIT&777/2>>
 (1)
1877
1878  015340                       TDON1:  LDMA    IMM,ISP17              ;GET LAST ACKED
 (1)           001565                    MICPC=MICPC+1
 (1)              001                    .IF IDN IMM,IMM
 (1)  015340   010153                    <MOVE!LDMAR!IMM!<ISP17&377>>
 (1)                                     .IFF
 (1)                                     <MOVE!LDMAR!IMM!<ISP17>>
 (1)              000                    .ENDC
 (1)
1879  015342                             SP      MEMX,SELB,SP17         ;STORE IT IN SP17
 (1)           001566                    MICPC=MICPC+1
 (1)  015342   043237                    <MOVE!SPX!MEMX!SELB!SP17>
 (1)
1880  015344                             LDMA    IMM,STC                ;GET START OF TMT CHAIN
 (1)           001567                    MICPC=MICPC+1
 (1)              001                    .IF IDN IMM,IMM
 (1)  015344   010067                    <MOVE!LDMAR!IMM!<STC&377>>
 (1)                                     .IFF
 (1)                                     <MOVE!LDMAR!IMM!<STC>>
 (1)              000                    .ENDC
 (1)
1881  015346                             LDMA    MEMX,SELB!SPBRX!SP0    ;ADDRESS THE LINK
 (1)           001570                    MICPC=MICPC+1
 (1)              001                    .IF IDN MEMX,IMM
 (1)                                     <MOVE!LDMAR!IMM!<SELB!SPBRX!SP0&377>>
 (1)                                     .IFF
 (1)  015346   053620                    <MOVE!LDMAR!MEMX!<SELB!SPBRX!SP0>>
 (1)              000                    .ENDC
 (1)
1882  015350                             BRWRTE  MEMX!INCMAR,SELB       ;GET THE FLAGS
 (1)           001571                    MICPC=MICPC+1
 (1)  015350   054620                    <MOVE!WRTEBR!MEMX!INCMAR!<SELB>>
 (1)
```

```
1883  015352                          BR1     TDON3                           ;IF BUFFER ASSIGNED PROCEED
 (1)          001572                  MICPC=MICPC+1
 (1)  015352  116530                  <JUMP!BR1CON!<TDON3-INIT&3000*4>!<TDON3-INIT&777/2>>
 (1)
1884  015354                          ALWAYS  RH3                             ;ELSE---EXIT
 (1)          001573                  MICPC=MICPC+1
 (1)  015354  104562                  <JUMP!ALCOND!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
 (1)
1885
1886  015356                    OVRRUN: BRWRTE  IMM,4
 (1)          001574                  MICPC=MICPC+1
 (1)  015356  000404                  <MOVE!WRTEBR!IMM!<4>>
 (1)
1887  015360                          ALWAYS  NTRS0
 (1)          001575                  MICPC=MICPC+1
 (1)  015360  114663                  <JUMP!ALCOND!<NTRS0-INIT&3000*4>!<NTRS0-INIT&777/2>>
 (1)
1888
1889                              ; INPUTS:
1890                              ;       SP0 = RECEIVE CHARACTER
1891
1892  015362                    PASWRD: SP      IBUS,LNOSW,SP13         ;READ PASSWD SWITCH
 (1)          001576                  MICPC=MICPC+1
 (1)  015362  023333                  <MOVE!SPX!IBUS!LNOSW!SP13>
 (1)
1893  015364                          Z       10$                     ;IF ALL ONES NO RLD ENABLED
 (1)          001577                  MICPC=MICPC+1
 (1)  015364  115603                  <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
 (1)
1894  015366                          BRWRTE  IMM,6                   ;CHECK FOR ENTER MOP MODE
 (1)          001600                  MICPC=MICPC+1
 (1)  015366  000406                  <MOVE!WRTEBR!IMM!<6>>
 (1)
1895  015370                          CMP     BR,SP0
 (1)          001601                  MICPC=MICPC+1
 (1)  015370  060360                  <SUBTC!BR!SP0>
 (1)
1896  015372                          Z       20$                     ;IF EQUAL   ENTER MOP
 (1)          001602                  MICPC=MICPC+1
 (1)  015372  115611                  <JUMP!ZCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
 (1)
1897  015374                    10$:  BRWRTE  BR,SELA!SP1             ;READ STATUS BYTE
 (1)          001603                  MICPC=MICPC+1
 (1)  015374  060601                  <MOVE!WRTEBR!BR!<SELA!SP1>>
 (1)
1898  015376                          BRSHFT                          ;SHIFT IT RIGHT
 (1)          001604                  MICPC=MICPC+1
 (1)  015376  001620                  <MOVE!SHFTBR!WRTEBR!SELB>
 (1)
1899  015400                          BR1     RHX                     ;MESSAGE WITH NO BUFFER ASSIGNED
 (1)          001605                  MICPC=MICPC+1
 (1)  015400  106740                  <JUMP!BR1CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
 (1)
1900  015402                          BRSHFT                          ;SHIFT RIGHT AGAIN
 (1)          001606                  MICPC=MICPC+1
 (1)  015402  001620                  <MOVE!SHFTBR!WRTEBR!SELB>
```



```
 (1)
1901  015404                          BR1     RCVM0                   ;DLE RECEIVED IN NORMAL MODE
 (1)          001607                  MICPC=MICPC+1
 (1)  015404  106732                  <JUMP!BR1CON!<RCVM0-INIT&3000*4>!<RCVM0-INIT&777/2>>
 (1)
1902  015406                          ALWAYS  RK3                     ;HANDLE MAINT MODE MESSAGE
 (1)          001610                  MICPC=MICPC+1
 (1)  015406  104641                  <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
 (1)
1903  015410                    20$:  SP      BR,DECA,SP4               ;COUNT FOR NUMB OF COMPARES
 (1)          001611                  MICPC=MICPC+1
 (1)  015410  063164                  <MOVE!SPX!BR!DECA!SP4>
 (1)
1904  015412                          STATE   EM2
 (1)          001612                  MICPC=MICPC+1
 (1)  015412  000735                  <MOVE!WRTEBR!IMM!<EM2-INIT&777/2>>
1905  015414                          ALWAYS  REXIT
 (1)          001613                  MICPC=MICPC+1
 (1)  015414  100450                  <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
 (1)
1906                              ;
1907                                      .ENABL  LSB
1908                              ;
1909  015416                    RCVM1:  LDMA    IMM,NAKST               ;RESET NAKS SENT
 (1)          001614                  MICPC=MICPC+1
 (1)             001                  .IF IDN IMM,IMM
 (1)  015416  010001                  <MOVE!LDMAR!IMM!<NAKST&377>>
 (1)                                  .IFF
 (1)                                  <MOVE!LDMAR!IMM!<NAKST>>
 (1)                 000              .ENDC
 (1)
1910  015420                          MEM     IMM,1                   ;..
 (1)          001615                  MICPC=MICPC+1
 (1)  015420  002401                  <MOVE!WRMEM!IMM!<1>>
 (1)
1911  015422                          LDMA    IMM,BC                  ;ADDRESS ORIGINAL RECV BYTE COUNT
 (1)          001616                  MICPC=MICPC+1
 (1)             001                  .IF IDN IMM,IMM
 (1)  015422  010167                  <MOVE!LDMAR!IMM!<BC&377>>
 (1)                                  .IFF
 (1)                                  <MOVE!LDMAR!IMM!<BC>>
 (1)                 000              .ENDC
 (1)
1912  015424                          SP      MEMX!INCMAR,SELB,SP4    ;MOVE BYTE COUNT TO SP4
 (1)          001617                  MICPC=MICPC+1
 (1)  015424  057224                  <MOVE!SPX!MEMX!INCMAR!SELB!SP4>
 (1)
1913  015426                          SP      MEMX!INCMAR,SELB,SP5    ;AND SP5
 (1)          001620                  MICPC=MICPC+1
 (1)  015426  057225                  <MOVE!SPX!MEMX!INCMAR!SELB!SP5>
 (1)
1914  015430                          MEM     BR,DECA!SP11                    ;COPY SP11 FROM MEMORY
 (1)          001621                  MICPC=MICPC+1
 (1)  015430  062571                  <MOVE!WRMEM!BR!<DECA!SP11>>
 (1)
1915  015432                          LDMA    IMM,NXTSP
```

```
(1)          001622                     MICPC=MICPC+1
(1)             001                     .IF IDN IMM,IMM
(1)  015432  010241                     <MOVE!LDMAR!IMM!<NXTSP&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<NXTSP>>
(1)             000                     .ENDC
(1)
1916  015434                            SP      MEMX!LDMAR,SELB,SP3             ;COPY TO SP3
(1)          001623                     MICPC=MICPC+1
(1)  015434  053223                     <MOVE!SPX!MEMX!LDMAR!SELB!SP3>
(1)
1917  015436                            MEMINC  IMM,204                         ;RECEIVE DONE IMAGE
(1)          001624                     MICPC=MICPC+1
(1)  015436  016604                     <MOVE!WRMEM!INCMAR!IMM!<204>>
(1)
1918  015440                            MEM     BR!LDMAR,SELA!SP14              ;COPY LINK ADDRESS TO NEXT INTER
(1)          001625                     MICPC=MICPC+1
(1)  015440  072614                     <MOVE!WRMEM!BR!LDMAR!<SELA!SP14>>
(1)
1919  015442                            MEMINC  IMM,0                           ;ZERO THE FLAGS
(1)          001626                     MICPC=MICPC+1
(1)  015442  016400                     <MOVE!WRMEM!INCMAR!IMM!<0>>
(1)
1920  015444                            SP      IMM!INCMAR,SP0,300              ;WRITE A 300 TO SP0
(1)          001627                     MICPC=MICPC+1
(1)  015444  017300                     <MOVE!SPX!IMM!INCMAR!SP0!300>
(1)
1921  015446                            BRWRTE  IMM!INCMAR,2                    ;PREPARE TO ADDRESS NEXT
(1)          001630                     MICPC=MICPC+1
(1)  015446  014402                     <MOVE!WRTEBR!IMM!INCMAR!<2>>
(1)
1922                                                                            ;INTERRUPT STACK AND INCREMENT
1923                                                                            ;THE MAR
1924  015450                            MEM     MEMX,AANDB!SP0                  ;MASK OFF ORIGINAL HIGH BYTE
(1)          001631                     MICPC=MICPC+1
(1)  015450  042660                     <MOVE!WRMEM!MEMX!<AANDB!SP0>>
(1)
1925                                                                            ;OF COUNT SAVING EXTENDED MEM BITS
1926  015452                            MEMINC  MEMX,AORB!SP5                   ;COPY TO MEMORY LINK
(1)          001632                     MICPC=MICPC+1
(1)  015452  056705                     <MOVE!WRMEM!INCMAR!MEMX!<AORB!SP5>>
(1)
1927  015454                            MEMINC  BR,SELA!SP4
(1)          001633                     MICPC=MICPC+1
(1)  015454  076604                     <MOVE!WRMEM!INCMAR!BR!<SELA!SP4>>
(1)
1928  015456                            LDMA    IMM,NXTSP                       ;ADDRESS NEXT INT STACK
(1)          001634                     MICPC=MICPC+1
(1)             001                     .IF IDN IMM,IMM
(1)  015456  010241                     <MOVE!LDMAR!IMM!<NXTSP&377>>
(1)                                     .IFF
(1)                                     <MOVE!LDMAR!IMM!<NXTSP>>
(1)             000                     .ENDC
(1)
1929  015460                            MEM     BR,ADD!SP3
(1)          001635                     MICPC=MICPC+1
```



```
(1)  015460  062403                     <MOVE!WRMEM!BR!<ADD!SP3>>
(1)
1930  015462                            BRWRTE  IMM,<<MMEND-2>>                 ;ADDRESSEND OF INT STACK
(1)          001636                     MICPC=MICPC+1
(1)  015462  000776                     <MOVE!WRTEBR!IMM!<<MMEND-2>>>
(1)
1931  015464                            CMP     BR,SP3                          ;WRAP AROUND
(1)          001637                     MICPC=MICPC+1
(1)  015464  060363                     <SUBTC!BR!SP3>
(1)
1932  015466                            Z       40$                             ;IF YES-- BRANCH
(1)          001640                     MICPC=MICPC+1
(1)  015466  115551                     <JUMP!ZCOND!<40$-INIT&3000*4>!<40$-INIT&777/2>>
(1)
1933  015470                      20$:
1934  015470                            BRWRTE  IMM,5                           ;INDEX TO NEXT BUFFER
(1)          001641                     MICPC=MICPC+1
(1)  015470  000405                     <MOVE!WRTEBR!IMM!<5>>
(1)
1935  015472                            SP      BR,ADD,SP14                     ;UPDATE COPY OF POINTER
(1)          001642                     MICPC=MICPC+1
(1)  015472  063014                     <MOVE!SPX!BR!ADD!SP14>
(1)
1936  015474                            BRWRTE  IMM,STC                         ;ADDRESS OF WRAP AROUND POINT
(1)          001643                     MICPC=MICPC+1
(1)  015474  000467                     <MOVE!WRTEBR!IMM!<STC>>
(1)
1937  015476                            CMP     BR,SP14                         ;WRAPAROUND?
(1)          001644                     MICPC=MICPC+1
(1)  015476  060374                     <SUBTC!BR!SP14>
(1)
1938  015500                            Z       50$                             ;IF YES---BRANCH
(1)          001645                     MICPC=MICPC+1
(1)  015500  115653                     <JUMP!ZCOND!<50$-INIT&3000*4>!<50$-INIT&777/2>>
(1)
1939  015502                      30$:  BRWRTE  IMM,20                          ;MASK FOR INTERUPT PENDING
(1)          001646                     MICPC=MICPC+1
(1)  015502  000420                     <MOVE!WRTEBR!IMM!<20>>
(1)
1940  015504                            SP      DP,AORB,SP1                     ;UPDATE PORT STATUS WORD
(1)          001647                     MICPC=MICPC+1
(1)  015504  063301                     <MOVE!SPX!DP!AORB!SP1>
(1)
1941             001                     .IF DF  $LOW
1942                                     BRWRTE  DP,<SELA!SP10>                  ;READ LINE STATUS WORD
1943                                     BR4     FLUSH                           ;IF CLEAR ACTIVE SET---FLUSH
1944                               RM1:  STATE   RCVA
1945                                     ALWAYS  REXIT
1946             000                     .ENDC
1947             001                     .IF NDF $LOW
1948  015506                            ALWAYS  FLUSH
(1)          001650                     MICPC=MICPC+1
(1)  015506  104415                     <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
(1)
1949             000                     .ENDC
```

```
   (1)          001651                          MICPC=MICPC+1
   (1)  015510  002642                          <MOVE!WRMEM!IMM!<INTSTK>>
   (1)
1951    015512                                   ALWAYS  20$
   (1)          001652                          MICPC=MICPC+1
   (1)  015512  114641                          <JUMP!ALCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
   (1)
1952    015514                          50$:     BRWRTE  IMM,RCL1                ;POINT TO START OF RECEIVE QUEUE
   (1)          001653                          MICPC=MICPC+1
   (1)  015514  000424                          <MOVE!WRTEBR!IMM!<RCL1>>
   (1)
1953    015516                                   SP      BR,SELB,SP14
   (1)          001654                          MICPC=MICPC+1
   (1)  015516  063234                          <MOVE!SPX!BR!SELB!SP14>
   (1)
1954    015520                                   ALWAYS  30$
   (1)          001655                          MICPC=MICPC+1
   (1)  015520  114646                          <JUMP!ALCOND!<30$-INIT&3000*4>!<30$-INIT&777/2>>
   (1)
1955                                             ,DSABL  LSB
1956    015522                          NTHRES:  LDMA    IMM,ST
   (1)          001656                          MICPC=MICPC+1
   (1)             001                          ,IF IDN IMM,IMM
   (1)  015522  010152                          <MOVE!LDMAR!IMM!<ST&377>>
   (1)                                          ,IFF
   (1)                                          <MOVE!LDMAR!IMM!<ST>>
   (1)             000                          ,ENDC
   (1)
1957    015524                                   SPBR    MEMX,SELB,SP0
   (1)          001657                          MICPC=MICPC+1
   (1)  015524  043620                          <MOVE!SPBRX!MEMX!SELB!SP0>
   (1)
1958    015526                                   BRWRTE  BR,ADD!SP0                      ;SHIFT LEFT
   (1)          001660                          MICPC=MICPC+1
   (1)  015526  060400                          <MOVE!WRTEBR!BR!<ADD!SP0>>
   (1)
1959    015530                                   BR4     OVRRUN
   (1)          001661                          MICPC=MICPC+1
   (1)  015530  117174                          <JUMP!BR4CON!<OVRRUN-INIT&3000*4>!<OVRRUN-INIT&777/2>>
   (1)
1960    015532                                   BRWRTE  IMM,1
   (1)          001662                          MICPC=MICPC+1
   (1)  015532  000401                          <MOVE!WRTEBR!IMM!<1>>
   (1)
1961    015534                          ERRXX:
1962    015534                          NTRS0:   LDMA    IMM,<<RTHRS+3>>
   (1)          001663                          MICPC=MICPC+1
   (1)             001                          ,IF IDN IMM,IMM
   (1)  015534  010177                          <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
   (1)                                          ,IFF
   (1)                                          <MOVE!LDMAR!IMM!<<RTHRS+3>>>
   (1)             000                          ,ENDC
   (1)
1963    015536                                   MEMINC  IMM,0
   (1)          001664                          MICPC=MICPC+1
   (1)  015536  016400                          <MOVE!WRMEM!INCMAR!IMM!<0>>
```



```
   (1)
1964    015540                                   MEM     BR,SELB
   (1)          001665                          MICPC=MICPC+1
   (1)  015540  062620                          <MOVE!WRMEM!BR!<SELB>>
   (1)
1965    015542                          NTRS1:   LDMA    IMM,NXTSP
   (1)          001666                          MICPC=MICPC+1
   (1)             001                          ,IF IDN IMM,IMM
   (1)  015542  010241                          <MOVE!LDMAR!IMM!<NXTSP&377>>
   (1)                                          ,IFF
   (1)                                          <MOVE!LDMAR!IMM!<NXTSP>>
   (1)             000                          ,ENDC
   (1)
1966    015544                                   LDMA    MEMX,SELB!SPX!SP0
   (1)          001667                          MICPC=MICPC+1
   (1)             001                          ,IF IDN MEMX,IMM
   (1)                                          <MOVE!LDMAR!IMM!<SELB!SPX!SP0&377>>
   (1)                                          ,IFF
   (1)  015544  053220                          <MOVE!LDMAR!MEMX!<SELB!SPX!SP0>>
   (1)             000                          ,ENDC
   (1)
1967    015546                                   MEMINC  IMM,201
   (1)          001670                          MICPC=MICPC+1
   (1)  015546  016601                          <MOVE!WRMEM!INCMAR!IMM!<201>>
   (1)
1968    015550                                   MEM     IMM,<<RTHRS>>
   (1)          001671                          MICPC=MICPC+1
   (1)  015550  002574                          <MOVE!WRMEM!IMM!<<RTHRS>>>
   (1)
1969    015552                                   LDMA    IMM,NXTSP
   (1)          001672                          MICPC=MICPC+1
   (1)             001                          ,IF IDN IMM,IMM
   (1)  015552  010241                          <MOVE!LDMAR!IMM!<NXTSP&377>>
   (1)                                          ,IFF
   (1)                                          <MOVE!LDMAR!IMM!<NXTSP>>
   (1)             000                          ,ENDC
   (1)
1970    015554                                   MEM     IMM,INTSTK                      ;ASSUME QUEUE WRAP AROUND
   (1)          001673                          MICPC=MICPC+1
   (1)  015554  002642                          <MOVE!WRMEM!IMM!<INTSTK>>
   (1)
1971    015556                                   BRWRTE  IMM,<<MMEND-2>>
   (1)          001674                          MICPC=MICPC+1
   (1)  015556  000776                          <MOVE!WRTEBR!IMM!<<MMEND-2>>>
   (1)
1972    015560                                   CMP     BR,SP0
   (1)          001675                          MICPC=MICPC+1
   (1)  015560  060360                          <SUBTC!BR!SP0>
   (1)
1973    015562                                   Z       NTRS2                           ;IT DID WRAP AROUND
   (1)          001676                          MICPC=MICPC+1
   (1)  015562  115701                          <JUMP!ZCOND!<NTRS2-INIT&3000*4>!<NTRS2-INIT&777/2>>
   (1)
1974    015564                                   BRWRTE  IMM,2                           ;OFFSET TO NEXT PAIR
   (1)          001677                          MICPC=MICPC+1
   (1)  015564  000102                          <MOVE!WRTEBR!IMM!<2>>
```

```
(1)          001622                         MICPC=MICPC+1
(1)             001                         .IF IDN IMM,IMM
(1)  015432  010241                         <MOVE!LDMAR!IMM!<NXTSP&377>>
(1)                                         .IFF
(1)                                         <MOVE!LDMAR!IMM!<NXTSP>>
(1)             000                         .ENDC
(1)
1916  015434                                SP      MEMX!LDMAR,SELB,SP3                 ;COPY TO SP3
(1)          001623                         MICPC=MICPC+1
(1)  015434  053223                         <MOVE!SPX!MEMX!LDMAR!SELB!SP3>
(1)
1917  015436                                MEMINC  IMM,204                             ;RECEIVE DONE IMAGE
(1)          001624                         MICPC=MICPC+1
(1)  015436  016604                         <MOVE!WRMEM!INCMAR!IMM!<204>>
(1)
1918  015440                                MEM     BR!LDMAR,SELA!SP14                  ;COPY LINK ADDRESS TO NEXT INTER
(1)          001625                         MICPC=MICPC+1
(1)  015440  072614                         <MOVE!WRMEM!BR!LDMAR!<SELA!SP14>>
(1)
1919  015442                                MEMINC  IMM,0                               ;ZERO THE FLAGS
(1)          001626                         MICPC=MICPC+1
(1)  015442  016400                         <MOVE!WRMEM!INCMAR!IMM!<0>>
(1)
1920  015444                                SP      IMM!INCMAR,SP0,300                  ;WRITE A 300 TO SP0
(1)          001627                         MICPC=MICPC+1
(1)  015444  017300                         <MOVE!SPX!IMM!INCMAR!SP0!300>
(1)
1921  015446                                BRWRTE  IMM!INCMAR,2                        ;PREPARE TO ADDRESS NEXT
(1)          001630                         MICPC=MICPC+1
(1)  015446  014402                         <MOVE!WRTEBR!IMM!INCMAR!<2>>
(1)
1922                                                                                    ;INTERRUPT STACK AND INCREMENT
1923                                                                                    ;THE MAR
1924  015450                                MEM     MEMX,AANDB!SP0                      ;MASK OFF ORIGINAL HIGH BYTE
(1)          001631                         MICPC=MICPC+1
(1)  015450  042660                         <MOVE!WRMEM!MEMX!<AANDB!SP0>>
(1)
1925                                                                                    ;OF COUNT SAVING EXTENDED MEM BITS
1926  015452                                MEMINC  MEMX,AORB!SP5                       ;COPY TO MEMORY LINK
(1)          001632                         MICPC=MICPC+1
(1)  015452  056705                         <MOVE!WRMEM!INCMAR!MEMX!<AORB!SP5>>
(1)
1927  015454                                MEMINC  BR,SELA!SP4
(1)          001633                         MICPC=MICPC+1
(1)  015454  076604                         <MOVE!WRMEM!INCMAR!BR!<SELA!SP4>>
(1)
1928  015456                                LDMA    IMM,NXTSP                           ;ADDRESS NEXT INT STACK
(1)          001634                         MICPC=MICPC+1
(1)             001                         .IF IDN IMM,IMM
(1)  015456  010241                         <MOVE!LDMAR!IMM!<NXTSP&377>>
(1)                                         .IFF
(1)                                         <MOVE!LDMAR!IMM!<NXTSP>>
(1)             000                         .ENDC
(1)
1929  015460                                MEM     BR,ADD!SP3
(1)          001635                         MICPC=MICPC+1
```



```
(1)  015460  062403                         <MOVE!WRMEM!BR!<ADD!SP3>>
(1)
1930  015462                                BRWRTE  IMM,<<MMEND-2>>                     ;ADDRESSEND OF INT STACK
(1)          001636                         MICPC=MICPC+1
(1)  015462  000776                         <MOVE!WRTEBR!IMM!<<MMEND-2>>>
(1)
1931  015464                                CMP     BR,SP3                              ;WRAP AROUND
(1)          001637                         MICPC=MICPC+1
(1)  015464  060363                         <SUBTC!BR!SP3>
(1)
1932  015466                                Z       40$                                 ;IF YES-- BRANCH
(1)          001640                         MICPC=MICPC+1
(1)  015466  115651                         <JUMP!ZCOND!<40$-INIT&3000*4>!<40$-INIT&777/2>>
(1)
1933  015470                          20$:
1934  015470                                BRWRTE  IMM,5                               ;INDEX TO NEXT BUFFER
(1)          001641                         MICPC=MICPC+1
(1)  015470  000405                         <MOVE!WRTEBR!IMM!<5>>
(1)
1935  015472                                SP      BR,ADD,SP14                         ;UPDATE COPY OF POINTER
(1)          001642                         MICPC=MICPC+1
(1)  015472  063014                         <MOVE!SPX!BR!ADD!SP14>
(1)
1936  015474                                BRWRTE  IMM,STC                             ;ADDRESS OF WRAP AROUND POINT
(1)          001643                         MICPC=MICPC+1
(1)  015474  000467                         <MOVE!WRTEBR!IMM!<STC>>
(1)
1937  015476                                CMP     BR,SP14                             ;WRAPAROUND?
(1)          001644                         MICPC=MICPC+1
(1)  015476  060374                         <SUBTC!BR!SP14>
(1)
1938  015500                                Z       50$                                 ;IF YES---BRANCH
(1)          001645                         MICPC=MICPC+1
(1)  015500  115653                         <JUMP!ZCOND!<50$-INIT&3000*4>!<50$-INIT&777/2>>
(1)
1939  015502                          30$:  BRWRTE  IMM,20                              ;MASK FOR INTERRUPT PENDING
(1)          001646                         MICPC=MICPC+1
(1)  015502  000420                         <MOVE!WRTEBR!IMM!<20>>
(1)
1940  015504                                SP      DP,AORB,SP1                         ;UPDATE PORT STATUS WORD
(1)          001647                         MICPC=MICPC+1
(1)  015504  063301                         <MOVE!SPX!DP!AORB!SP1>
(1)
1941             001                         .IF DF  $LOW
1942                                         BRWRTE  DP,<SELA!SP10>                      ;READ LINE STATUS WORD
1943                                         BR4     FLUSH                               ;IF CLEAR ACTIVE SET---FLUSH
1944                                 RM1:    STATE   RCVA
1945                                         ALWAYS  REXIT
1946             000                         .ENDC
1947             001                         .IF NDF $LOW
1948  015506                                 ALWAYS  FLUSH
(1)          001650                         MICPC=MICPC+1
(1)  015506  104415                         <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
(1)
1949             000                         .ENDC
```

```
   (1)          001651                          MICPC=MICPC+1
   (1)  015510  002642                          <MOVE!WRMEM!IMM!<INTSTK>>
   (1)
1951    015512                                  ALWAYS  20$
   (1)          001652                          MICPC=MICPC+1
   (1)  015512  114641                          <JUMP!ALCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
   (1)
1952    015514                          50$:    BRWRTE  IMM,RCL1                ;POINT TO START OF RECEIVE QUEUE
   (1)          001653                          MICPC=MICPC+1
   (1)  015514  000424                          <MOVE!WRTEBR!IMM!<RCL1>>
   (1)
1953    015516                                  SP      BR,SELB,SP14
   (1)          001654                          MICPC=MICPC+1
   (1)  015516  063234                          <MOVE!SPX!BR!SELB!SP14>
   (1)
1954    015520                                  ALWAYS  30$
   (1)          001655                          MICPC=MICPC+1
   (1)  015520  114646                          <JUMP!ALCOND!<30$-INIT&3000*4>!<30$-INIT&777/2>>
   (1)
1955                                            ,DSABL  LSB
1956    015522                          NTHRES: LDMA    IMM,ST
   (1)          001656                          MICPC=MICPC+1
   (1)             001                          ,IF IDN IMM,IMM
   (1)  015522  010152                          <MOVE!LDMAR!IMM!<ST&377>>
   (1)                                          ,IFF
   (1)                                          <MOVE!LDMAR!IMM!<ST>>
   (1)             000                          ,ENDC
   (1)
1957    015524                                  SPBR    MEMX,SELB,SP0
   (1)          001657                          MICPC=MICPC+1
   (1)  015524  043620                          <MOVE!SPBRX!MEMX!SELB!SP0>
   (1)
1958    015526                                  BRWRTE  BR,ADD!SP0                      ;SHIFT LEFT
   (1)          001660                          MICPC=MICPC+1
   (1)  015526  060400                          <MOVE!WRTEBR!BR!<ADD!SP0>>
   (1)
1959    015530                                  BR4     OVRRUN
   (1)          001661                          MICPC=MICPC+1
   (1)  015530  117174                          <JUMP!BR4CON!<OVRRUN-INIT&3000*4>!<OVRRUN-INIT&777/2>>
   (1)
1960    015532                                  BRWRTE  IMM,1
   (1)          001662                          MICPC=MICPC+1
   (1)  015532  000401                          <MOVE!WRTEBR!IMM!<1>>
   (1)
1961    015534                          ERRXX:
1962    015534                          NTRS0:  LDMA    IMM,<<RTHRS+3>>
   (1)          001663                          MICPC=MICPC+1
   (1)             001                          ,IF IDN IMM,IMM
   (1)  015534  010177                          <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
   (1)                                          ,IFF
   (1)                                          <MOVE!LDMAR!IMM!<<RTHRS+3>>>
   (1)             000                          ,ENDC
   (1)
1963    015536                                  MEMINC  IMM,0
   (1)          001664                          MICPC=MICPC+1
   (1)  015536  016400                          <MOVE!WRMEM!INCMAR!IMM!<0>>
```



```
   (1)
1964    015540                                  MEM     BR,SELB
   (1)          001665                          MICPC=MICPC+1
   (1)  015540  062620                          <MOVE!WRMEM!BR!<SELB>>
   (1)
1965    015542                          NTRS1:  LDMA    IMM,NXTSP
   (1)          001666                          MICPC=MICPC+1
   (1)             001                          ,IF IDN IMM,IMM
   (1)  015542  010241                          <MOVE!LDMAR!IMM!<NXTSP&377>>
   (1)                                          ,IFF
   (1)                                          <MOVE!LDMAR!IMM!<NXTSP>>
   (1)             000                          ,ENDC
   (1)
1966    015544                                  LDMA    MEMX,SELB!SPX!SP0
   (1)          001667                          MICPC=MICPC+1
   (1)             001                          ,IF IDN MEMX,IMM
   (1)                                          <MOVE!LDMAR!IMM!<SELB!SPX!SP0&377>>
   (1)                                          ,IFF
   (1)  015544  053220                          <MOVE!LDMAR!MEMX!<SELB!SPX!SP0>>
   (1)             000                          ,ENDC
   (1)
1967    015546                                  MEMINC  IMM,201
   (1)          001670                          MICPC=MICPC+1
   (1)  015546  016601                          <MOVE!WRMEM!INCMAR!IMM!<201>>
   (1)
1968    015550                                  MEM     IMM,<<RTHRS>>
   (1)          001671                          MICPC=MICPC+1
   (1)  015550  002574                          <MOVE!WRMEM!IMM!<<RTHRS>>>
   (1)
1969    015552                                  LDMA    IMM,NXTSP
   (1)          001672                          MICPC=MICPC+1
   (1)             001                          ,IF IDN IMM,IMM
   (1)  015552  010241                          <MOVE!LDMAR!IMM!<NXTSP&377>>
   (1)                                          ,IFF
   (1)                                          <MOVE!LDMAR!IMM!<NXTSP>>
   (1)             000                          ,ENDC
   (1)
1970    015554                                  MEM     IMM,INTSTK                      ;ASSUME QUEUE WRAP AROUND
   (1)          001673                          MICPC=MICPC+1
   (1)  015554  002642                          <MOVE!WRMEM!IMM!<INTSTK>>
   (1)
1971    015556                                  BRWRTE  IMM,<<MMEND-2>>
   (1)          001674                          MICPC=MICPC+1
   (1)  015556  000776                          <MOVE!WRTEBR!IMM!<<MMEND-2>>>
   (1)
1972    015560                                  CMP     BR,SP0
   (1)          001675                          MICPC=MICPC+1
   (1)  015560  060360                          <SUBTC!BR!SP0>
   (1)
1973    015562                                  Z       NTRS2                           ;IT DID WRAP AROUND
   (1)          001676                          MICPC=MICPC+1
   (1)  015562  115701                          <JUMP!ZCOND!<NTRS2-INIT&3000*4>!<NTRS2-INIT&777/2>>
   (1)
1974    015564                                  BRWRTE  IMM,2                           ;OFFSET TO NEXT PAIR
   (1)          001677                          MICPC=MICPC+1
   (1)  015564  000402                          <MOVE!WRTEBR!IMM!<2>>
```

```
(1)
1975  015566                        MEM     BR,ADD!SP0                      ;UPDATE QUEUE POINTER
(1)           001700                MICPC=MICPC+1
(1)   015566  062600                <MOVE!WRMEM!BR!<ADD!SP0>>
(1)
1976  015570                  NTRS2:  BRWRTE  IMM,20
(1)           001701                MICPC=MICPC+1
(1)   015570  000420                <MOVE!WRTEBR!IMM!<20>>
(1)
1977  015572                        SPBR    BR,AORB,SP1
(1)           001702                MICPC=MICPC+1
(1)   015572  063701                <MOVE!SPBRX!BR!AORB!SP1>
(1)
1978  015574                        BR0     TAB1                    ;FLAGGED BY ERROR TYPE
(1)           001703                MICPC=MICPC+1
(1)   015574  116334                <JUMP!BR0CON!<TAB1-INIT&3000*4>!<TAB1-INIT&777/2>>
(1)
1979  015576                  SNAK:   LDMA    IMM,ISP11
(1)           001704                MICPC=MICPC+1
(1)              001                .IF IDN IMM,IMM
(1)   015576  010171                <MOVE!LDMAR!IMM!<ISP11&377>>
(1)                                 .IFF
(1)                                 <MOVE!LDMAR!IMM!<ISP11>>
(1)              000                .ENDC
(1)
1980  015600                        SP      MEMX,SELB,SP11
(1)           001705                MICPC=MICPC+1
(1)   015600  043231                <MOVE!SPX!MEMX!SELB!SP11>
(1)
1981  015602                        SP      BR,INCA,SP11                    ;INCREMENT MSG EXPECTED
(1)           001706                MICPC=MICPC+1
(1)   015602  063071                <MOVE!SPX!BR!INCA!SP11>
(1)
1982  015604                  SNAK1:  BRWRTE  IMM,1                   ;UNNUMB PENING BIT TO BR
(1)           001707                MICPC=MICPC+1
(1)   015604  000401                <MOVE!WRTEBR!IMM!<1>>
(1)
1983  015606                  SNAK2:  SP      BR,AORB,SP10                    ;UPDATE LINE STATUS WORD
(1)           001710                MICPC=MICPC+1
(1)   015606  063310                <MOVE!SPX!BR!AORB!SP10>
(1)
1984  015610                        ALWAYS  FLUSH
(1)           001711                MICPC=MICPC+1
(1)   015610  104415                <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
(1)
1985                                ;
1986  015612                  EMTRIG: BRWRTE  IMM,24
(1)           001712                MICPC=MICPC+1
(1)   015612  000424                <MOVE!WRTEBR!IMM!<24>>
(1)
1987  015614                        OUTPUT  BR,<SELB!OBA1>
(1)           001713                MICPC=MICPC+1
(1)   015614  062226                <MOVE!WROUT!BR!<SELB!OBA1>>
(1)
1988  015616                        BRWRTE  IMM,0
(1)           001714                MICPC=MICPC+1
```



```
(1)   015616  000400                <MOVE!WRTEBR!IMM!<0>>
(1)
1989  015620                        OUTPUT  BR,<SELB!OBA2>
(1)           001715                MICPC=MICPC+1
(1)   015620  062227                <MOVE!WROUT!BR!<SELB!OBA2>>
(1)
1990  015622                        SPBR    IBUS,BM873,SP0          ;READ BM873 ADDRESS---
(1)           001716                MICPC=MICPC+1
(1)   015622  023740                <MOVE!SPBRX!IBUS!BM873!SP0>
(1)
1991  015624                        OUTPUT  BR,SELB!OUTDA1          ;SET UP LOW BYTE OF ADDRESS
(1)           001717                MICPC=MICPC+1
(1)   015624  062222                <MOVE!WROUT!BR!<SELB!OUTDA1>>
(1)
1992  015626                        BRWRTE  IMM,366                 ;HIGH BYTE BASE FOR ROM BOOT
(1)           001720                MICPC=MICPC+1
(1)   015626  000766                <MOVE!WRTEBR!IMM!<366>>
(1)
1993  015630                        OUTPUT  BR,SELB!OUTDA2          ;
(1)           001721                MICPC=MICPC+1
(1)   015630  062223                <MOVE!WROUT!BR!<SELB!OUTDA2>>
(1)
1994  015632                  EM6:    BRWRTE  IMM,21                  ;MASK FOR TIMER AND ALSO TO START NPR
(1)           001722                MICPC=MICPC+1
(1)   015632  000421                <MOVE!WRTEBR!IMM!<21>>
(1)
1995  015634                        OUT     BR,<SELB!OBR>
(1)           001723                MICPC=MICPC+1
(1)   015634  061231                <MOVE!WROUTX!BR!<SELB!OBR>>
(1)
1996  015636                        OUT     BR,<SELB!ONPR>
(1)           001724                MICPC=MICPC+1
(1)   015636  061230                <MOVE!WROUTX!BR!<SELB!ONPR>>
(1)
1997  015640                  EM1:    BRWRTE  IBUS,NPR                ;READ NPR CONTROL
(1)           001725                MICPC=MICPC+1
(1)   015640  120600                <MOVE!WRTEBR!IBUS!<NPR>>
(1)
1998  015642                        BR0     CKTIME
(1)           001726                MICPC=MICPC+1
(1)   015642  102371                <JUMP!BR0CON!<CKTIME-INIT&3000*4>!<CKTIME-INIT&777/2>>
(1)
1999  015644                        MEMADR  RM1                     ;IF NPR DONE
(1)           001727                MICPC=MICPC+1
(1)              001                .IF B
(1)   015644  002420                <MOVE!WRMEM!<RM1-INIT&777/2>>
(1)                                 .IFF
(1)                                 <MOVE!WRMEM!1!<RM1-INIT&777/2>>
(1)              000                .ENDC
2000  015646                        ALWAYS  ACLOW
(1)           001730                MICPC=MICPC+1
(1)   015646  100764                <JUMP!ALCOND!<ACLOW-INIT&3000*4>!<ACLOW-INIT&777/2>>
(1)
2001  015650                  TABUPD: SPBR    IBUS,RCVCON,SP0         ;READ RECEIVER CONTROL REG
(1)           001731                MICPC=MICPC+1
(1)   015650  023640                <MOVE!SPBRX!IBUS!RCVCON!SP0>
```

```
  (1)
2002  015552                          BRWRTE  BR,ADD!SP0              ;SHIFT LEFT
  (1)         001732                  MICPC=MICPC+1
  (1)  015652  060400                 <MOVE!WRTEBR!BR!<ADD!SP0>>
  (1)
2003  015654                          BR7     IDLE                    ;RECEIVE ACTIVE--IDLE
  (1)         001733                  MICPC=MICPC+1
  (1)  015654  103451                 <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
  (1)
2004  015656                    TAB1: LDMA    IMM,IMG10
  (1)         001734                  MICPC=MICPC+1
  (1)            001                  .IF IDN IMM,IMM
  (1)  015656  010154                 <MOVE!LDMAR!IMM!<IMG10&377>>
  (1)                                 .IFF
  (1)                                 <MOVE!LDMAR!IMM!<IMG10>>
  (1)            000                  .ENDC
  (1)
2005  015660                          BRWRTE  IMM,2
  (1)         001735                  MICPC=MICPC+1
  (1)  015660  000402                 <MOVE!WRTEBR!IMM!<2>>
  (1)
2006  015662                          MEMINC  BR,AANDB!SP10           ;SAVE BIT 1 OF SP10
  (1)         001736                  MICPC=MICPC+1
  (1)  015662  076670                 <MOVE!WRMEM!INCMAR!BR!<AANDB!SP10>>
  (1)
2007  015664                          MEMINC  BR,SELA!SP11            ;SAVE SP11
  (1)         001737                  MICPC=MICPC+1
  (1)  015664  076611                 <MOVE!WRMEM!INCMAR!BR!<SELA!SP11>>
  (1)
2008  015666                          MEMINC  BR,SELA!SP12            ;SAVE SP12
  (1)         001740                  MICPC=MICPC+1
  (1)  015666  076612                 <MOVE!WRMEM!INCMAR!BR!<SELA!SP12>>
  (1)
2009  015670                          MEMINC  BR,SELA!SP14            ;SAVE SP14
  (1)         001741                  MICPC=MICPC+1
  (1)  015670  076614                 <MOVE!WRMEM!INCMAR!BR!<SELA!SP14>>
  (1)
2010  015672                          MEMINC  BR,SELA!SP16            ;SAVE SP16
  (1)         001742                  MICPC=MICPC+1
  (1)  015672  076616                 <MOVE!WRMEM!INCMAR!BR!<SELA!SP16>>
  (1)
2011  015674                          MEMINC  BR,SELA!SP17            ;SAVE SP17
  (1)         001743                  MICPC=MICPC+1
  (1)  015674  076617                 <MOVE!WRMEM!INCMAR!BR!<SELA!SP17>>
  (1)
2012                                                                ;MAR NOW POINTS TO BASE
2013  015676                          STATE   RB2             ;DO NOT CHANGE BR UNTIL RB0
  (1)         001744                  MICPC=MICPC+1
  (1)  015676  000460                 <MOVE!WRTEBR!IMM!<RB2-INIT&777/2>>
2014  015700                          LDMA    IMM,TABST       ;POINT TO TABLE UPDATE STATE
  (1)         001745                  MICPC=MICPC+1
  (1)            001                  .IF IDN IMM,IMM
  (1)  015700  010210                 <MOVE!LDMAR!IMM!<TABST&377>>
  (1)                                 .IFF
  (1)                                 <MOVE!LDMAR!IMM!<TABST>>
  (1)            000                  .ENDC
```



```
  (1)
2015  015702                          PSTATE  TBU1                    ;NEW PORT STATE ADDRESS
  (1)  015702                         MEM     IMM,<<TBU1-INIT&777/2>>
  (2)         001746                  MICPC=MICPC+1
  (2)  015702  002774                 <MOVE!WRMEM!IMM!<<TBU1-INIT&777/2>>>
  (2)
2016  015704                          SP      IMM,4,SP4               ;INITIALIZE COUNT
  (1)         001747                  MICPC=MICPC+1
  (1)  015704  003004                 <MOVE!SPX!IMM!4!SP4>
  (1)
2017                                                                ;NOTE: FIRST 6 RAM LOCATIONS ARE NOT WRITTEN
2018                                                                ;TO CORE TABLE.
2019  015706                          LDMA    IMM,BASE
  (1)         001750                  MICPC=MICPC+1
  (1)            001                  .IF IDN IMM,IMM
  (1)  015706  010017                 <MOVE!LDMAR!IMM!<BASE&377>>
  (1)                                 .IFF
  (1)                                 <MOVE!LDMAR!IMM!<BASE>>
  (1)            000                  .ENDC
  (1)
2020  015710                          ALWAYS  RB0
  (1)         001751                  MICPC=MICPC+1
  (1)  015710  104442                 <JUMP!ALCOND!<RB0-INIT&3000*4>!<RB0-INIT&777/2>>
  (1)
2021  015712                    EC2:  BRWRTE  IMM,2                           ;INCREMENT COUNT/TEST
  (1)         001752                  MICPC=MICPC+1
  (1)  015712  000402                 <MOVE!WRTEBR!IMM!<2>>
  (1)
2022  015714                          SP      BR,ADD,SP4
  (1)         001753                  MICPC=MICPC+1
  (1)  015714  063004                 <MOVE!SPX!BR!ADD!SP4>
  (1)
2023  015716                          SP      IBUS,IOBA1,SP0                ;POINT TO NEXT ADDRESS
  (1)         001754                  MICPC=MICPC+1
  (1)  015716  023140                 <MOVE!SPX!IBUS!IOBA1!SP0>
  (1)
2024  015720                          OUTPUT  BR,ADD!OBA1
  (1)         001755                  MICPC=MICPC+1
  (1)  015720  062006                 <MOVE!WROUT!BR!<ADD!OBA1>>
  (1)
2025  015722                          SP      IBUS,IOBA2,SP0
  (1)         001756                  MICPC=MICPC+1
  (1)  015722  023160                 <MOVE!SPX!IBUS!IOBA2!SP0>
  (1)
2026  015724                          OUTPUT  BR,AC!OBA2
  (1)         001757                  MICPC=MICPC+1
  (1)  015724  062107                 <MOVE!WROUT!BR!<AC!OBA2>>
  (1)
2027  015726                          C       TABMXT
  (1)         001760                  MICPC=MICPC+1
  (1)  015726  105366                 <JUMP!CCOND!<TABMXT-INIT&3000*4>!<TABMXT-INIT&777/2>>
  (1)
2028  015730                    ECX:  BRWRTE  BR,SELA!SP1             ;READ PORT STATUS
  (1)         001761                  MICPC=MICPC+1
  (1)  015730  060601                 <MOVE!WRTEBR!BR!<SELA!SP1>>
  (1)
```

```
 2029  015732                          BR0     30$                     ;INIT MODE, WRITE OUT 200 BYTES
  (1)          001762                  MICPC=MICPC+1
  (1)  015732  116374                  <JUMP!BR0CON!<30$-INIT&3000*4>!<30$-INIT&777/2>>
  (1)
 2030                                                                  ;OTHERWISE ONLY WRITE OUT ERROR COUNTERS
 2031  015734                          BRWRTE  BR!LDMAR,SELA!SP4       ;READ COUNTER
  (1)          001763                  MICPC=MICPC+1
  (1)  015734  070604                  <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
  (1)
 2032  015736                          BR4     20$                     ;ALL DONE
  (1)          001764                  MICPC=MICPC+1
  (1)  015736  117371                  <JUMP!BR4CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
  (1)
 2033  015740                  10$:    OUTPUT  MEMX!INCMAR,SELB!OUTDA1         ;STORE COUNTS OF ERRORS
  (1)          001765                  MICPC=MICPC+1
  (1)  015740  056222                  <MOVE!WROUT!MEMX!INCMAR!<SELB!OUTDA1>>
  (1)
 2034  015742                          OUTPUT  MEMX!INCMAR,SELB!OUTDA2
  (1)          001766                  MICPC=MICPC+1
  (1)  015742  056223                  <MOVE!WROUT!MEMX!INCMAR!<SELB!OUTDA2>>
  (1)
 2035             001                  .IF NDF SLOW
 2036  015744                          SP      IBUS.NPR,SP0
  (1)          001767                  MICPC=MICPC+1
  (1)  015744  123200                  <MOVE!SPX!IBUS!NPR!SP0>
  (1)
 2037             000                  .ENDC
 2038  015746                          ALWAYS  RK8
  (1)          001770                  MICPC=MICPC+1
  (1)  015746  104622                  <JUMP!ALCOND!<RK8-INIT&3000*4>!<RK8-INIT&777/2>>
  (1)
 2039  015750                  20$:    LDMA    IMM,TABST
  (1)          001771                  MICPC=MICPC+1
  (1)             001                  .IF IDN IMM,IMM
  (1)  015750  010210                  <MOVE!LDMAR!IMM!<TABST&377>>
  (1)                                  .IFF
  (1)                                  <MOVE!LDMAR!IMM!<TABST>>
  (1)             000                  .ENDC
  (1)
 2040  015752                          PSTATE  I3
  (1)  015752                          MEM     IMM,<<I3-INIT&777/2>>
  (2)          001772                  MICPC=MICPC+1
  (2)  015752  002460                  <MOVE!WRMEM!IMM!<<I3-INIT&777/2>>>
  (2)
 2041  015754                          ALWAYS  RM1
  (1)          001773                  MICPC=MICPC+1
  (1)  015754  104420                  <JUMP!ALCOND!<RM1-INIT&3000*4>!<RM1-INIT&777/2>>
  (1)
 2042  015756                  30$:    BRWRTE  BR!LDMAR,SELA!SP4       ;READ COUNTER
  (1)          001774                  MICPC=MICPC+1
  (1)  015756  070604                  <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
  (1)
 2043  015760                          BR7     20$                     ;ALL DONE
  (1)          001775                  MICPC=MICPC+1
  (1)  015760  117771                  <JUMP!BR7CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
  (1)
```



```
 2044  015762                          ALWAYS  10$                     ;KEEP GOING
  (1)          001776                  MICPC=MICPC+1
  (1)  015762  114765                  <JUMP!ALCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
  (1)
 2045  015764                          $ZERO
  (1)          001777                  MICPC=MICPC+1
  (1)  015764  000000                  000000
  (1)
 2046                                  ;
 2047          000001                  .END


. ABS.  015766     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 ,DDCMP/CRF/DS:CRF_DMCHGH,HILOW,DDCHGH
 RUN-TIME:  16  31  .1 SECONDS
 RUN-TIME RATIO:  189/48=3.8
 CORE USED:  7K  (13 PAGES)


 1623                            00200
 1624
 1625
 1626                                  ;*************************** TEST 1 ***************************
 1627                                  ;*TEST OF BR RIGHT SHIFT
 1628                                  ;*VERIFY THAT A DEST OF BR RSH (011) OF A MICRO-INSTRUCTION
 1629                                  ;*SHIFTS THE RESULTING BR DATA RIGHT ONCE.
 1630                                  ;;*************************************************************
 1631
 1632                                  ;  TEST 1
 1633                                  ;--------------
 1634  015766  012737  000001  001226  TST1:   MOV     #1,TSTNO
 1635  015774  012737  016100  001216          MOV     #TST2,NEXT
 1636                                                          ;R1 CONTAINS BASE DMC11 ADDRESS
 1637  016002  104412                         MSTCLR          ;MASTER CLEAR DMC11
```

```
1638  016004  013701  001404                        MOV     DMCSR,R1        ;R1 = DMC BASE ADDRESS
1639  016010  005011                                CLR     (R1)            ;CLEAR SELO
1640  016012  012705  052525                        MOV     #52525,R5       ;START WITH 125
1641  016016  010561  000004                        MOV     R5,4(R1)        ;PORT4_125
1642  016022  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1643  016024  120500                                120500                  ;BR _ PORT4
1644  016026  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1645  016030  061620                                061620                  ;BR RSH_BR, SHIFT BR RIGHT
1646  016032  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1647  016034  061225                                061225                  ;PORT5_BR
1648  016036  006005                                ROR     R5              ;R5 = "EXPECTED"
1649  016040  116104  000005                        MOVB    5(R1),R4        ;R4 = "FOUND"
1650  016044  120504                                CMPB    R5,R4           ;DID BR SHIFT RIGHT ONCE?
1651  016046  001401                                BEQ     1$              ;BR IF YES
1652  016050  104012                                HLT     12              ;BR RIGHT SHIFT ERROR
1653  016052                                1$:
1654  016052  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1655  016054  061620                                061620                  ;BR RSH_BR, SHFT BR RIGHT AGAIN
1656  016056  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1657  016060  061225                                061225                  ;PORT5_BR
1658  016062  006005                                ROR     R5              ;R5 = "EXPECTED"
1659  016064  116104  000005                        MOVB    5(R1),R4        ;R4 = "FOUND"
1660  016070  120504                                CMPB    R5,R4           ;DID BR SHIFT RIGHT?
1661  016072  001401                                BEQ     2$              ;BR IF YES
1662  016074  104012                                HLT     12              ;BR RIGHT SHIFT ERROR
1663  016076  104400                        2$:     SCOPE                   ;SCOPE THIS TEST
1664
1665
1666                                                ;*************************** TEST 2 ****************************
1667                                                ;*IOP CRAM WRITE/READ TEST
1668                                                ;*FLOAT A 1 THROUGH EACH CRAM LOCATION
1669                                                ;!************************************************************
1670
1671                                                ;  TEST 2
1672                                                ;--------------
1673  016100  012737  000002  001226        TST2:   MOV     #2,TSTNO
1674  016106  012737  016214  001216                MOV     #TST3,NEXT
1675  016114  012737  016140  001220                MOV     #3$,LOCK
1676                                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
1677  016122  032737  100000  001366                BIT     #BIT15,STAT1    ;DOES DMC HAVE CRAM?
1678  016130  001430                                BEQ     5$              ;SKIP TEST IF NO CRAM
1679  016132  005000                                CLR     R0              ;R0 = CRAM ADDRESS
1680  016134  012702  000001                1$:     MOV     #1,R2           ;R2 = WRITE DATA
1681  016140                                2$:
1682  016140  012711  002000                3$:     MOV     #BIT10,(R1)     ;SET ROMO
1683  016144  010061  000004                        MOV     R0,4(R1)        ;WRITE ADDRESS TO SEL4
1684  016150  010261  000006                        MOV     R2,6(R1)        ;LOAD SEL6 WITH WRITE DATA
1685  016154  052711  020000                        BIS     #BIT13,(R1)     ;WRITE SEL6 INTO CRAM
1686  016160  016104  000004                        MOV     4(R1),R4        ;READ CRAM INTO "FOUND"
1687  016164  020204                                CMP     R2,R4           ;IS DATA CORRECT?
1688  016166  001401                                BEQ     4$              ;BR IF OK
1689  016170  104001                                HLT     1               ;ERROR
1690  016172  104401                        4$:     SCOP1
1691  016174  000241                                CLC                     ;CLEAR CARRY
1692  016176  006102                                ROL     R2              ;SHIFT WRITE DATA
1693  016200  001357                                BNE     2$              ;BR IF NOT DONE THIS ADDRESS
```



```
1694  016202  005200                                INC     R0              ;BUMP TO NEXT CRAM ADDRESS
1695  016204  022700  002000                        CMP     #2000,R0        ;DONE YET?
1696  016210  001351                                BNE     1$              ;BR IF NO
1697  016212  104400                        5$:     SCOPE                   ;SCOPE THIS TEST
1698
1699
1700                                                ;*************************** TEST 3 ****************************
1701                                                ;*IOP CRAM WRITE/READ TEST
1702                                                ;*FLOAT A 0 THROUGH EACH CRAM LOCATION
1703                                                ;!************************************************************
1704
1705                                                ;  TEST 3
1706                                                ;--------------
1707  016214  012737  000003  001226        TST3:   MOV     #3,TSTNO
1708  016222  012737  016336  001216                MOV     #TST4,NEXT
1709  016230  012737  016260  001220                MOV     #3$,LOCK
1710                                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
1711  016236  104412                                MSTCLR                  ;MASTER CLEAR DMC11
1712  016240  032737  100000  001366                BIT     #BIT15,STAT1    ;DOES DMC HAVE CRAM?
1713  016246  001432                                BEQ     5$              ;SKIP TEST IF NO CRAM
1714  016250  005000                                CLR     R0              ;R0 = CRAM ADDRESS
1715  016252  012702  000001                1$:     MOV     #1,R2           ;R2 = WRITE DATA
1716  016256                                2$:
1717  016256  005102                                COM     R2              ;MAKE IT A FLOATING ZERO
1718  016260  012711  002000                3$:     MOV     #BIT10,(R1)     ;SET ROMO
1719  016264  010061  000004                        MOV     R0,4(R1)        ;WRITE ADDRESS TO SEL4
1720  016270  010261  000006                        MOV     R2,6(R1)        ;LOAD SEL6 WITH WRITE DATA
1721  016274  052711  020000                        BIS     #BIT13,(R1)     ;WRITE SEL6 INTO CRAM
1722  016300  016104  000004                        MOV     4(R1),R4        ;READ CRAM INTO "FOUND"
1723  016304  020204                                CMP     R2,R4           ;IS DATA CORRECT?
1724  016306  001401                                BEQ     4$              ;BR IF OK
1725  016310  104001                                HLT     1               ;ERROR
1726  016312  104401                        4$:     SCOP1
1727  016314  005102                                COM     R2              ;BACK TO FLOATING ONE
1728  016316  000241                                CLC                     ;CLEAR CARRY
1729  016320  006102                                ROL     R2              ;SHIFT WRITE DATA
1730  016322  001355                                BNE     2$              ;BR IF NOT DONE THIS ADDRESS
1731  016324  005200                                INC     R0              ;BUMP TO NEXT CRAM ADDRESS
1732  016326  022700  002000                        CMP     #2000,R0        ;DONE YET?
1733  016332  001347                                BNE     1$              ;BR IF NO
1734  016334  104400                        5$:     SCOPE                   ;SCOPE THIS TEST
1735
1736
1737                                                ;*************************** TEST 4 ****************************
1738                                                ;*IOP CRAM DUAL ADDRESSING TEST
1739                                                ;*WRITE EACH ADDRESS INTO ITSELF,READ EACH
1740                                                ;*ADDRESS TO VERIFY CORRECT ADDRESSING
1741                                                ;!************************************************************
1742
1743                                                ;  TEST 4
1744                                                ;--------------
1745  016336  012737  000004  001226        TST4:   MOV     #4,TSTNO
1746  016344  012737  016516  001216                MOV     #TST5,NEXT
1747  016352  012737  016374  001220                MOV     #1$,LOCK
1748                                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
1749  016360  104412                                MSTCLR                  ;MASTER CLEAR DMC11
```

```
1750  016362  032737  100000  001366          BIT     #BIT15,STAT1    ;DOES DMC HAVE CRAM?
1751  016370  001451                          BEQ     5$              ;SKIP TEST IF NO CRAM
1752  016372  005000                          CLR     R0              ;R0 =CRAM ADDRESS
1753  016374  010002                  1$:     MOV     R0,R2           ;SAVE R2 FOR TYPEOUT
1754  016376  012711  002000                  MOV     #BIT10,(R1)     ;SET ROMO
1755  016402  010061  000004                  MOV     R0,4(R1)        ;WRITE ADDRESS TO SEL4
1756  016406  010061  000006                  MOV     R0,6(R1)        ;LOAD SEL6 WITH WRITE DATA
1757  016412  052711  020000                  BIS     #BIT13,(R1)     ;WRITE CRAM
1758  016416  005061  000006                  CLR     6(R1)           ;CLEAR SEL 6
1759  016422  016104  000006                  MOV     6(R1),R4        ;SHOULD READ BACK OWN ADDRESS
1760  016426  020004                          CMP     R0,R4           ;IS DATA CORRECT?
1761  016430  001401                          BEQ     2$              ;BR IF YES
1762  016432  104001                          HLT     1               ;DATA ERROR
1763  016434  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
1764  016436  005200                          INC     R0              ;BUMP TO NEXT ADDRESS
1765  016440  022700  002000                  CMP     #2000,R0        ;DONE WRITING YET?
1766  016444  001353                          BNE     1$              ;BR IF NO
1767  016446  005000                          CLR     R0              ;RESTART AT ADDRESS 0
1768  016450  012737  016456  001220          MOV     #3$,LOCK        ;NEW SCOP1
1769  016456  010002                  3$:     MOV     R0,R2           ;SAVE R2 FOR TYPEOUT
1770  016460  012711  002000                  MOV     #BIT10,(R1)     ;SET ROMO
1771  016464  010061  000004                  MOV     R0,4(R1)        ;SEL4 = CRAM ADDRESS
1772  016470  016104  000006                  MOV     6(R1),R4        ;READ CRAM INTO "FOUND"
1773  016474  020004                          CMP     R0,R4           ;IS DATA CORRECT?
1774  016476  001401                          BEQ     4$              ;BR IF YES
1775  016500  104002                          HLT     2               ;DUAL ADDRESSING ERROR
1776  016502  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
1777  016504  005200                          INC     R0              ;BUMP TO NEXT ADDRESS
1778  016506  022700  002000                  CMP     #2000,R0        ;DONE WRITING YET?
1779  016512  001361                          BNE     3$              ;BR IF NO
1780  016514  104400                  5$:     SCOPE                   ;SCOPE THIS TEST
1781
1782
1783                                          ;*************************** TEST 5 ***************************
1784                                          ;*IOP CRAM READ TEST
1785                                          ;*THIS TEST WRITES THE CRAM WITH THE CROM MICRO-CODE MAP
1786                                          ;*THEN READS IT BACK AND COMPARES EACH ADDRESS WITH THE
1787                                          ;*DUPLICATE OF THE CROM MICRO-CODE.
1788                                          ;;*********************************************************
1789
1790                                          ;  TEST 5
1791                                          ;--------------
1792  016516  012737  000005  001226  TST5:   MOV     #5,TSTNO
1793  016524  012737  016636  001216          MOV     #TST6,NEXT
1794  016532  012737  016566  001220          MOV     #1$,LOCK
1795                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
1796  016540  104412                          MSTCLR                  ;MASTER CLEAR DMC11
1797  016542  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT RAM OR ROM
1798  016550  001431                          BEQ     3$              ;SKIP TEST IF CROM
1799  016552  005011                          CLR     (R1)            ;CLEAR RUN
1800  016554  004737  035602                  JSR     PC,WROM         ;WRITE CRAM WITH MAP
1801  016560  012700  011766                  MOV     #ROMMAP,R0      ;SOFTWARE POINTER TO CROM DUPLICATE
1802  016564  005002                          CLR     R2              ;R2 = CROM ADDRESS
1803  016566  010261  000004          1$:     MOV     R2,4(R1)        ;WRITE CROM ADDRESS TO SEL4
1804  016572  012711  002000                  MOV     #BIT10,(R1)     ;SET CROMO
1805  016576  011005                          MOV     (R0),R5         ;PUT "EXPECTED" IN R5
```



```
1806  016600  016104  000006                  MOV     6(R1),R4        ;PUT "FOUND" IN R4
1807  016604  020504                          CMP     R5,R4           ;COMPARE HARD ROM TO SOFT DUPLICATE
1808  016606  001401                          BEQ     2$              ;BR IF OK
1809  016610  104003                          HLT     3               ;CRAM READ ERROR!
1810  016612  005011                  2$:     CLR     (R1)            ;CLR  BIT10
1811  016614  005061  000006                  CLR     6(R1)           ;CLEAR SEL6
1812  016620  104401                          SCOP1                   ;LOOP TO 1$ IF SW09=1
1813  016622  005202                          INC     R2              ;INC TO NEXT CROM ADDRESS
1814  016624  005720                          TST     (R0)+           ;POP R0 BY 2
1815  016626  022702  002000                  CMP     #2000,R2        ;DONE 1K YET?
1816  016632  001355                          BNE     1$              ;BR IF NO
1817  016634  104400                  3$:     SCOPE                   ;SCOPE THIS TEST
1818
1819
1820                                          ;*************************** TEST 6 ***************************
1821                                          ;*IOP MAIN MEMORY TEST
1822                                          ;*FLOAT A 1 THROUGH ALL MAIN MEMORY LOCATIONS
1823                                          ;;*********************************************************
1824
1825                                          ;  TEST 6
1826                                          ;--------------
1827  016636  012737  000006  001226  TST6:   MOV     #6,TSTNO
1828  016644  012737  017024  001216          MOV     #TST7,NEXT
1829  016652  012737  016702  001220          MOV     #65$,LOCK
1830                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
1831  016660  104412                          MSTCLR                  ;MASTER CLEAR DMC11
1832  016662  032737  100000  001366          BIT     #BIT15,STAT1    ;IS THIS AN IOP?
1833  016670  001454                          BEQ     2$              ;SKIP TEST IF NO
1834  016672  005037  034704                  CLR     FLAG            ;START WITH ADDRESS 0
1835  016676  012700  000001          1$:     MOV     #1,R0           ;START WITH BIT 0
1836  016702  042737  000377  016734  65$:     BIC     #377,66$                ;CLEAR ADDRESS FIELD OF INSTRUCTION
1837  016710  042737  000003  016740          BIC     #3,68$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
1838  016716  153737  034704  016734          BISB    FLAG,66$                ;ADD ADDRESS TO INSTRUCTION
1839  016724  153737  034705  016740          BISB    FLAG+1,68$      ;ADD ADDRESS TO INSTRUCTION
1840  016732  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1841  016734  010000                  66$:    010000                  ;LOAD MAR LO WITH ADDRESS IN FLAG
1842  016736  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1843  016740  004000                  68$:    004000                  ;LOAD MAR HI
1844  016742  010061  000004                  MOV     R0,4(R1)        ;WRITE PATTERN IN PORT4
1845  016746  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1846  016750  122500                          122500                  ;MOVE PORT4 TO MEMORY
1847  016752  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1848  016754  040620                          040620                  ;MOVE MEMORY TO BR
1849  016756  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1850  016760  061225                          61225                   ;MOVE BR TO PORT5
1851  016762  010005                          MOV     R0,P5           ;PUT "EXPECTED" IN R5
1852  016764  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" IN R4
1853  016770  120504                          CMPB    R5,R4           ;DATA CORRECT?
1854  016772  001401                          BEQ     67$             ;BR IF YES
1855  016774  104010                          HLT     10              ;DATA ERROR
1856  016776  104401                  67$:    SCOP1                   ;SW09=1?
1857  017000  000241                          CLC                     ;CLEAR CARRY
1858  017002  106100                          ROLB    R0              ;SHIFT BIT IN R0
1859  017004  001336                          BNE     65$             ;DONE IF R0=0
1860  017006  005237  034704                  INC     FLAG            ;NEXT ADDRESS
1861  017012  022737  002000  034704          CMP     #2000,FLAG      ;LAST ADDRESS?
```

```
1862  017020  001326                          BNE     1$              ;BR IF NO
1863  017022  104400                  2$:     SCOPE                   ;SCOPE THIS TEST
1864
1865
1866                                          ;*************************** TEST 7 ***************************
1867                                          ;*IOP MAIN MEMORY TEST
1868                                          ;*FLOAT A 0 THROUGH ALL MAIN MEMORY LOCATIONS
1869                                          ;!*************************************************************
1870
1871                                          ;  TEST 7
1872                                          ;--------------
1873  017024  012737  000007  001226  TST7:   MOV     #7,TSTNO
1874  017032  012737  017216  001216          MOV     #TST10,NEXT
1875  017040  012737  017072  001220          MOV     #65$,LOCK
1876                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
1877  017046  104412                          MSTCLR                  ;MASTER CLEAR DMC11
1878  017050  032737  100000  001366          BIT     #BIT15,STAT1    ;IS THIS AN IOP?
1879  017056  001456                          BEQ     2$              ;SKIP TEST IF NO
1880  017060  005037  034704                  CLR     FLAG            ;START WITH ADDRESS 0
1881  017064  012700  000001          1$:     MOV     #1,R0           ;START WITH BIT 0
1882  017070  005100                  64$:    COM     R0              ;CHANGE TO FLOATING 0
1883  017072  042737  000377  017124  65$:    BIC     #377,66$                ;CLEAR ADDRESS FIELD OF INSTRUCTION
1884  017100  042737  000003  017130          BIC     #3,68$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
1885  017106  153737  034704  017124          BISB    FLAG,66$                ;ADD ADDRESS TO INSTRUCTION
1886  017114  153737  034705  017130          BISB    FLAG+1,68$      ;ADD ADDRESS TO INSTRUCTION
1887  017122  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1888  017124  010000                  66$:    010000                  ;LOAD MAR LO WITH ADDRESS IN FLAG
1889  017126  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1890  017130  004000                  68$:    004000                  ;LOAD MAR HI
1891  017132  010061  000004                  MOV     R0,4(R1)        ;WRITE PATTERN IN PORT4
1892  017136  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1893  017140  122500                          122500                  ;MOVE PORT4 TO MEMORY
1894  017142  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1895  017144  040620                          040620                  ;MOVE MEMORY TO BR
1896  017146  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1897  017150  061225                          61225                   ;MOVE BR TO PORT5
1898  017152  010005                          MOV     R0,R5           ;PUT "EXPECTED" IN R5
1899  017154  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" IN R4
1900  017160  120504                          CMPB    R5,R4           ;DATA CORRECT?
1901  017162  001401                          BEQ     67$             ;BR IF YES
1902  017164  104010                          HLT     10              ;DATA ERROR
1903  017166  104401                  67$:    SCOP1                   ;SW09=1?
1904  017170  005100                          COM     R0              ;CHANGE TO FLOATING 1
1905  017172  000241                          CLC                     ;CLEAR CARRY
1906  017174  106100                          ROLB    R0              ;SHIFT BIT IN R0
1907  017176  001334                          BNE     64$             ;DONE IF R0=0
1908  017200  005237  034704                  INC     FLAG            ;NEXT ADDRESS
1909  017204  022737  002000  034704          CMP     #2000,FLAG      ;LAST ADDRESS?
1910  017212  001324                          BNE     1$              ;BR IF NO
1911  017214  104400                  2$:     SCOPE                   ;SCOPE THIS TEST
1912
1913
1914                                          ;*************************** TEST 10 **************************
1915                                          ;*IOP MAIN MEMORY DUAL ADDRESSING TEST
1916                                          ;*LOAD EACH MEMORY LOCATION WITH ITS OWN ADDRESS
1917                                          ;*READ BACK EACH LOCATION TO VERIFY CORRECT ADDRESSING
```



```
1918                                          ;!*************************************************************
1919
1920                                          ;  TEST 10
1921                                          ;--------------
1922  017216  012737  000010  001226  TST10:  MOV     #10,TSTNO
1923  017224  012737  017516  001216          MOV     #TST11,NEXT
1924  017232  012737  017256  001220          MOV     #1$,LOCK
1925                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
1926  017240  104412                          MSTCLR                  ;MASTER CLEAR DMC11
1927  017242  032737  100000  001366          BIT     #BIT15,STAT1    ;IS THIS AN IOP?
1928  017250  001521                          BEQ     9$              ;SKIP TEST IF NO
1929  017252  005037  034704                  CLR     FLAG            ;START AT ADDRESS 0
1930  017256  013702  034704          1$:     MOV     FLAG,R2         ;PUT DATA IN R2
1931  017262  042737  000377  017314          BIC     #377,2$         ;CLEAR ADDRESS FIELD OF INSTRUCTION
1932  017270  042737  000003  017320          BIC     #3,7$           ;CLEAR ADDRESS FIELD OF INSTRUCTION
1933  017276  153737  034704  017314          BISB    FLAG,2$         ;ADD ADDRESS TO INSTRUCTION
1934  017304  153737  034705  017320          BISB    FLAG+1,7$       ;ADD ADDRESS TO INSTRUCTION
1935  017312  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1936  017314  010000                  2$:     010000                  ;LOAD MAR LO
1937  017316  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1938  017320  004000                  7$:     004000                  ;LOAD MAR HI
1939  017322  010261  000004                  MOV     R2,4(R1)
1940  017326  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1941  017330  122500                          122500                  ;MOVE PORT4 TO MEMORY
1942  017332  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1943  017334  040620                          040620                  ;MOVE MEMORY TO THE BR
1944  017336  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1945  017340  061225                          61225                   ;MOV BR TO PORT5
1946  017342  010205                          MOV     R2,R5           ;PUT "EXPECTED" IN R5
1947  017344  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" IN R4
1948  017350  120504                          CMPB    R5,R4           ;DATA CORRECT?
1949  017352  001401                          BEQ     3$              ;BR IF YES
1950  017354  104010                          HLT     10              ;DATA ERROR
1951  017356  104401                  3$:     SCOP1                   ;SW09=1?
1952  017360  005237  034704                  INC     FLAG            ;NEXT ADDRESS
1953  017364  022737  002000  034704          CMP     #2000,FLAG      ;LAST ADDRESS
1954  017372  001331                          BNE     1$              ;BR IF NO
1955  017374  012737  017406  001220          MOV     #4$,LOCK        ;NEW SCOPE 1
1956  017402  005037  034704                  CLR     FLAG            ;RESTART AT ADDRESS 0
1957  017406  013702  034704          4$:     MOV     FLAG,R2         ;PUT DATA IN R2
1958  017412  042737  000377  017444          BIC     #377,5$         ;CLEAR ADDRESS FIELD OF INSTRUCTION
1959  017420  042737  000003  017450          BIC     #3,8$           ;CLEAR ADDRESS FIELD OF INSTRUCTION
1960  017426  153737  034704  017444          BISB    FLAG,5$         ;ADD ADDRESS TO INSTRUCTION
1961  017434  153737  034705  017450          BISB    FLAG+1,8$       ;ADD ADDRESS TO INSTRUCTION
1962  017442  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1963  017444  010000                  5$:     010000                  ;LOAD THE MAR LO
1964  017446  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1965  017450  004000                  8$:     004000                  ;LOAD MAR HI
1966  017452  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1967  017454  040620                          040620                  ;MOVE MEMORY TO THE BR
1968  017456  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1969  017460  061225                          61225                   ;MOV BR TO PORT5
1970  017462  010205                          MOV     R2,R5           ;PUT "EXPECTED" IN R5
1971  017464  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" IN R4
1972  017470  120504                          CMPB    R5,R4           ;DATA CORRECT?
1973  017472  001401                          BEQ     6$              ;BR IF YES
```

```
1974  017474  104010                          HLT     10              ;ADDRESSING ERROR
1975  017476  104401                  6$:     SCOP1                   ;SW09=1?
1976  017500  005237  034704                  INC     FLAG            ;NEXT ADDRESS
1977  017504  022737  002000  034704          CMP     #2000,FLAG      ;IS IT THE LAST
1978  017512  001335                          BNE     4$              ;BR IF NO
1979  017514  104400                  9$:     SCOPE                   ;SCOPE THIS TEST
1980
1981
1982                                          ;*************************** TEST 11 ***************************
1983                                          ;*IOP MAR TEST
1984                                          ;*PERFORM DUAL ADDRESSING TEST
1985                                          ;*USING MAR AUTO-INC FEATURE
1986                                          ;!************************************************************
1987
1988                                          ;  TEST 11
1989                                          ;--------------
1990  017516  012737  000011  001226  TST11:  MOV     #11,TSTNO
1991  017524  012737  017632  001216          MOV     #TST12,NEXT
1992                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
1993  017532  104412                          MSTCLR                  ;MASTER CLEAR DMC11
1994  017534  032737  100000  001366          BIT     #BIT15,STAT1    ;IS THIS AN IOP?
1995  017542  001432                          BEQ     4$              ;SKIP TEST IF NO
1996  017544  005002                          CLR     R2              ;START WITH A ZERO
1997  017546  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1998  017550  010000                          010000                  ;LOAD MAR WITH A ZERO
1999  017552  010261  000004          1$:     MOV     R2,4(R1)        ;WRITE DATA TO PORT4
2000  017556  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2001  017560  136500                          136500                  ;MEM_PORT4, AUTO-INC MAR
2002  017562  005202                          INC     R2              ;INCREMENT DATA
2003  017564  022702  002000                  CMP     #2000,R2        ;DONE YET?
2004  017570  001370                          BNE     1$              ;BR IF NO
2005  017572  005002                          CLR     R2              ;RESTART WITH A ZERO
2006  017574  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2007  017576  010000                          010000                  ;LOAD MAR WITH A ZERO
2008  017600                          2$:
2009  017600  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2010  017602  055224                          055224                  ;MOVE MEM TO PORT4
2011  017604  010205                          MOV     R2,R5           ;PUT "EXPECTED" IN R5
2012  017606  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
2013  017612  120504                          CMPB    R5,R4           ;DATA CORRECT?
2014  017614  001401                          BEQ     3$              ;BR IF YES
2015  017616  104011                          HLT     11              ;MAR ERROR
2016  017620  005202                  3$:     INC     R2              ;NEXT ADDRESS
2017  017622  022702  002000                  CMP     #2000,R2        ;DONE YET?
2018  017626  001364                          BNE     2$              ;BR IF NO
2019  017630  104400                  4$:     SCOPE                   ;SCOPE THIS TEST
2020
2021
2022                                          ;*************************** TEST 12 ***************************
2023                                          ;*IOP (CRAM) ODT BITS TEST
2024                                          ;*LOAD MAR WITH A 0 INC MAR UNTIL IT OVERFLOWS (2000 TIMES)
2025                                          ;*VERIFY THAT IBUS* 10 BIT5 IS SET ONLY WHEN MAR BIT 8 IS A ONE
2026                                          ;*AND THAT IBUS* 10 BIT6 IS SET ON MAR OVERFLOW(2000)
2027                                          ;!************************************************************
2028
2029                                          ;  TEST 12
```



```
2030                                          ;--------------
2031  017632  012737  000012  001226  TST12:  MOV     #12,TSTNO
2032  017640  012737  020040  001216          MOV     #TST13,NEXT
2033  017646  012737  017674  001220          MOV     #1$,LOCK
2034                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
2035  017654  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2036  017656  032737  100000  001366          BIT     #BIT15,STAT1    ;IS THIS AN IOP?
2037  017664  001464                          BEQ     2$              ;SKIP TEST IF NO
2038  017666  005002                          CLR     R2              ;R2=SAME AS MAR CONTENTS
2039  017670  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2040  017672  010000                          010000                  ;MAR_0
2041  017674                          1$:
2042  017674  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2043  017676  121204                          121204                  ;PORT4=IBUS* 10
2044  017700  005005                          CLR     R5              ;R5="EXPECTED"
2045  017702  032702  000400                  BIT     #BIT8,R2        ;IS BIT8 SET IN MAR?
2046  017706  001402                          BEQ     .+6             ;BR IF NO
2047  017710  012705  000040                  MOV     #BIT5,R5        ;IF YES THEN SET BIT5
2048  017714  016104  000004                  MOV     4(R1),R4        ;R4="FOUND"
2049  017720  042704  177637                  BIC     #177637,R4      ;CLEAR UNWANTED BITS
2050  017724  020504                          CMP     R5,R4           ;BITS 5&6 SHOULD BE CLEAR
2051  017726  001401                          BEQ     .+4             ;BR IF OK
2052  017730  104007                          HLT     7               ;ERROR BITS 5&6 NOT CLEAR
2053  017732  104401                          SCOP1                   ;LOOP TO 11$ IF SW09=1
2054  017734  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2055  017736  014000                          014000                  ;INC MAR
2056  017740  005202                          INC     R2              ;BUMP MEM ADDRESS
2057  017742  022702  002000                  CMP     #2000,R2        ;OVERFLOWED YET?
2058  017746  001352                          BNE     1$              ;BR IF NO
2059  017750  005037  001220                  CLR     LOCK            ;NO MORE SCOP1
2060  017754  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2061  017756  121204                          121204                  ;PART4_IBUS* 10
2062  017760  012705  000100                  MOV     #BIT6,R5        ;R5="EXPECTED"
2063  017764  016104  000004                  MOV     4(R1),R4        ;R4="FOUND"
2064  017770  042704  177637                  BIC     #177637,R4      ;CLEAR UNWANTED BITS
2065  017774  020504                          CMP     R5,R4           ;BIT6 SHOULD BE SET
2066  017776  001401                          BEQ     .+4             ;BR IF OK
2067  020000  104007                          HLT     7               ;ERROR, BIT6 NOT SET
2068  020002  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2069  020004  010000                          010000                  ;MAR_0
2070  020006  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2071  020010  004000                          004000                  ;MAR HI_0
2072  020012  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2073  020014  121204                          121204                  ;PORT4_IBUS* 10
2074  020016  005005                          CLR     R5              ;R5="EXPECTED"
2075  020020  016104  000004                  MOV     4(R1),R4        ;R4="FOUND"
2076  020024  042704  177637                  BIC     #177637,R4      ;CLEAR UNWANTED BITS
2077  020030  020504                          CMP     R5,R4           ;BITS 5&6 SHOULD BE CLEAR
2078  020032  001401                          BEQ     .+4             ;BR IF OK
2079  020034  104007                          HLT     7               ;ERROR 5&6 NOT BOTH CLEAR
2080  020036  104400                  2$:     SCOPE                   ;SCOPE THIS TEST
2081
2082
2083                                          ;*************************** TEST 13 ***************************
2084                                          ;*CROM READ TEST
2085                                          ;*THIS TEST READS EACH ROM LOCATION AND COMPARES
```

```
2086                                                ;*IT TO A SOFTWARE DUPLICATE OF THE CROM, THIS TEST
2087                                                ;*ALSO TESTS THE JUMP(I) MICRO-PROCESSOR INSTRUCTION,
2088                                                ;!************************************************************
2089
2090                                                ;  TEST 13
2091                                                ;--------------
2092  020040  012737  000013  001226        TST13:  MOV     #13,TSTNO
2093  020046  012737  020230  001216                MOV     #TST14,NEXT
2094  020054  012737  020106  001220                MOV     #1$,LOCK
2095                                                                    ;R1 CONTAINS BASE DMC11 ADDRESS
2096  020062  104412                                MSTCLR                  ;MASTER CLEAR DMC11
2097  020064  032737  100000  001366                BIT     #BIT15,STAT1    ;IS IT RAM OR ROM
2098  020072  001055                                BNE     4$              ;SKIP TEST IF CRAM
2099  020074  005011                                CLR     (R1)            ;CLEAR RUN
2100  020076  012700  011766                        MOV     #ROMMAP,R0      ;R0 POINTS TO SOFTWARE ROM MAP
2101  020102  005002                                CLR     R2              ;R2 CONTAINS ROM ADDRESS BITS 0-7
2102  020104  005003                                CLR     R3              ;R3 CONTAINS ROM ADDRESS BITS 8&9 IN BITS 11&12
2103  020106  042737  014377  020126        1$:     BIC     #14377,2$       ;CLEAR ADDRESS FIELDS OF INSTRUCTION
2104  020114  050237  020126                        BIS     R2,2$           ;ADD BITS 0-7 TO INSTRUCTION
2105  020120  050337  020126                        BIS     R3,2$           ;ADD BITS 11&12 TO INSTRUCTION
2106  020124  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2107  020126  100400                        2$:     100400                  ;JUMP(I) TO ROM ADDRESS IN R2 & R3
2108  020130  012711  002000                        MOV     #BIT10,(R1)     ;SET ROMO
2109  020134  011005                                MOV     (R0),R5         ;PUT "EXPECTED" IN R5
2110  020136  016104  000006                        MOV     6(R1),R4        ;PUT "FOUND" IN R4
2111  020142  020504                                CMP     R5,R4           ;COMPARE ROM CONTENTS TO SOFT DUP
2112  020144  001414                                BEQ     3$              ;BR IF OK
2113  020146  010337  001252                        MOV     R3,TEMP3        ;PUT ROM ADDRESS IN TEMP3
2114  020152  000241                                CLC                     ;FOR ERROR TYPEOUT
2115  020154  006037  001252                        ROR     TEMP3
2116  020160  006037  001252                        ROR     TEMP3
2117  020164  006037  001252                        ROR     TEMP3
2118  020170  050237  001252                        BIS     R2,TEMP3        ;TEMP3 NOW CONTAINS CORRECT ADDRESS
2119  020174  104004                                HLT     4               ;ROM READ ERROR
2120  020176  104401                        3$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2121  020200  005720                                TST     (R0)+           ;BUMP SOFT POINTER
2122  020202  005202                                INC     R2              ;BUMP ROM ADDRESS
2123  020204  022702  000400                        CMP     #400,R2         ;IS R2 TO MAX YET?
2124  020210  001336                                BNE     1$              ;BR IF NO
2125  020212  005002                                CLR     R2              ;YES, RESET R2 TO 0
2126  020214  062703  004000                        ADD     #4000,R3        ;INC  TO NEXT PAGE OF ROM
2127  020220  022703  020000                        CMP     #20000,R3       ;DONE YET?
2128  020224  001330                                BNE     1$              ;BR IF NO
2129  020226  104400                        4$:     SCOPE                   ;SCOPE THIS TEST
2130
2131
2132                                                ;************************** TEST 14 **************************
2133                                                ;*CROM TEST OF JUMP(I) NEVER MICRO-PROCESSOR INSTRUCTION,
2134                                                ;*PERFORM THE JUMP INSTRUCTION
2135                                                ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2136                                                ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE),
2137                                                ;!************************************************************
2138
2139                                                ;  TEST 14
2140                                                ;--------------
2141  020230  012737  000014  001226        TST14:  MOV     #14,TSTNO
```



```
2142  020236  012737  020420  001216                MOV     #TST15,NEXT
2143  020244  012737  020264  001220                MOV     #1$,LOCK
2144                                                                    ;R1 CONTAINS BASE DMC11 ADDRESS
2145  020252  104412                                MSTCLR                  ;MASTER CLEAR DMC11
2146  020254  032737  100000  001366                BIT     #BIT15,STAT1    ;IS IT CRAM?
2147  020262  001055                                BNE     6$+2            ;SKIP TEST IF YES
2148  020264                                1$:
2149  020264  004737  035430                        JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2150  020270  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2151  020272  100400                                100400                  ;START AT ROM PC=0
2152  020274  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2153  020276  114377                                1143771<400*0>          ;JUMP TO ROM PC OF 1777
2154  020300  004737  035522                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2155  020304  000002                                2                       ;INDEX
2156  020306  020504                                CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2157  020310  001401                                BEQ     2$              ;BR IF YES
2158  020312  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
2159  020314  104401                        2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2160  020316  012737  020324  001220                MOV     #3$,LOCK        ;NEW SCOP1
2161  020324                                3$:
2162  020324  004737  035430                        JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2163  020330  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2164  020332  100403                                100403                  ;START AT ROM PC=3
2165  020334  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2166  020336  100000                                1000001<400*0>  ;JUMP TO ROM PC OF 0
2167  020340  004737  035522                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2168  020344  000010                                10                      ;INDEX
2169  020346  020504                                CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2170  020350  001401                                BEQ     4$              ;BR IF YES
2171  020352  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
2172  020354  104401                        4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2173  020356  012737  020364  001220                MOV     #5$,LOCK        ;NEW SCOP1
2174  020364                                5$:
2175  020364  004737  035430                        JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2176  020370  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2177  020372  100406                                100406                  ;START AT ROM PC=6
2178  020374  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2179  020376  104125                                1041251<400*0>          ;JUMP TO ROM PC OF 525
2180  020400  004737  035522                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2181  020404  000016                                16                      ;INDEX
2182  020406  020504                                CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2183  020410  001401                                BEQ     6$              ;BR IF YES
2184  020412  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
2185  020414  104401                        6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2186  020416  104400                                SCOPE                   ;SCOPE THIS TEST
2187
2188
2189                                                ;************************** TEST 15 **************************
2190                                                ;*CROM TEST OF JUMP(I) ALWAYS MICRO-PROCESSOR INSTRUCTION,
2191                                                ;*PERFORM THE JUMP INSTRUCTION
2192                                                ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2193                                                ;!************************************************************
2194
2195                                                ;  TEST 15
2196                                                ;--------------
2197  020420  012737  000015  001226        TST15:  MOV     #15,TSTNO
```

```
2198  020426  012737  020574  001216          MOV     #TST16,NEXT
2199  020434  012737  020454  001220          MOV     #1$,LOCK
2200                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
2201  020442  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2202  020444  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2203  020452  001047                          BNE     6$+2            ;SKIP TEST IF YES
2204  020454                                1$:
2205  020454  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2206  020456  100400                          100400                  ;START AT ROM PC=0
2207  020460  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2208  020462  114777                          114377!<400*1>          ;JUMP TO ROM PC OF 1777
2209  020464  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2210  020470  003776                          3776                    ;INDEX
2211  020472  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2212  020474  001401                          BEQ     2$              ;BR IF YES
2213  020476  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2214  020500  104401                    2$:   SCOP1                   ;LOOP TO 1$ IF SW09=1
2215  020502  012737  020510  001220          MOV     #3$,LOCK        ;NEW SCOP1
2216  020510                                3$:
2217  020510  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2218  020512  100403                          100403                  ;START AT ROM PC=3
2219  020514  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2220  020516  100400                          100000!<400*1>  ;JUMP TO ROM PC OF 0
2221  020520  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2222  020524  000000                          0                       ;INDEX
2223  020526  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2224  020530  001401                          BEQ     4$              ;BR IF YES
2225  020532  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2226  020534  104401                    4$:   SCOP1                   ;LOOP TO 3$ IF SW09=1
2227  020536  012737  020544  001220          MOV     #5$,LOCK        ;NEW SCOP1
2228  020544                                5$:
2229  020544  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2230  020546  100406                          100406                  ;START AT ROM PC=6
2231  020550  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2232  020552  104525                          1041251<400*1>          ;JUMP TO ROM PC OF 525
2233  020554  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2234  020560  001252                          1252                    ;INDEX
2235  020562  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2236  020564  001401                          BEQ     6$              ;BR IF YES
2237  020566  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2238  020570  104401                    6$:   SCOP1                   ;LOOP TO 5$ IF SW59=1
2239  020572  104400                          SCOPE                   ;SCOPE THIS TEST
2240
2241
2242                                          ;*************************** TEST 16 ***************************
2243                                          ;*CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
2244                                          ;*SET THE C BIT, PERFORM THE JUMP INSTRUCTION,
2245                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2246                                          ;!*************************************************************
2247
2248                                          ;  TEST 16
2249                                          ;--------------
2250  020574  012737  000016  001226    TST16:MOV     #16,TSTNO
2251  020602  012737  020764  001216          MOV     #TST17,NEXT
2252  020610  012737  020630  001220          MOV     #1$,LOCK
2253                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
```



```
2254  020616  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2255  020620  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2256  020626  001055                          BNE     6$+2            ;SKIP TEST IF YES
2257  020630                                1$:
2258  020630  004737  035476                  JSR     PC,SETC ;SET THE C BIT'
2259  020634  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2260  020636  100400                          100400                  ;START AT ROM PC=0
2261  020640  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2262  020642  115377                          114377!<400*2>          ;JUMP TO ROM PC OF 1777
2263  020644  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2264  020650  003776                          3776                    ;INDEX
2265  020652  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2266  020654  001401                          BEQ     2$              ;BR IF YES
2267  020656  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2268  020660  104401                    2$:   SCOP1                   ;LOOP TO 1$ IF SW09=1
2269  020662  012737  020670  001220          MOV     #3$,LOCK        ;NEW SCOP1
2270  020670                                3$:
2271  020670  004737  035476                  JSR     PC,SETC ;SET THE C BIT'
2272  020674  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2273  020676  100403                          100403                  ;START AT ROM PC=3
2274  020700  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2275  020702  101000                          100000!<400*2>  ;JUMP TO ROM PC OF 0
2276  020704  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2277  020710  000000                          0                       ;INDEX
2278  020712  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2279  020714  001401                          BEQ     4$              ;BR IF YES
2280  020716  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2281  020720  104401                    4$:   SCOP1                   ;LOOP TO 3$ IF SW09=1
2282  020722  012737  020730  001220          MOV     #5$,LOCK        ;NEW SCOP1
2283  020730                                5$:
2284  020730  004737  035476                  JSR     PC,SETC ;SET THE C BIT'
2285  020734  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2286  020736  100406                          100406                  ;START AT ROM PC=6
2287  020740  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2288  020742  105125                          104125!<400*2>          ;JUMP TO ROM PC OF 525
2289  020744  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2290  020750  001252                          1252                    ;INDEX
2291  020752  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2292  020754  001401                          BEQ     6$              ;BR IF YES
2293  020756  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2294  020760  104401                    6$:   SCOP1                   ;LOOP TO 5$ IF SW59=1
2295  020762  104400                          SCOPE                   ;SCOPE THIS TEST
2296
2297
2298                                          ;*************************** TEST 17 ***************************
2299                                          ;*CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
2300                                          ;*SET THE Z BIT, PERFORM THE JUMP INSTRUCTION,
2301                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2302                                          ;!*************************************************************
2303
2304                                          ;  TEST 17
2305                                          ;--------------
2306  020764  012737  000017  001226    TST17:MOV     #17,TSTNO
2307  020772  012737  021154  001216          MOV     #TST20,NEXT
2308  021000  012737  021020  001220          MOV     #1$,LOCK
2309                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
```

```
021006  104412                                  MSTCLR                  ;MASTER CLEAR DMC11
021010  032737  100000  001366                  BIT     #BIT15,STAT1    ;IS IT CRAM?
021016  001055                                  BNE     6$+2            ;SKIP TEST IF YES
021020                                  1$:
021020  004737  035514                          JSR     PC,SETZ ;SET THE Z BIT'
021024  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021026  100400                                  100400                  ;START AT ROM PC=0
021030  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021032  115777                                  114377!<400*3>          ;JUMP TO ROM PC OF 1777
021034  004737  035522                          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021040  003776                                  3776                    ;INDEX
021042  020504                                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021044  001401                                  BEQ     2$              ;BR IF YES
021046  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
021050  104401                          2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
021052  012737  021060  001220                  MOV     #3$,LOCK        ;NEW SCOP1
021060                                  3$:
021060  004737  035514                          JSR     PC,SETZ ;SET THE Z BIT'
021064  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021066  100403                                  100403                  ;START AT ROM PC=3
021070  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021072  101400                                  100000!<400*3>  ;JUMP TO ROM PC OF 0
021074  004737  035522                          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021100  000000                                  0                       ;INDEX
021102  020504                                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021104  001401                                  BEQ     4$              ;BR IF YES
021106  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
021110  104401                          4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
021112  012737  021120  001220                  MOV     #5$,LOCK        ;NEW SCOP1
021120                                  5$:
021120  004737  035514                          JSR     PC,SETZ ;SET THE Z BIT'
021124  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021126  100406                                  100406                  ;START AT ROM PC=6
021130  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021132  105525                                  104125!<400*3>          ;JUMP TO ROM PC OF 525
021134  004737  035522                          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021140  001252                                  1252                    ;INDEX
021142  020504                                  CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
021144  001401                                  BEQ     6$              ;BR IF YES
021146  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
021150  104401                          6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
021152  104400                                  SCOPE                   ;SCOPE THIS TEST


                                                ;************************** TEST 20 **************************
                                                ;*CROM TEST OF JUMP(I) ON BR0 SET MICRO-PROCESSOR INSTRUCTION,
                                                ;*SET THE BR0 BIT, PERFORM THE JUMP INSTRUCTION,
                                                ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
                                                ;;************************************************************

                                                ;  TEST 20
                                                ;--------------
021154  012737  000020  001226          TST20:  MOV     #20,TSTNO
021162  012737  021344  001216                  MOV     #TST21,NEXT
021170  012737  021210  001220                  MOV     #1$,LOCK
                                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
```



```
021176  104412                                  MSTCLR                  ;MASTER CLEAR DMC11
021200  032737  100000  001366                  BIT     #BIT15,STAT1    ;IS IT CRAM?
021206  001055                                  BNE     6$+2            ;SKIP TEST IF YES
021210                                  1$:
021210  004737  035446                          JSR     PC,SETBR0       ;SET THE BR0 BIT'
021214  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021216  100400                                  100400                  ;START AT ROM PC=0
021220  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021222  116377                                  114377!<400*4>          ;JUMP TO ROM PC OF 1777
021224  004737  035522                          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021230  003776                                  3776                    ;INDEX
021232  020504                                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021234  001401                                  BEQ     2$              ;BR IF YES
021236  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
021240  104401                          2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
021242  012737  021250  001220                  MOV     #3$,LOCK        ;NEW SCOP1
021250                                  3$:
021250  004737  035446                          JSR     PC,SETBR0       ;SET THE BR0 BIT'
021254  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021256  100403                                  100403                  ;START AT ROM PC=3
021260  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021262  102000                                  100000!<400*4>  ;JUMP TO ROM PC OF 0
021264  004737  035522                          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021270  000000                                  0                       ;INDEX
021272  020504                                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021274  001401                                  BEQ     4$              ;BR IF YES
021276  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
021300  104401                          4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
021302  012737  021310  001220                  MOV     #5$,LOCK        ;NEW SCOP1
021310                                  5$:
021310  004737  035446                          JSR     PC,SETBR0       ;SET THE BR0 BIT'
021314  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021316  100406                                  100406                  ;START AT ROM PC=6
021320  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021322  106125                                  104125!<400*4>          ;JUMP TO ROM PC OF 525
021324  004737  035522                          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021330  001252                                  1252                    ;INDEX
021332  020504                                  CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
021334  001401                                  BEQ     6$              ;BR IF YES
021336  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
021340  104401                          6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
021342  104400                                  SCOPE                   ;SCOPE THIS TEST


                                                ;************************** TEST 21 **************************
                                                ;*CROM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION,
                                                ;*SET THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
                                                ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
                                                ;;************************************************************

                                                ;  TEST 21
                                                ;--------------
021344  012737  000021  001226          TST21:  MOV     #21,TSTNO
021352  012737  021534  001216                  MOV     #TST22,NEXT
021360  012737  021400  001220                  MOV     #1$,LOCK
                                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
```

```
021366  104412                          MSTCLR                  ;MASTER CLEAR DMC11
021370  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
021376  001055                          BNE     6$+2            ;SKIP TEST IF YES
021400                          1$:
021400  004737  035454                  JSR     PC,SETBR1       ;SET THE BR1 BIT'
021404  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021406  100400                          100400                  ;START AT ROM PC=0
021410  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021412  116777                          1143771<400*5>          ;JUMP TO ROM PC OF 1777
021414  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021420  003776                          3776                    ;INDEX
021422  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021424  001401                          BEQ     2$              ;BR IF YES
021426  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
021430  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
021432  012737  021440  001220          MOV     #3$,LOCK        ;NEW SCOP1
021440                          3$:
021440  004737  035454                  JSR     PC,SETBR1       ;SET THE BR1 BIT'
021444  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021446  100403                          100403                  ;START AT ROM PC=3
021450  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021452  102400                          1000001<400*5>  ;JUMP TO ROM PC OF 0
021454  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021460  000000                          0                       ;INDEX
021462  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021464  001401                          BEQ     4$              ;BR IF YES
021466  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
021470  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
021472  012737  021500  001220          MOV     #5$,LOCK        ;NEW SCOP1
021500                          5$:
021500  004737  035454                  JSR     PC,SETBR1       ;SET THE BR1 BIT'
021504  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021506  100406                          100406                  ;START AT ROM PC=6
021510  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021512  106525                          1041251<400*5>          ;JUMP TO ROM PC OF 525
021514  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021520  001252                          1252                    ;INDEX
021522  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
021524  001401                          BEQ     6$              ;BR IF YES
021526  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
021530  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
021532  104400                          SCOPE                   ;SCOPE THIS TEST

                                        ;************************** TEST 22 **************************
                                        ;*CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION,
                                        ;*SET THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
                                        ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
                                        ;!*************************************************************

                                        ;  TEST 22
                                        ;--------------
021534  012737  000022  001225  TST22:  MOV     #22,TSTNO
021542  012737  021724  001216          MOV     #TST23,NEXT
021550  012737  021570  001220          MOV     #1$,LOCK
                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
```



```
021556  104412                          MSTCLR                  ;MASTER CLEAR DMC11
021560  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
021566  001055                          BNE     6$+2            ;SKIP TEST IF YES
021570                          1$:
021570  004737  035462                  JSR     PC,SETBR4       ;SET THE BR4 BIT'
021574  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021576  100400                          100400                  ;START AT ROM PC=0
021600  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021602  117377                          1143771<400*6>          ;JUMP TO ROM PC OF 1777
021604  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021610  003776                          3776                    ;INDEX
021612  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021614  001401                          BEQ     2$              ;BR IF YES
021616  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
021620  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
021622  012737  021630  001220          MOV     #3$,LOCK        ;NEW SCOP1
021630                          3$:
021630  004737  035462                  JSR     PC,SETBR4       ;SET THE BR4 BIT'
021634  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021636  100403                          100403                  ;START AT ROM PC=3
021640  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021642  103000                          1000001<400*6>  ;JUMP TO ROM PC OF 0
021644  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021650  000000                          0                       ;INDEX
021652  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021654  001401                          BEQ     4$              ;BR IF YES
021656  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
021660  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
021662  012737  021670  001220          MOV     #5$,LOCK        ;NEW SCOP1
021670                          5$:
021670  004737  035462                  JSR     PC,SETBR4       ;SET THE BR4 BIT'
021674  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021676  100406                          100406                  ;START AT ROM PC=6
021700  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021702  107125                          1041251<400*6>          ;JUMP TO ROM PC OF 525
021704  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021710  001252                          1252                    ;INDEX
021712  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
021714  001401                          BEQ     6$              ;BR IF YES
021716  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
021720  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
021722  104400                          SCOPE                   ;SCOPE THIS TEST

                                        ;************************** TEST 23 **************************
                                        ;*CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION,
                                        ;*SET THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
                                        ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
                                        ;!*************************************************************

                                        ;  TEST 23
                                        ;--------------
021724  012737  000023  001226  TST23:  MOV     #23,TSTNO
021732  012737  022114  001216          MOV     #TST24,NEXT
021740  012737  021760  001220          MOV     #1$,LOCK
                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
```

```
2534  021746  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2535  021750  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2536  021756  001055                          BNE     6$+2            ;SKIP TEST IF YES
2537  021760                          1$:
2538  021760  004737  035470                  JSR     PC,SETBR7       ;SET THE BR7 BIT'
2539  021764  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2540  021766  100400                          100400                  ;START AT ROM PC=0
2541  021770  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2542  021772  117777                          114377!<400*7>          ;JUMP TO ROM PC OF 1777
2543  021774  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2544  022000  003776                          3776                    ;INDEX
2545  022002  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2546  022004  001401                          BEQ     2$              ;BR IF YES
2547  022006  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2548  022010  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2549  022012  012737  022020  001220          MOV     #3$,LOCK        ;NEW SCOP1
2550  022020                          3$:
2551  022020  004737  035470                  JSR     PC,SETBR7       ;SET THE BR7 BIT'
2552  022024  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2553  022026  100403                          100403                  ;START AT ROM PC=3
2554  022030  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2555  022032  103400                          100000!<400*7>  ;JUMP TO ROM PC OF 0
2556  022034  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2557  022040  000000                          0                       ;INDEX
2558  022042  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2559  022044  001401                          BEQ     4$              ;BR IF YES
2560  022046  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2561  022050  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2562  022052  012737  022060  001220          MOV     #5$,LOCK        ;NEW SCOP1
2563  022060                          5$:
2564  022060  004737  035470                  JSR     PC,SETBR7       ;SET THE BR7 BIT'
2565  022064  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2566  022066  100406                          100406                  ;START AT ROM PC=6
2567  022070  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2568  022072  107525                          104125!<400*7>          ;JUMP TO ROM PC OF 525
2569  022074  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2570  022100  001252                          1252                    ;INDEX
2571  022102  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2572  022104  001401                          BEQ     6$              ;BR IF YES
2573  022106  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2574  022110  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2575  022112  104400                          SCOPE                   ;SCOPE THIS TEST
2576
2577
2578                                          ;************************** TEST 24 *************************
2579                                          ;*CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
2580                                          ;*CLEAR THE C BIT, PERFORM THE JUMP INSTRUCTION,
2581                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2582                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2583                                          ;!*********************************************************************
2584
2585                                          ;  TEST 24
2586                                          ;--------------
2587  022114  012737  000024  001226  TST24:  MOV     #24,TSTNO
2588  022122  012737  022304  001216          MOV     #TST25,NEXT
2589  022130  012737  022150  001220          MOV     #1$,LOCK
```



```
2590                                          ;R1 CONTAINS BASE DMC11 ADDRESS
2591  022136  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2592  022140  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2593  022146  001055                          BNE     6$+2            ;SKIP TEST IF YES
2594  022150                          1$:
2595  022150  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2596  022154  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2597  022156  100400                          100400                  ;START AT ROM PC=0
2598  022160  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2599  022162  115377                          114377!<400*2>          ;JUMP TO ROM PC OF 1777
2600  022164  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2601  022170  000002                          2                       ;INDEX
2602  022172  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2603  022174  001401                          BEQ     2$              ;BR IF YES
2604  022176  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2605  022200  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2606  022202  012737  022210  001220          MOV     #3$,LOCK        ;NEW SCOP1
2607  022210                          3$:
2608  022210  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2609  022214  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2610  022216  100403                          100403                  ;START AT ROM PC=3
2611  022220  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2612  022222  101000                          100000!<400*2>  ;JUMP TO ROM PC OF 0
2613  022224  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2614  022230  000010                          10                      ;INDEX
2615  022232  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2616  022234  001401                          BEQ     4$              ;BR IF YES
2617  022236  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2618  022240  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2619  022242  012737  022250  001220          MOV     #5$,LOCK        ;NEW SCOP1
2620  022250                          5$:
2621  022250  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2622  022254  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2623  022256  100406                          100406                  ;START AT ROM PC=6
2624  022260  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2625  022262  105125                          104125!<400*2>          ;JUMP TO ROM PC OF 525
2626  022264  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2627  022270  000016                          16                      ;INDEX
2628  022272  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2629  022274  001401                          BEQ     6$              ;BR IF YES
2630  022276  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2631  022300  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2632  022302  104400                          SCOPE                   ;SCOPE THIS TEST
2633
2634
2635                                          ;************************** TEST 25 *************************
2636                                          ;*CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
2637                                          ;*CLEAR THE Z BIT, PERFORM THE JUMP INSTRUCTION,
2638                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2639                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2640                                          ;!*********************************************************************
2641
2642                                          ;  TEST 25
2643                                          ;--------------
2644  022304  012737  000025  001226  TST25:  MOV     #25,TSTNO
2645  022312  012737  022474  001216          MOV     #TST26,NEXT
```

```
2646  022320  012737  022340  001220          MOV     #1$,LOCK
2647                                                          ;R1 CONTAINS BASE DMC11 ADDRESS
2648  022326  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2649  022330  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2650  022336  001055                          BNE     6$+2            ;SKIP TEST IF YES
2651  022340                          1$:
2652  022340  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2653  022344  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2654  022346  100400                          100400                  ;START AT ROM PC=0
2655  022350  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2656  022352  115777                          1143771!<400*3>         ;JUMP TO ROM PC OF 1777
2657  022354  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2658  022360  000002                          2                       ;INDEX
2659  022362  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2660  022364  001401                          BEQ     2$              ;BR IF YES
2661  022366  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2662  022370  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2663  022372  012737  022400  001220          MOV     #3$,LOCK        ;NEW SCOP1
2664  022400                          3$:
2665  022400  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2666  022404  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2667  022406  100403                          100403                  ;START AT ROM PC=3
2668  022410  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2669  022412  101400                          1000001<400*3>  ;JUMP TO ROM PC OF 0
2670  022414  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2671  022420  000010                          10                      ;INDEX
2672  022422  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2673  022424  001401                          BEQ     4$              ;BR IF YES
2674  022426  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2675  022430  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2676  022432  012737  022440  001220          MOV     #5$,LOCK        ;NEW SCOP1
2677  022440                          5$:
2678  022440  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2679  022444  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2680  022446  100406                          100406                  ;START AT ROM PC=6
2681  022450  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2682  022452  105525                          1041251<400*3>          ;JUMP TO ROM PC OF 525
2683  022454  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2684  022460  000016                          16                      ;INDEX
2685  022462  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2686  022464  001401                          BEQ     6$              ;BR IF YES
2687  022466  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2688  022470  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2689  022472  104400                          SCOPE                   ;SCOPE THIS TEST
2690
2691
2692                                          ;************************** TEST 26 **************************
2693                                          ;*CROM TEST OF JUMP(I) ON BR0 SET MICRO-PROCESSOR INSTRUCTION,
2694                                          ;*CLEAR THE BR0 BIT, PERFORM THE JUMP INSTRUCTION,
2695                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2696                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE),
2697                                          ;!************************************************************
2698
2699                                          ;  TEST 26
2700                                          ;--------------
2701  022474  012737  000026  001226  TST26:  MOV     #26,TSTNO
```



```
2702  022502  012737  022664  001216          MOV     #TST27,NEXT
2703  022510  012737  022530  001220          MOV     #1$,LOCK
2704                                                          ;R1 CONTAINS BASE DMC11 ADDRESS
2705  022516  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2706  022520  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2707  022526  001055                          BNE     6$+2            ;SKIP TEST IF YES
2708  022530                          1$:
2709  022530  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2710  022534  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2711  022536  100400                          100400                  ;START AT ROM PC=0
2712  022540  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2713  022542  116377                          1143771!<400*4>         ;JUMP TO ROM PC OF 1777
2714  022544  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2715  022550  000002                          2                       ;INDEX
2716  022552  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2717  022554  001401                          BEQ     2$              ;BR IF YES
2718  022556  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2719  022560  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2720  022562  012737  022570  001220          MOV     #3$,LOCK        ;NEW SCOP1
2721  022570                          3$:
2722  022570  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2723  022574  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2724  022576  100403                          100403                  ;START AT ROM PC=3
2725  022600  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2726  022602  102000                          1000001<400*4>  ;JUMP TO ROM PC OF 0
2727  022604  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2728  022610  000010                          10                      ;INDEX
2729  022612  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2730  022614  001401                          BEQ     4$              ;BR IF YES
2731  022616  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2732  022620  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2733  022622  012737  022630  001220          MOV     #5$,LOCK        ;NEW SCOP1
2734  022630                          5$:
2735  022630  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2736  022634  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2737  022636  100406                          100406                  ;START AT ROM PC=6
2738  022640  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2739  022642  106125                          1041251<400*4>          ;JUMP TO ROM PC OF 525
2740  022644  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2741  022650  000016                          16                      ;INDEX
2742  022652  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2743  022654  001401                          BEQ     6$              ;BR IF YES
2744  022656  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2745  022660  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2746  022662  104400                          SCOPE                   ;SCOPE THIS TEST
2747
2748
2749                                          ;************************** TEST 27 **************************
2750                                          ;*CROM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION,
2751                                          ;*CLEAR THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
2752                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2753                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE),
2754                                          ;!************************************************************
2755
2756                                          ;  TEST 27
2757                                          ;--------------
```

```
2758  022664  012737  000027  001226  TST27:  MOV     #27,TSTNO
2759  022672  012737  023054  001216          MOV     #TST30,NEXT
2760  022700  012737  022720  001220          MOV     #1$,LOCK
2761                                          ;R1 CONTAINS BASE DMC11 ADDRESS
2762  022706  104412                         MSTCLR                  ;MASTER CLEAR DMC11
2763  022710  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2764  022716  001055                         BNE     6$+2            ;SKIP TEST IF YES
2765  022720                          1$:
2766  022720  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2767  022724  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2768  022726  100400                         100400                  ;START AT ROM PC=0
2769  022730  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2770  022732  116777                         114377!<400*5>          ;JUMP TO ROM PC OF 1777
2771  022734  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2772  022740  000002                         2                       ;INDEX
2773  022742  020504                         CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2774  022744  001401                         BEQ     2$              ;BR IF YES
2775  022746  104006                         HLT     6               ;ERROR, CROM PC IS WRONG
2776  022750  104401                  2$:    SCOP1                   ;LOOP TO 1$ IF SW09=1
2777  022752  012737  022760  001220          MOV     #3$,LOCK        ;NEW SCOP1
2778  022760                          3$:
2779  022760  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2780  022764  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2781  022766  100403                         100403                  ;START AT ROM PC=3
2782  022770  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2783  022772  102400                         1000001<400*5>  ;JUMP TO ROM PC OF 0
2784  022774  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2785  023000  000010                         10                      ;INDEX
2786  023002  020504                         CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2787  023004  001401                         BEQ     4$              ;BR IF YES
2788  023006  104006                         HLT     6               ;ERROR, CROM PC IS WRONG
2789  023010  104401                  4$:    SCOP1                   ;LOOP TO 3$ IF SW09=1
2790  023012  012737  023020  001220          MOV     #5$,LOCK        ;NEW SCOP1
2791  023020                          5$:
2792  023020  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2793  023024  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2794  023026  100406                         100406                  ;START AT ROM PC=6
2795  023030  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2796  023032  106525                         1041251<400*5>          ;JUMP TO ROM PC OF 525
2797  023034  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2798  023040  000016                         16                      ;INDEX
2799  023042  020504                         CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2800  023044  001401                         BEQ     6$              ;BR IF YES
2801  023046  104006                         HLT     6               ;ERROR, CROM PC IS WRONG
2802  023050  104401                  6$:    SCOP1                   ;LOOP TO 5$ IF SW59=1
2803  023052  104400                         SCOPE                   ;SCOPE THIS TEST
2804
2805
2806                                          ;************************** TEST 30 **************************
2807                                          ;*CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION,
2808                                          ;*CLEAR THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
2809                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2810                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2811                                          ;!*************************************************************
2812
2813                                          ;  TEST 30
```



```
2814                                          ;--------------
2815  023054  012737  000030  001226  TST30:  MOV     #30,TSTNO
2816  023062  012737  023244  001216          MOV     #TST31,NEXT
2817  023070  012737  023110  001220          MOV     #1$,LOCK
2818                                          ;R1 CONTAINS BASE DMC11 ADDRESS
2819  023076  104412                         MSTCLR                  ;MASTER CLEAR DMC11
2820  023100  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2821  023106  001055                         BNE     6$+2            ;SKIP TEST IF YES
2822  023110                          1$:
2823  023110  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2824  023114  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2825  023116  100400                         100400                  ;START AT ROM PC=0
2826  023120  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2827  023122  117377                         114377!<400*6>          ;JUMP TO ROM PC OF 1777
2828  023124  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2829  023130  000002                         2                       ;INDEX
2830  023132  020504                         CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2831  023134  001401                         BEQ     2$              ;BR IF YES
2832  023136  104006                         HLT     6               ;ERROR, CROM PC IS WRONG
2833  023140  104401                  2$:    SCOP1                   ;LOOP TO 1$ IF SW09=1
2834  023142  012737  023150  001220          MOV     #3$,LOCK        ;NEW SCOP1
2835  023150                          3$:
2836  023150  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2837  023154  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2838  023156  100403                         100403                  ;START AT ROM PC=3
2839  023160  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2840  023162  103000                         1000001<400*6>  ;JUMP TO ROM PC OF 0
2841  023164  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2842  023170  000010                         10                      ;INDEX
2843  023172  020504                         CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2844  023174  001401                         BEQ     4$              ;BR IF YES
2845  023176  104006                         HLT     6               ;ERROR, CROM PC IS WRONG
2846  023200  104401                  4$:    SCOP1                   ;LOOP TO 3$ IF SW09=1
2847  023202  012737  023210  001220          MOV     #5$,LOCK        ;NEW SCOP1
2848  023210                          5$:
2849  023210  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2850  023214  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2851  023216  100406                         100406                  ;START AT ROM PC=6
2852  023220  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2853  023222  107125                         1041251<400*6>          ;JUMP TO ROM PC OF 525
2854  023224  004737  035522                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2855  023230  000016                         16                      ;INDEX
2856  023232  020504                         CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2857  023234  001401                         BEQ     6$              ;BR IF YES
2858  023236  104006                         HLT     6               ;ERROR, CROM PC IS WRONG
2859  023240  104401                  6$:    SCOP1                   ;LOOP TO 5$ IF SW59=1
2860  023242  104400                         SCOPE                   ;SCOPE THIS TEST
2861
2862
2863                                          ;************************** TEST 31 **************************
2864                                          ;*CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION,
2865                                          ;*CLEAR THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
2866                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2867                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2868                                          ;!*************************************************************
2869
```

```
                                                ; TEST 31
                                                ;--------------
023244  012737  000031  001226       TST31:     MOV     #31,TSTNO
023252  012737  023434  001216                  MOV     #TST32,NEXT
023260  012737  023300  001220                  MOV     #1$,LOCK
                                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
023266  104412                                  MSTCLR                  ;MASTER CLEAR DMC11
023270  032737  100000  001366                  BIT     #BIT15,STAT1    ;IS IT CRAM?
023276  001055                                  BNE     6$+2            ;SKIP TEST IF YES
023300                               1$:
023300  004737  035430                          JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
023304  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023306  100400                                  100400                  ;START AT ROM PC=0
023310  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023312  117777                                  114377!<400*7>          ;JUMP TO ROM PC OF 1777
023314  004737  035522                          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
023320  000002                                  2                       ;INDEX
023322  020504                                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
023324  001401                                  BEQ     2$              ;BR IF YES
023326  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
023330  104401                       2$:        SCOP1                   ;LOOP TO 1$ IF SW09=1
023332  012737  023340  001220                  MOV     #3$,LOCK        ;NEW SCOP1
023340                               3$:
023340  004737  035430                          JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
023344  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023346  100403                                  100403                  ;START AT ROM PC=3
023350  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023352  103400                                  100000!<400*7>  ;JUMP TO ROM PC OF 0
023354  004737  035522                          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
023360  000010                                  10                      ;INDEX
023362  020504                                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
023364  001401                                  BEQ     4$              ;BR IF YES
023366  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
023370  104401                       4$:        SCOP1                   ;LOOP TO 3$ IF SW09=1
023372  012737  023400  001220                  MOV     #5$,LOCK        ;NEW SCOP1
023400                               5$:
023400  004737  035430                          JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
023404  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023406  100406                                  100406                  ;START AT ROM PC=6
023410  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023412  107525                                  104125!<400*7>          ;JUMP TO ROM PC OF 525
023414  004737  035522                          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
023420  000016                                  16                      ;INDEX
023422  020504                                  CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
023424  001401                                  BEQ     6$              ;BR IF YES
023426  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
023430  104401                       6$:        SCOP1                   ;LOOP TO 5$ IF SW59=1
023432  104400                                  SCOPE                   ;SCOPE THIS TEST


                                                ;************************* TEST 32 *************************
                                                ;*CRAM TEST OF JUMP(I) NEVER MICRO-PROCESSOR INSTRUCTION.
                                                ;*PERFORM THE JUMP INSTRUCTION
                                                ;*VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
                                                ;*IN THE LOCATION IT IS AT. THIS INSTRUCTION LOADS THE
                                                ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC. AT THIS POINT
```



```
                                                ;*THE BR DATA IS MOVED TO PORT4. IF THIS DATA IS CORRECT
                                                ;*THE CRAM PC IS CORRECT. IF THE CRAM PC IS NOT RIGHT,
                                                ;*THEN PORT4 CONTAINS A 37
                                                ;;***********************************************************

                                                ; TEST 32
                                                ;--------------
023434  012737  000032  001226       TST32:     MOV     #32,TSTNO
023442  012737  023630  001216                  MOV     #TST33,NEXT
023450  012737  023474  001220                  MOV     #1$,LOCK
                                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
023456  104412                                  MSTCLR                  ;MASTER CLEAR DMC11
023460  032737  100000  001366                  BIT     #BIT15,STAT1    ;IS IT CRAM?
023466  001457                                  BEQ     6$+2            ;SKIP TEST IF NO
023470  004737  035654                          JSR     PC,MEMSET       ;SET MEM AND RAM
023474                               1$:
023474  004737  035430                          JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
023500  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023502  100400                                  100400                  ;START AT ROM PC=0
023504  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023506  114377                                  114377!<400*0>          ;JUMP TO ROM PC OF 1777
023510  004737  035550                          JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
023514  000001                                  1                       ;EXPECTED DATA
023516  120504                                  CMPB    R5,R4           ;IS ROM PC CORRECT?
023520  001401                                  BEQ     2$              ;BR IF YES
023522  104005                                  HLT     5               ;ERROR, CRAM PC IS WRONG
023524  104401                       2$:        SCOP1                   ;LOOP TO 1$ IF SW09=1
023526  012737  023534  001220                  MOV     #3$,LOCK        ;NEW SCOP1
023534                               3$:
023534  004737  035430                          JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
023540  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023542  100403                                  100403                  ;START AT ROM PC=3
023544  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023546  100000                                  100000!<400*0>  ;JUMP TO ROM PC OF 0
023550  004737  035550                          JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
023554  000004                                  4                       ;EXPECTED DATA
023556  120504                                  CMPB    R5,R4           ;IS ROM PC CORRECT?
023560  001401                                  BEQ     4$              ;BR IF YES
023562  104005                                  HLT     5               ;ERROR, CRAM PC IS WRONG
023564  104401                       4$:        SCOP1                   ;LOOP TO 3$ IF SW09=1
023566  012737  023574  001220                  MOV     #5$,LOCK        ;NEW SCOP1
023574                               5$:
023574  004737  035430                          JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
023600  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023602  100406                                  100406                  ;START AT ROM PC=6
023604  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023606  104125                                  104125!<400*0>          ;JUMP TO ROM PC OF 525
023610  004737  035550                          JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
023614  000007                                  7                       ;EXPECTED DATA
023616  120504                                  CMPB    R5,R4           ;IS ROM PC CORRECT?
023620  001401                                  BEQ     6$              ;BR IF YES
023622  104005                                  HLT     5               ;ERROR, CRAM PC IS WRONG
023624  104401                       6$:        SCOP1                   ;LOOP TO 5$ IF SW59=1
023626  104400                                  SCOPE                   ;SCOPE THIS TEST
```

```
                                        ;************************** TEST 33 **************************
                                        ;*CRAM TEST OF JUMP(I) ALWAYS MICRO-PROCESSOR INSTRUCTION,
                                        ;*PERFORM THE JUMP INSTRUCTION
                                        ;*VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
                                        ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
                                        ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
                                        ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
                                        ;*THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
                                        ;*THEN PORT4 WILL CONTAIN A 37
                                        ;!*************************************************************

                                        ;  TEST 33
                                        ;--------------
023630  012737  000033  001226  TST33:  MOV     #33,TSTNO
023636  012737  024010  001216          MOV     #TST34,NEXT
023644  012737  023670  001220          MOV     #1$,LOCK
                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
023652  104412                          MSTCLR                  ;MASTER CLEAR DMC11
023654  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
023662  001451                          BEQ     6$+2            ;SKIP TEST IF NO
023664  004737  035654                  JSR     PC,MEMSET       ;SET MEM AND RAM
023670                          1$:
023670  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023672  100400                          100400                  ;START AT ROM PC=0
023674  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023676  114777                          1143771!<400*1>         ;JUMP TO ROM PC OF 1777
023700  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
023704  000377                          377                     ;EXPECTED DATA
023706  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
023710  001401                          BEQ     2$              ;BR IF YES
023712  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
023714  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
023716  012737  023724  001220          MOV     #3$,LOCK        ;NEW SCOP1
023724                          3$:
023724  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023726  100403                          100403                  ;START AT ROM PC=3
023730  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023732  100400                          1000001<400*1>  ;JUMP TO ROM PC OF 0
023734  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
023740  000000                          0                       ;EXPECTED DATA
023742  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
023744  001401                          BEQ     4$              ;BR IF YES
023746  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
023750  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
023752  012737  023760  001220          MOV     #5$,LOCK        ;NEW SCOP1
023760                          5$:
023760  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023762  100406                          100406                  ;START AT ROM PC=6
023764  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023766  104525                          1041251<400*1>          ;JUMP TO ROM PC OF 525
023770  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
023774  000125                          125                     ;EXPECTED DATA
023776  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
024000  001401                          BEQ     6$              ;BR IF YES
024002  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
024004  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
```



```
024006  104400                          SCOPE                   ;SCOPE THIS TEST


                                        ;************************** TEST 34 **************************
                                        ;*CRAM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION,
                                        ;*SET THE C BIT, PERFORM THE JUMP INSTRUCTION,
                                        ;*VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
                                        ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
                                        ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
                                        ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
                                        ;*THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
                                        ;*THEN PORT4 WILL CONTAIN A 37
                                        ;!*************************************************************


                                        ;  TEST 34
                                        ;--------------
024010  012737  000034  001226  TST34:  MOV     #34,TSTNO
024016  012737  024204  001216          MOV     #TST35,NEXT
024024  012737  024050  001220          MOV     #1$,LOCK
                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
024032  104412                          MSTCLR                  ;MASTER CLEAR DMC11
024034  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
024042  001457                          BEQ     6$+2            ;SKIP TEST IF NO
024044  004737  035654                  JSR     PC,MEMSET       ;SET MEM AND RAM
024050                          1$:
024050  004737  035476                  JSR     PC,SETC ;SET THE C BIT'
024054  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024056  100400                          100400                  ;START AT ROM PC=0
024060  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024062  115377                          1143771!<400*2>         ;JUMP TO ROM PC OF 1777
024064  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
024070  000377                          377                     ;EXPECTED DATA
024072  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
024074  001401                          BEQ     2$              ;BR IF YES
024076  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
024100  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
024102  012737  024110  001220          MOV     #3$,LOCK        ;NEW SCOP1
024110                          3$:
024110  004737  035476                  JSR     PC,SETC ;SET THE C BIT'
024114  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024116  100403                          100403                  ;START AT ROM PC=3
024120  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024122  101000                          100000!<400*2>  ;JUMP TO ROM PC OF 0
024124  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
024130  000000                          0                       ;EXPECTED DATA
024132  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
024134  001401                          BEQ     4$              ;BR IF YES
024136  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
024140  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
024142  012737  024150  001220          MOV     #5$,LOCK        ;NEW SCOP1
024150                          5$:
024150  004737  035476                  JSR     PC,SETC ;SET THE C BIT'
024154  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024156  100406                          100406                  ;START AT ROM PC=6
024160  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024162  105125                          1041251<400*2>          ;JUMP TO ROM PC OF 525
```

```
3094  024164  004737  035550                      JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3095  024170  000125                              125                     ;EXPECTED DATA
3096  024172  120504                              CMPB    R5,R4           ;IS ROM PC CORRECT?
3097  024174  001401                              BEQ     6$              ;BR IF YES
3098  024176  104005                              HLT     5               ;ERROR, CRAM PC IS WRONG
3099  024200  104401                      6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
3100  024202  104400                              SCOPE                   ;SCOPE THIS TEST
3101
3102
3103                                              ;************************** TEST 35 **************************
3104                                              ;*CRAM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION,
3105                                              ;*SET THE Z BIT, PERFORM THE JUMP INSTRUCTION,
3106                                              ;*VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
3107                                              ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
3108                                              ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
3109                                              ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
3110                                              ;*THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
3111                                              ;*THEN PORT4 WILL CONTAIN A 37
3112                                              ;!*************************************************************
3113
3114                                              ;  TEST 35
3115                                              ;--------------
3116  024204  012737  000035  001226      TST35:  MOV     #35,TSTNO
3117  024212  012737  024400  001216              MOV     #TST36,NEXT
3118  024220  012737  024244  001220              MOV     #1$,LOCK
3119                                                                      ;R1 CONTAINS BASE DMC11 ADDRESS
3120  024226  104412                              MSTCLR                  ;MASTER CLEAR DMC11
3121  024230  032737  100000  001366              BIT     #BIT15,STAT1    ;IS IT CRAM?
3122  024236  001457                              BEQ     6$+2            ;SKIP TEST IF NO
3123  024240  004737  035654                      JSR     PC,MEMSET       ;SET MEM AND RAM
3124  024244                              1$:
3125  024244  004737  035514                      JSR     PC,SETZ ;SET THE Z BIT'
3126  024250  104414                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3127  024252  100400                              100400                  ;START AT ROM PC=0
3128  024254  104414                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3129  024256  115777                              1143771<400*3>          ;JUMP TO ROM PC OF 1777
3130  024260  004737  035550                      JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3131  024264  000377                              377                     ;EXPECTED DATA
3132  024266  120504                              CMPB    R5,R4           ;IS ROM PC CORRECT?
3133  024270  001401                              BEQ     2$              ;BR IF YES
3134  024272  104005                              HLT     5               ;ERROR, CRAM PC IS WRONG
3135  024274  104401                      2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
3136  024276  012737  024304  001220              MOV     #3$,LOCK        ;NEW SCOP1
3137  024304                              3$:
3138  024304  004737  035514                      JSR     PC,SETZ ;SET THE Z BIT'
3139  024310  104414                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3140  024312  100403                              100403                  ;START AT ROM PC=3
3141  024314  104414                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3142  024316  101400                              100000!<400*3>  ;JUMP TO ROM PC OF 0
3143  024320  004737  035550                      JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3144  024324  000000                              0                       ;EXPECTED DATA
3145  024326  120504                              CMPB    R5,R4           ;IS ROM PC CORRECT?
3146  024330  001401                              BEQ     4$              ;BR IF YES
3147  024332  104005                              HLT     5               ;ERROR, CRAM PC IS WRONG
3148  024334  104401                      4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
3149  024336  012737  024344  001220              MOV     #5$,LOCK        ;NEW SCOP1
```



```
3150  024344                              5$:
3151  024344  004737  035514                      JSR     PC,SETZ ;SET THE Z BIT'
3152  024350  104414                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3153  024352  100406                              100406                  ;START AT ROM PC=6
3154  024354  104414                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3155  024356  105525                              1041251<400*3>          ;JUMP TO ROM PC OF 525
3156  024360  004737  035550                      JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3157  024364  000125                              125                     ;EXPECTED DATA
3158  024366  120504                              CMPB    R5,R4           ;IS ROM PC CORRECT?
3159  024370  001401                              BEQ     6$              ;BR IF YES
3160  024372  104005                              HLT     5               ;ERROR, CRAM PC IS WRONG
3161  024374  104401                      6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
3162  024376  104400                              SCOPE                   ;SCOPE THIS TEST
3163
3164
3165                                              ;************************** TEST 36 **************************
3166                                              ;*CRAM TEST OF JUMP(I) ON BR0 SET MICRO-PROCESSOR INSTRUCTION,
3167                                              ;*SET THE BR0 BIT, PERFORM THE JUMP INSTRUCTION,
3168                                              ;*VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
3169                                              ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
3170                                              ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
3171                                              ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
3172                                              ;*THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
3173                                              ;*THEN PORT4 WILL CONTAIN A 37
3174                                              ;!*************************************************************
3175
3176                                              ;  TEST 36
3177                                              ;--------------
3178  024400  012737  000036  001226      TST36:  MOV     #36,TSTNO
3179  024406  012737  024574  001216              MOV     #TST37,NEXT
3180  024414  012737  024440  001220              MOV     #1$,LOCK
3181                                                                      ;R1 CONTAINS BASE DMC11 ADDRESS
3182  024422  104412                              MSTCLR                  ;MASTER CLEAR DMC11
3183  024424  032737  100000  001366              BIT     #BIT15,STAT1    ;IS IT CRAM?
3184  024432  001457                              BEQ     6$+2            ;SKIP TEST IF NO
3185  024434  004737  035654                      JSR     PC,MEMSET       ;SET MEM AND RAM
3186  024440                              1$:
3187  024440  004737  035446                      JSR     PC,SETBR0       ;SET THE BR0 BIT'
3188  024444  104414                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3189  024446  100400                              100400                  ;START AT ROM PC=0
3190  024450  104414                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3191  024452  116377                              1143771<400*4>          ;JUMP TO ROM PC OF 1777
3192  024454  004737  035550                      JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3193  024460  000377                              377                     ;EXPECTED DATA
3194  024462  120504                              CMPB    R5,R4           ;IS ROM PC CORRECT?
3195  024464  001401                              BEQ     2$              ;BR IF YES
3196  024466  104005                              HLT     5               ;ERROR, CRAM PC IS WRONG
3197  024470  104401                      2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
3198  024472  012737  024500  001220              MOV     #3$,LOCK        ;NEW SCOP1
3199  024500                              3$:
3200  024500  004737  035446                      JSR     PC,SETBR0       ;SET THE BR0 BIT'
3201  024504  104414                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3202  024506  100403                              100403                  ;START AT ROM PC=3
3203  024510  104414                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3204  024512  102000                              100000!<400*4>  ;JUMP TO ROM PC OF 0
3205  024514  004737  035550                      JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
```

```
3206  024520  000000                          0                       ;EXPECTED DATA
3207  024522  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3208  024524  001401                          BEQ     4$              ;BR IF YES
3209  024526  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3210  024530  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
3211  024532  012737  024540  001220          MOV     #5$,LOCK        ;NEW SCOP1
3212  024540                          5$:
3213  024540  004737  035446                  JSR     PC,SETBR0       ;SET THE BR0 BIT'
3214  024544  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3215  024546  100406                          100406                  ;START AT ROM PC=6
3216  024550  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3217  024552  106125                          104125!<400*4>          ;JUMP TO ROM PC OF 525
3218  024554  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3219  024560  000125                          125                     ;EXPECTED DATA
3220  024562  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3221  024564  001401                          BEQ     6$              ;BR IF YES
3222  024566  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3223  024570  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
3224  024572  104400                          SCOPE                   ;SCOPE THIS TEST
3225
3226
3227                                          ;************************** TEST 37 **************************
3228                                          ;*CRAM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION,
3229                                          ;*SET THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
3230                                          ;*VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
3231                                          ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
3232                                          ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
3233                                          ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
3234                                          ;*THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
3235                                          ;*THEN PORT4 WILL CONTAIN A 37
3236                                          ;;*************************************************************
3237
3238                                          ;  TEST 37
3239                                          ;--------------
3240  024574  012737  000037  001226  TST37:  MOV     #37,TSTNO
3241  024602  012737  024770  001216          MOV     #TST40,NEXT
3242  024610  012737  024634  001220          MOV     #1$,LOCK
3243                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
3244  024616  104412                          MSTCLR                  ;MASTER CLEAR DMC11
3245  024620  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
3246  024626  001457                          BEQ     6$+2            ;SKIP TEST IF NO
3247  024630  004737  035654                  JSR     PC,MEMSET       ;SET MEM AND RAM
3248  024634                          1$:
3249  024634  004737  035454                  JSR     PC,SETBR1       ;SET THE BR1 BIT'
3250  024640  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3251  024642  100400                          100400                  ;START AT ROM PC=0
3252  024644  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3253  024646  116777                          114377!<400*5>          ;JUMP TO ROM PC OF 1777
3254  024650  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3255  024654  000377                          377                     ;EXPECTED DATA
3256  024656  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3257  024660  001401                          BEQ     2$              ;BR IF YES
3258  024662  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3259  024664  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
3260  024666  012737  024674  001220          MOV     #3$,LOCK        ;NEW SCOP1
3261  024674                          3$:
```



```
3262  024674  004737  035454                  JSR     PC,SETBR1       ;SET THE BR1 BIT'
3263  024700  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3264  024702  100403                          100403                  ;START AT ROM PC=3
3265  024704  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3266  024706  102400                          100000!<400*5>  ;JUMP TO ROM PC OF 0
3267  024710  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3268  024714  000000                          0                       ;EXPECTED DATA
3269  024716  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3270  024720  001401                          BEQ     4$              ;BR IF YES
3271  024722  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3272  024724  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
3273  024726  012737  024734  001220          MOV     #5$,LOCK        ;NEW SCOP1
3274  024734                          5$:
3275  024734  004737  035454                  JSR     PC,SETBR1       ;SET THE BR1 BIT'
3276  024740  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3277  024742  100406                          100406                  ;START AT ROM PC=6
3278  024744  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3279  024746  106525                          104125!<400*5>          ;JUMP TO ROM PC OF 525
3280  024750  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3281  024754  000125                          125                     ;EXPECTED DATA
3282  024756  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3283  024760  001401                          BEQ     6$              ;BR IF YES
3284  024762  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3285  024764  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
3286  024766  104400                          SCOPE                   ;SCOPE THIS TEST
3287
3288
3289                                          ;************************** TEST 40 **************************
3290                                          ;*CRAM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION,
3291                                          ;*SET THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
3292                                          ;*VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
3293                                          ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
3294                                          ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
3295                                          ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
3296                                          ;*THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
3297                                          ;*THEN PORT4 WILL CONTAIN A 37
3298                                          ;;*************************************************************
3299
3300                                          ;  TEST 40
3301                                          ;--------------
3302  024770  012737  000040  001226  TST40:  MOV     #40,TSTNO
3303  024776  012737  025164  001216          MOV     #TST41,NEXT
3304  025004  012737  025030  001220          MOV     #1$,LOCK
3305                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
3306  025012  104412                          MSTCLR                  ;MASTER CLEAR DMC11
3307  025014  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
3308  025022  001457                          BEQ     6$+2            ;SKIP TEST IF NO
3309  025024  004737  035654                  JSR     PC,MEMSET       ;SET MEM AND RAM
3310  025030                          1$:
3311  025030  004737  035462                  JSR     PC,SETBR4       ;SET THE BR4 BIT'
3312  025034  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3313  025036  100400                          100400                  ;START AT ROM PC=0
3314  025040  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3315  025042  116777                          114377!<400*6>          ;JUMP TO ROM PC OF 1777
3316  025044  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3317  025050  000377                          377                     ;EXPECTED DATA
```

```
3318  025052  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3319  025054  001401                          BEQ     2$              ;BR IF YES
3320  025056  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3321  025060  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
3322  025062  012737  025070  001220          MOV     #3$,LOCK        ;NEW SCOP1
3323  025070                           3$:
3324  025070  004737  035462                  JSR     PC,SETBR4       ;SET THE BR4 BIT'
3325  025074  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3326  025076  100403                          100403                  ;START AT ROM PC=3
3327  025100  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3328  025102  103000                          1000001<400*6>  ;JUMP TO ROM PC OF 0
3329  025104  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3330  025110  000000                          0                       ;EXPECTED DATA
3331  025112  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3332  025114  001401                          BEQ     4$              ;BR IF YES
3333  025116  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3334  025120  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
3335  025122  012737  025130  001220          MOV     #5$,LOCK        ;NEW SCOP1
3336  025130                           5$:
3337  025130  004737  035462                  JSR     PC,SETBR4       ;SET THE BR4 BIT'
3338  025134  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3339  025136  100406                          100406                  ;START AT ROM PC=6
3340  025140  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3341  025142  107125                          104125<400*6>           ;JUMP TO ROM PC OF 525
3342  025144  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3343  025150  000125                          125                     ;EXPECTED DATA
3344  025152  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3345  025154  001401                          BEQ     6$              ;BR IF YES
3346  025156  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3347  025160  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
3348  025162  104400                          SCOPE                   ;SCOPE THIS TEST
3349
3350
3351                                           ;************************** TEST 41 ************************
3352                                           ;*CRAM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION.
3353                                           ;*SET THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
3354                                           ;*VERIFY THE JUMP DID OCCUR BY CLOCKING THE INSRUCTION
3355                                           ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
3356                                           ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
3357                                           ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT,
3358                                           ;*THE JUMP WAS SUCCESSFUL, IF THE JUMP WAS UNSUCCESSFUL
3359                                           ;*THEN PORT4 WILL CONTAIN A'37
3360                                           ;!*********************************************************
3361
3362                                           ;  TEST 41
3363                                           ;--------------
3364  025164  012737  000041  001226  TST41:  MOV     #41,TSTNO
3365  025172  012737  025360  001216          MOV     #TST42,NEXT
3366  025200  012737  025224  001220          MOV     #1$,LOCK
3367                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
3368  025206  104412                          MSTCLR                  ;MASTER CLEAR DMC11
3369  025210  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
3370  025216  001457                          BEQ     6$+2            ;SKIP TEST IF NO
3371  025220  004737  035654                  JSR     PC,MEMSET       ;SET MEM AND RAM
3372  025224                           1$:
3373  025224  004737  035470                  JSR     PC,SETBR7       ;SET THE BR7 BIT'
```



```
3374  025230  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3375  025232  100400                          100400                  ;START AT ROM PC=0
3376  025234  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3377  025236  117777                          1143771<400*7>          ;JUMP TO ROM PC OF 1777
3378  025240  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3379  025244  000377                          377                     ;EXPECTED DATA
3380  025246  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3381  025250  001401                          BEQ     2$              ;BR IF YES
3382  025252  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3383  025254  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
3384  025256  012737  025264  001220          MOV     #3$,LOCK        ;NEW SCOP1
3385  025264                           3$:
3386  025264  004737  035470                  JSR     PC,SETBR7       ;SET THE BR7 BIT'
3387  025270  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3388  025272  100403                          100403                  ;START AT ROM PC=3
3389  025274  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3390  025276  103400                          1000001<400*7>  ;JUMP TO ROM PC OF 0
3391  025300  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3392  025304  000000                          0                       ;EXPECTED DATA
3393  025306  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3394  025310  001401                          BEQ     4$              ;BR IF YES
3395  025312  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3396  025314  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
3397  025316  012737  025324  001220          MOV     #5$,LOCK        ;NEW SCOP1
3398  025324                           5$:
3399  025324  004737  035470                  JSR     PC,SETBR7       ;SET THE BR7 BIT'
3400  025330  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3401  025332  100406                          100406                  ;START AT ROM PC=6
3402  025334  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3403  025336  107525                          104125<400*7>           ;JUMP TO ROM PC OF 525
3404  025340  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3405  025344  000125                          125                     ;EXPECTED DATA
3406  025346  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3407  025350  001401                          BEQ     6$              ;BR IF YES
3408  025352  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3409  025354  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
3410  025356  104400                          SCOPE                   ;SCOPE THIS TEST
3411
3412
3413                                           ;************************** TEST 42 ************************
3414                                           ;*CRAM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
3415                                           ;*CLEAR THE C BIT, PERFORM THE JUMP INSTRUCTION,
3416                                           ;*VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
3417                                           ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
3418                                           ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
3419                                           ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
3420                                           ;*THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
3421                                           ;*THEN PORT4 CONTAINS A 37
3422                                           ;!*********************************************************
3423
3424                                           ;  TEST 42
3425                                           ;--------------
3426  025360  012737  000042  001226  TST42:  MOV     #42,TSTNO
3427  025366  012737  025554  001216          MOV     #TST43,NEXT
3428  025374  012737  025420  001220          MOV     #1$,LOCK
3429                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
```

```
3430  025402  104412                          MSTCLR                  ;MASTER CLEAR DMC11
3431  025404  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
3432  025412  001457                          BEQ     6$+2            ;SKIP TEST IF NO
3433  025414  004737  035654                  JSR     PC,MEMSET       ;SET MEM AND RAM
3434  025420                          1$:
3435  025420  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
3436  025424  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3437  025426  100400                          100400                  ;START AT ROM PC=0
3438  025430  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3439  025432  115377                          1143771<400*2>          ;JUMP TO ROM PC OF 1777
3440  025434  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3441  025440  000001                          1                       ;EXPECTED DATA
3442  025442  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3443  025444  001401                          BEQ     2$              ;BR IF YES
3444  025446  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3445  025450  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
3446  025452  012737  025460  001220          MOV     #3$,LOCK        ;NEW SCOP1
3447  025460                          3$:
3448  025460  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
3449  025464  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3450  025466  100403                          100403                  ;START AT ROM PC=3
3451  025470  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3452  025472  101000                          1000001<400*2>  ;JUMP TO ROM PC OF 0
3453  025474  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3454  025500  000004                          4                       ;EXPECTED DATA
3455  025502  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3456  025504  001401                          BEQ     4$              ;BR IF YES
3457  025506  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3458  025510  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
3459  025512  012737  025520  001220          MOV     #5$,LOCK        ;NEW SCOP1
3460  025520                          5$:
3461  025520  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
3462  025524  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3463  025526  100406                          100406                  ;START AT ROM PC=6
3464  025530  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3465  025532  105125                          1041251<400*2>          ;JUMP TO ROM PC OF 525
3466  025534  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3467  025540  000007                          7                       ;EXPECTED DATA
3468  025542  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3469  025544  001401                          BEQ     6$              ;BR IF YES
3470  025546  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3471  025550  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
3472  025552  104400                          SCOPE                   ;SCOPE THIS TEST
3473
3474
3475                                          ;*************************** TEST 43 ***************************
3476                                          ;*CRAM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
3477                                          ;*CLEAR THE Z BIT, PERFORM THE JUMP INSTRUCTION,
3478                                          ;*VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
3479                                          ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
3480                                          ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
3481                                          ;*THE BR DATA IS MOVED TO PORT4. IF THIS DATA IS CORRECT
3482                                          ;*THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
3483                                          ;*THEN PORT4 CONTAINS A 37
3484                                          ;;*************************************************************
3485
```



```
3486                                          ;  TEST 43
3487                                          ;--------------
3488  025554  012737  000043  001226  TST43:  MOV     #43,TSTNO
3489  025562  012737  025750  001216          MOV     #TST44,NEXT
3490  025570  012737  025614  001220          MOV     #1$,LOCK
3491                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
3492  025576  104412                          MSTCLR                  ;MASTER CLEAR DMC11
3493  025600  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
3494  025606  001457                          BEQ     6$+2            ;SKIP TEST IF NO
3495  025610  004737  035654                  JSR     PC,MEMSET       ;SET MEM AND RAM
3496  025614                          1$:
3497  025614  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
3498  025620  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3499  025622  100400                          100400                  ;START AT ROM PC=0
3500  025624  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3501  025626  115777                          1143771<400*3>          ;JUMP TO ROM PC OF 1777
3502  025630  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3503  025634  000001                          1                       ;EXPECTED DATA
3504  025636  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3505  025640  001401                          BEQ     2$              ;BR IF YES
3506  025642  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3507  025644  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
3508  025646  012737  025654  001220          MOV     #3$,LOCK        ;NEW SCOP1
3509  025654                          3$:
3510  025654  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
3511  025660  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3512  025662  100403                          100403                  ;START AT ROM PC=3
3513  025664  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3514  025666  101400                          1000001<400*3>  ;JUMP TO ROM PC OF 0
3515  025670  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3516  025674  000004                          4                       ;EXPECTED DATA
3517  025676  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3518  025700  001401                          BEQ     4$              ;BR IF YES
3519  025702  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3520  025704  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
3521  025706  012737  025714  001220          MOV     #5$,LOCK        ;NEW SCOP1
3522  025714                          5$:
3523  025714  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
3524  025720  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3525  025722  100406                          100406                  ;START AT ROM PC=6
3526  025724  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3527  025726  105525                          1041251<400*3>          ;JUMP TO ROM PC OF 525
3528  025730  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3529  025734  000007                          7                       ;EXPECTED DATA
3530  025736  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3531  025740  001401                          BEQ     6$              ;BR IF YES
3532  025742  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3533  025744  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
3534  025746  104400                          SCOPE                   ;SCOPE THIS TEST
3535
3536
3537                                          ;*************************** TEST 44 ***************************
3538                                          ;*CRAM TEST OF JUMP(I) ON BR0 SET MICRO-PROCESSOR INSTRUCTION.
3539                                          ;*CLEAR THE BR0 BIT, PERFORM THE JUMP INSTRUCTION,
3540                                          ;*VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
3541                                          ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
```

```
                                        ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
                                        ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
                                        ;*THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
                                        ;*THEN PORT4 CONTAINS A 37
                                        ;!*****************************************************************

                                        ;  TEST 44
                                        ;--------------
025750  012737  000044  001226  TST44:  MOV     #44,TSTNO
025756  012737  026144  001216          MOV     #TST45,NEXT
025764  012737  026010  001220          MOV     #1$,LOCK
                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
025772  104412                          MSTCLR                  ;MASTER CLEAR DMC11
025774  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
026002  001457                          BEQ     6$+2            ;SKIP TEST IF NO
026004  004737  035654                  JSR     PC,MEMSET       ;SET MEM AND RAM
026010                          1$:
026010  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
026014  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026016  100400                          100400                  ;START AT ROM PC=0
026020  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026022  116377                          114377!<400*4>          ;JUMP TO ROM PC OF 1777
026024  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
026030  000001                          1                       ;EXPECTED DATA
026032  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
026034  001401                          BEQ     2$              ;BR IF YES
026036  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
026040  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
026042  012737  026050  001220          MOV     #3$,LOCK        ;NEW SCOP1
026050                          3$:
026050  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
026054  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026056  100403                          100403                  ;START AT ROM PC=3
026060  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026062  102000                          1000001<400*4>  ;JUMP TO ROM PC OF 0
026064  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
026070  000004                          4                       ;EXPECTED DATA
026072  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
026074  001401                          BEQ     4$              ;BR IF YES
026076  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
026100  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
026102  012737  026110  001220          MOV     #5$,LOCK        ;NEW SCOP1
026110                          5$:
026110  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
026114  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026116  100406                          100406                  ;START AT ROM PC=6
026120  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026122  106125                          1041251<400*4>          ;JUMP TO ROM PC OF 525
026124  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
026130  000007                          7                       ;EXPECTED DATA
026132  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
026134  001401                          BEQ     6$              ;BR IF YES
026136  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
026140  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
026142  104400                          SCOPE                   ;SCOPE THIS TEST
```



```
                                        ;************************** TEST 45 **************************
                                        ;*CRAM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION,
                                        ;*CLEAR THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
                                        ;*VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
                                        ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
                                        ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
                                        ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
                                        ;*THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
                                        ;*THEN PORT4 CONTAINS A 37
                                        ;!*****************************************************************

                                        ;  TEST 45
                                        ;--------------
026144  012737  000045  001226  TST45:  MOV     #45,TSTNO
026152  012737  026340  001216          MOV     #TST46,NEXT
026160  012737  026204  001220          MOV     #1$,LOCK
                                                                ;P1 CONTAINS BASE DMC11 ADDRESS
026166  104412                          MSTCLR                  ;MASTER CLEAR DMC11
026170  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
026176  001457                          BEQ     6$+2            ;SKIP TEST IF NO
026200  004737  035654                  JSR     PC,MEMSET       ;SET MEM AND RAM
026204                          1$:
026204  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
026210  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026212  100400                          100400                  ;START AT ROM PC=0
026214  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026216  116777                          114377!<400*5>          ;JUMP TO ROM PC OF 1777
026220  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
026224  000001                          1                       ;EXPECTED DATA
026226  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
026230  001401                          BEQ     2$              ;BR IF YES
026232  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
026234  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
026236  012737  026244  001220          MOV     #3$,LOCK        ;NEW SCOP1
026244                          3$:
026244  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
026250  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026252  100403                          100403                  ;START AT ROM PC=3
026254  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026256  102400                          1000001<400*5>  ;JUMP TO ROM PC OF 0
026260  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
026264  000004                          4                       ;EXPECTED DATA
026266  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
026270  001401                          BEQ     4$              ;BR IF YES
026272  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
026274  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
026276  012737  026304  001220          MOV     #5$,LOCK        ;NEW SCOP1
026304                          5$:
026304  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
026310  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026312  100406                          100406                  ;START AT ROM PC=6
026314  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026316  106525                          1041251<400*5>          ;JUMP TO ROM PC OF 525
026320  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
026324  000007                          7                       ;EXPECTED DATA
```

```
3654  026326  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3655  026330  001401                          BEQ     6$              ;BR IF YES
3656  026332  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3657  026334  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
3658  026336  104400                          SCOPE                   ;SCOPE THIS TEST
3659
3660
3661                                          ;************************** TEST 46 **************************
3662                                          ;*CRAM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION,
3663                                          ;*CLEAR THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
3664                                          ;*VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
3665                                          ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
3666                                          ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
3667                                          ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
3668                                          ;*THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
3669                                          ;*THEN PORT4 CONTAINS A 37
3670                                          ;!************************************************************
3671
3672                                          ;  TEST 46
3673                                          ;--------------
3674  026340  012737  000046  001226  TST46:  MOV     #46,TSTNO
3675  026346  012737  026534  001216          MOV     #TST47,NEXT
3676  026354  012737  026400  001220          MOV     #1$,LOCK
3677                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
3678  026362  104412                          MSTCLR                  ;MASTER CLEAR DMC11
3679  026364  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
3680  026372  001457                          BEQ     6$+2            ;SKIP TEST IF NO
3681  026374  004737  035654                  JSR     PC,MEMSET       ;SET MEM AND RAM
3682  026400                          1$:
3683  026400  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
3684  026404  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3685  026406  100400                          100400                  ;START AT ROM PC=0
3686  026410  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3687  026412  117377                          1143771<400*6>          ;JUMP TO ROM PC OF 1777
3688  026414  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3689  026420  000001                          1                       ;EXPECTED DATA
3690  026422  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3691  026424  001401                          BEQ     2$              ;BR IF YES
3692  026426  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3693  026430  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
3694  026432  012737  026440  001220          MOV     #3$,LOCK        ;NEW SCOP1
3695  026440                          3$:
3696  026440  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
3697  026444  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3698  026446  100403                          100403                  ;START AT ROM PC=3
3699  026450  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3700  026452  103000                          1000001<400*6>  ;JUMP TO ROM PC OF 0
3701  026454  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3702  026460  000004                          4                       ;EXPECTED DATA
3703  026462  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3704  026464  001401                          BEQ     4$              ;BR IF YES
3705  026466  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3706  026470  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
3707  026472  012737  026500  001220          MOV     #5$,LOCK        ;NEW SCOP1
3708  026500                          5$:
3709  026500  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
```



```
3710  026504  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3711  026506  100406                          100406                  ;START AT ROM PC=6
3712  026510  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3713  026512  107125                          1041251<400*6>          ;JUMP TO ROM PC OF 525
3714  026514  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3715  026520  000007                          7                       ;EXPECTED DATA
3716  026522  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3717  026524  001401                          BEQ     6$              ;BR IF YES
3718  026526  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3719  026530  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
3720  026532  104400                          SCOPE                   ;SCOPE THIS TEST
3721
3722
3723                                          ;************************** TEST 47 **************************
3724                                          ;*CRAM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION,
3725                                          ;*CLEAR THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
3726                                          ;*VERIFY THE JUMP DID NOT OCCUR BY CLOCKING THE INSTRUCTION
3727                                          ;*IN THE LOCATION IT IS AT, THIS INSTRUCTION LOADS THE
3728                                          ;*BR WITH THE LOWEST 8 BITS OF THE CRAM PC, AT THIS POINT
3729                                          ;*THE BR DATA IS MOVED TO PORT4, IF THIS DATA IS CORRECT
3730                                          ;*THE CRAM PC IS CORRECT, IF THE CRAM PC IS NOT RIGHT,
3731                                          ;*THEN PORT4 CONTAINS A 37
3732                                          ;!************************************************************
3733
3734                                          ;  TEST 47
3735                                          ;--------------
3736  026534  012737  000047  001226  TST47:  MOV     #47,TSTNO
3737  026542  012737  026730  001216          MOV     #TST50,NEXT
3738  026550  012737  026574  001220          MOV     #1$,LOCK
3739                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
3740  026556  104412                          MSTCLR                  ;MASTER CLEAR DMC11
3741  026560  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
3742  026566  001457                          BEQ     6$+2            ;SKIP TEST IF NO
3743  026570  004737  035654                  JSR     PC,MEMSET       ;SET MEM AND RAM
3744  026574                          1$:
3745  026574  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
3746  026600  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3747  026602  100400                          100400                  ;START AT ROM PC=0
3748  026604  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3749  026606  117777                          1143771<400*7>          ;JUMP TO ROM PC OF 1777
3750  026610  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3751  026614  000001                          1                       ;EXPECTED DATA
3752  026616  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3753  026620  001401                          BEQ     2$              ;BR IF YES
3754  026622  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3755  026624  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
3756  026626  012737  026634  001220          MOV     #3$,LOCK        ;NEW SCOP1
3757  026634                          3$:
3758  026634  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
3759  026640  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3760  026642  100403                          100403                  ;START AT ROM PC=3
3761  026644  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3762  026646  103000                          1000001<400*7>  ;JUMP TO ROM PC OF 0
3763  026650  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3764  026654  000004                          4                       ;EXPECTED DATA
3765  026656  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
```

```
3766  026660  001401                          BEQ     4$              ;BR IF YES
3767  026662  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3768  026664  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
3769  026666  012737  026674  001220          MOV     #5$,LOCK        ;NEW SCOP1
3770  026674                          5$:
3771  026674  004737  035430                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
3772  026700  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3773  026702  100406                          100406                  ;START AT ROM PC=6
3774  026704  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3775  026706  107525                          1041251<400*7>          ;JUMP TO ROM PC OF 525
3776  026710  004737  035550                  JSR     PC,RAMDAT       ;R4=CRAM PC (LSB 8 BITS)
3777  026714  000007                          7                       ;EXPECTED DATA
3778  026716  120504                          CMPB    R5,R4           ;IS ROM PC CORRECT?
3779  026720  001401                          BEQ     6$              ;BR IF YES
3780  026722  104005                          HLT     5               ;ERROR, CRAM PC IS WRONG
3781  026724  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
3782  026726  104400                          SCOPE                   ;SCOPE THIS TEST
3783
3784
3785                                          ;************************** TEST 50 *************************
3786                                          ;*FREE RUNNING FLAG MODE DATA TEST
3787                                          ;*TRANSMIT A MESSAGE AND VERIFY THE RECEIVED DATA
3788                                          ;*IF NO TURNAROUND CONNECTOR IS ON LINE UNIT LOOP IS SET.
3789                                          ;*ALL FOLLOWING TESTS ARE FREE RUNNING AND ARE PERFORMED
3790                                          ;*ONLY ON DMC'S WITH LINE UNITS. IF YOU WISH TO PERFORM
3791                                          ;*THESE FREE RUNNING TESTS ON A KMC (NORMALLY THE FREE RUNNING TESTS
3792                                          ;*WILL FAIL ON A KMC, THE TIMER IS TOO FAST) THEN YOU MUST
3793                                          ;*MANUALLY SET BIT0 OF STAT3 IN THE STATUS MAP.
3794                                          ;;*********************************************************
3795
3796                                          ;  TEST 50
3797                                          ;--------------
3798  026730  012737  000050  001226  TST50:  MOV     #50,TSTNO
3799  026736  012737  027742  001216          MOV     #TST51,NEXT
3800                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
3801  026744  104412                          MSTCLR                  ;MASTER CLEAR DMC11
3802  026746  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT A DMC?
3803  026754  001406                          BEQ     .+16            ;BR IF YES
3804  026756  032737  000001  001372          BIT     #BIT0,STAT3     ;KMC WITH BIT0 SET?
3805  026764  001002                          BNE     .+6             ;BR IF YES
3806  026766  000137  027740                  JMP     14$             ;SKIP TEST
3807  026772  032737  010000  001366          BIT     #BIT12,STAT1    ;LU PRESENT?
3808  027000  001372                          BNE     .-12            ;BR IF NO
3809  027002  004737  035602                  JSR     PC,WROM         ;WRITE MAP IN CRAM
3810  027006  013700  034760                  MOV     RCOUNT,R0       ;CLEAR RECEIVER BUFFER
3811  027012  062700  000002                  ADD     #2,R0           ;CLEAR 2 MORE LOCATIONS
3812  027016  012702  034762                  MOV     #RBUF,R2        ;CLEAR OUT RECEIVE BUFFER
3813  027022  105022                  10$:    CLRB    (R2)+           ;CLEAR BUFFER
3814  027024  005300                          DEC     R0              ;DONE YET!
3815  027026  001375                          BNE     10$             ;NO
3816  027030  005037  034706                  CLR     TFLAG           ;SET TFLAG TO 0
3817  027034  005037  034710                  CLR     RFLAG           ;SET RFLAG TO 0
3818  027040  012711  040000                  MOV     #BIT14,(R1)     ;MASTER CLEAR
3819  027044  032737  100000  001366          BIT     #BIT15,STAT1    ;CRAM?
3820  027052  001402                          BEQ     .+6             ;BR IF NO
3821  027054  012711  100000                  MOV     #BIT15,(R1)     ;IF CRAM SET RUN
```



```
3822  027060  105227  000000                  INCB    #0              ;DELAY
3823  027064  001375                          BNE     .-4             ;DELAY
3824  027066  005037  001416                  CLR     TEMP            ;GET SET TO DELAY
3825  027072  005711                  1$:     TST     (R1)            ;RUN SET?
3826  027074  100405                          BMI     .+14            ;BR IF YES
3827  027076  005237  001416                  INC     TEMP            ;INC DELAY
3828  027102  001373                          BNE     1$              ;BR IF NOT DONE
3829  027104  104014                          HLT     14              ;ERROR RUN NOT SET
3830  027106  000771                          BR      1$              ;TRY AGAIN
3831  027110  032737  040000  001366          BIT     #BIT14,STAT1    ;TURNAROUND CONNECTOR?
3832  027116  001002                          BNE     .+6             ;BR IF YES
3833  027120  052711  004000                  BIS     #BIT11,(R1)     ;SET LINE UNIT LOOP
3834  027124  152711  000043                  BISB#43,(R1)            ;BASE I
3835  027130  005037  001416                  CLR     TEMP            ;GET SET TO DELAY
3836  027134  105711                  2$:     TSTB    (R1)            ;RDI SET?
3837  027136  100404                          BMI     .+12            ;BR IF YES
3838  027140  005237  001416                  INC     TEMP            ;INC DELAY
3839  027144  001373                          BNE     2$              ;BR IF NOT DONE
3840  027146  104014                          HLT     14              ;ERROR,RDI NOT SET
3841  027150  012761  035030  000004          MOV     #BASE,4(R1)     ;SET UP BASE ADDRESS
3842  027156  005061  000006                  CLR     6(R1)           ;CLEAR COUNT
3843  027162  142711  000040                  BICB    #40,(R1)        ;CLEAR RQI
3844  027166  005037  001416                  CLR     TEMP            ;GET SET TO DELAY
3845  027172  105711                  3$:     TSTB    (R1)            ;IS RDI GONE?
3846  027174  100020                          BPL     8$              ;BR IF YES
3847  027176  005237  001416                  INC     TEMP            ;INC DELAY
3848  027202  001373                          BNE     3$              ;BR IF NOT DONE
3849  027204  105761  000002                  TSTB    2(R1)           ;IS THERE A CNTL O ERROR
3850  027210  100011                          BPL     18$             ;BR IF NO
3851  027212  016137  000004  001252          MOV     4(R1),TEMP3     ;SAVE SEL4 FOR TYPEOUT
3852  027220  016137  000006  001254          MOV     6(R1),TEMP4     ;SAVE SEL6 FOR TYPEOUT
3853  027226  104016                          HLT     16              ;CNTL O ERROR
3854  027230  000137  027740                  JMP     14$             ;FATAL ERROR STOP
3855  027234  104014                  18$:    HLT     14              ;ERROR RDI STILL SET
3856  027236                          8$:
3857  027236  152711  000041                  BISB    #41,(R1)        ;ASK FOR CNTL I
3858  027242  105711                  64$:    TSTB    (R1)            ;WAIT FOR RDI
3859  027244  100376                          BPL     64$             ;BR IF NOT SET?
3860  027246  005061  000006                  CLR     6(R1)           ;SET FULL DUPLEX
3861  027252  142711  000040                  BICB    #40,(R1)        ;CLEAR RQI
3862  027256  105711                  65$:    TSTB    (R1)            ;RDI UP?
3863  027260  100776                          BMI     65$             ;BR IF YES
3864  027262  152711  000044                  BISB    #44,(R1)        ;REC BA/CC
3865  027266  005037  001416                  CLR     TEMP            ;GET SET TO DELAY
3866  027272  105711                  4$:     TSTB    (R1)            ;IS RDI SET?
3867  027274  100404                          BMI     .+12            ;BR IF YES
3868  027276  005237  001416                  INC     TEMP            ;INC DELAY
3869  027302  001373                          BNE     4$              ;BR IF DELAY NOT DONE
3870  027304  104014                          HLT     14              ;ERROR RDI NOT SET
3871  027306  012761  034762  000004          MOV     #RBUF,4(R1)     ;LOAD REC BA
3872  027314  013761  034760  000006          MOV     RCOUNT,6(R1)    ;LOAD REC COUNT
3873  027322  142711  000040                  BICB    #40,(R1)        ;CLEAR RQI
3874  027326  005037  001416                  CLR     TEMP            ;GET SET TO DELAY
3875  027332  105711                  5$:     TSTB    (R1)            ;RDI GONE?
3876  027334  100004                          BPL     .+12            ;BR IF YES
3877  027336  005237  001416                  INC     TEMP            ;INC DELAY
```

```
3878  027342  001373                          BNE     5$              ;BR IF NO DONE
3879  027344  104014                          HLT     14              ;ERROR RDI STILL SET
3880  027346  152711  000040                  BISB    #40,(R1)        ;XMIT BA/CC
3881  027352  005037  001416                  CLR     TEMP            ;GET SET TO DELAY
3882  027356  105711                  6$:     TSTB    (R1)            ;RDI SET?
3883  027360  100404                          BMI     .+12            ;BR IF YES
3884  027362  005237  001416                  INC     TEMP            ;INC DELAY
3885  027366  001373                          BNE     6$              ;BR IF NOT DONE
3886  027370  104014                          HLT     14              ;ERROR RDI NOT SET
3887  027372  012761  034714  000004          MOV     #TBUF,4(R1)     ;LOAD XMIT BUFFER
3888  027400  013761  034712  000006          MOV     TCOUNT,6(R1)    ;LOAD COUNT
3889  027406  142711  000040                  BICB    #40,(R1)        ;CLEAR RQI
3890  027412  005037  001416                  CLR     TEMP            ;GET SET TO DELAY
3891  027416  105711                  7$:     TSTB    (R1)            ;RDI GONE?
3892  027420  100004                          BPL     .+12            ;BR IF YES
3893  027422  005237  001416                  INC     TEMP            ;INC DELAY
3894  027426  001373                          BNE     7$              ;BR IF NOT DONE DELAY
3895  027430  104014                          HLT     14              ;ERROR RDI STILL SET
3896  027432  005037  001416          16$:    CLR     TEMP            ;GET SET TO DELAY
3897  027436  012737  000022  001246          MOV     #22,TEMP1       ;GET SET FOR LONG DELAY
3898  027444  105761  000002          11$:    TSTB    2(R1)           ;RDO SET?
3899  027450  100407                          BMI     17$             ;BR IF YES
3900  027452  005237  001416                  INC     TEMP            ;INC DELAY
3901  027456  001372                          BNE     11$             ;BR IF DELAY NOT DONE
3902  027460  005337  001246                  DEC     TEMP1           ;DEC DELAY COUNT
3903  027464  001367                          BNE     11$             ;BR IF NOT DONE DELAY
3904  027466  104014                          HLT     14              ;ERROR RDO NOT SET
3905  027470  016137  000002  001250  17$:    MOV     2(R1),TEMP2     ;SAVE SEL2
3906  027476  001001                          BNE     .+4             ;BR IF OK
3907  027500  104014                          HLT     14              ;ERROR!!! SEL2 = 0!!!!!!
3908  027502  032761  000004  000002          BIT     #BIT2,2(R1)     ;REC OR XMIT?
3909  027510  001032                          BNE     13$             ;BR IF REC
3910  027512  005737  034706          12$:    TST     TFLAG           ;FIRST TIME HERE?
3911  027516  001401                          BEQ     .+4             ;BR IF YES
3912  027520  104014                          HLT     14              ;ERROR MULTIPLE XMIT DONES
3913  027522  012737  177777  034706          MOV     #-1,TFLAG       ;SET TFLAG TO -1
3914  027530  132761  000001  000002          BITB    #BIT0,2(R1)     ;IS IT CONTROL 0
3915  027536  001401                          BEQ     .+4             ;BR IF NO
3916  027540  104014                          HLT     14              ;XMIT ERROR
3917  027542  022761  034714  000004          CMP     #TBUF,4(R1)     ;XMIT BA CORRECT?
3918  027550  001401                          BEQ     .+4             ;BR IF YES
3919  027552  104014                          HLT     14              ;XMIT BA ERROR
3920  027554  023761  034712  000006          CMP     TCOUNT,6(R1)    ;COUNT OK?
3921  027562  001401                          BEQ     .+4             ;BR IF YES
3922  027564  104014                          HLT     14              ;XMIT COUNT ERROR
3923  027566  142761  000207  000002          BICB    #207,2(R1)      ;CLEAR RDO AND BITS 0-2
3924  027574  000453                          BR      15$             ;CONTINUE
3925  027576  005737  034710          13$:    TST     RFLAG           ;FIRST TIME HERE?
3926  027602  001401                          BEQ     .+4             ;BR IF YES
3927  027604  104014                          HLT     14              ;ERROR MULTIPLE REC DONES
3928  027606  012737  177777  034710          MOV     #-1,RFLAG       ;SET RFLAG TO -1
3929  027614  132761  000001  000002          BITB    #BIT0,2(R1)     ;IS IT CNTL 0
3930  027622  001401                          BEQ     .+4             ;BR IF NO
3931  027624  104014                          HLT     14              ;RECEIVE ERROR
3932  027626  022761  034762  000004          CMP     #RBUF,4(R1)     ;REC BA CORRECT?
3933  027634  001401                          BEQ     .+4             ;BR IF YES
```



```
3934  027636  104014                          HLT     14              ;REC BA ERROR
3935  027640  023761  034760  000006          CMP     RCOUNT,6(R1)    ;COUNT OK?
3936  027646  001401                          BEQ     .+4             ;BR IF YES
3937  027650  104014                          HLT     14              ;REC COUNT ERROR
3938  027652  013700  034760                  MOV     RCOUNT,R0       ;GET SET TO CHECK DATA
3939  027656  012702  034714                  MOV     #TBUF,R2        ;R2 POINTS TO GOOD DATA
3940  027662  012703  034762                  MOV     #RBUF,R3        ;R3 POINTS TO RECEIVE DATA
3941  027666  010337  001252          9$:     MOV     R3,TEMP3        ;SAVE ADDRESS FOR TYPEOUT
3942  027672  112205                          MOVB    (R2)+,R5        ;R5 = XMIT DATA
3943  027674  112304                          MOVB    (R3)+,R4        ;R4 = RECEIVE DATA
3944  027676  120504                          CMPB    R5,R4           ;CHECK DATA
3945  027700  001401                          BEQ     .+4             ;BR IF OK
3946  027702  104013                          HLT     13              ;DATA ERROR
3947  027704  005300                          DEC     R0              ;DEC COUNT
3948  027706  001367                          BNE     9$              ;BR IF NOT DONE
3949  027710  005713                          TST     (R3)            ;THIS SHOULD BE 0, ELSE
3950  027712  001401                          BEQ     .+4             ;IT RECEIVED TO MUCH!!
3951  027714  104014                          HLT     14              ;ERROR
3952  027716  142761  000207  000002          BICB    #207,2(R1)      ;CLEAR RDO AND BITS 0-2
3953  027724  005737  034710          15$:    TST     RFLAG           ;REC DONE?
3954  027730  001640                          BEQ     16$             ;BR IF NO
3955  027732  005737  034706                  TST     TFLAG           ;XMIT DONE?
3956  027736  001635                          BEQ     16$             ;BR IF NO
3957  027740  104400                  14$:    SCOPE                   ;SCOPE THIS TEST
3958
3959
3960                                          ;*************************** TEST 51 ****************************
3961                                          ;*OVERUN TEST
3962                                          ;*IN FREE RUNNING MODE SEND MESSAGE WITH NO RECEIVE
3963                                          ;*BUFFER AVAILABLE, VERIFY THAT AN OVERRUN ERROR OCCURS
3964                                          ;;*************************************************************
3965
3966                                          ;  TEST 51
3967                                          ;--------------
3968  027742  012737  000051  001226  TST51:  MOV     #51,TSTNO
3969  027750  012737  030170  001216          MOV     #TST52,NEXT
3970                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
3971  027756  104412                          MSTCLR                  ;MASTER CLEAR DMC11
3972  027760  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT A DMC?
3973  027766  001406                          BEQ     .+16            ;BR IF YES
3974  027770  032737  000001  001372          BIT     #BIT0,STAT3     ;KMC WITH BIT0 SET?
3975  027776  001002                          BNE     .+6             ;BR IF YES
3976  030000  000137  030152                  JMP     10$             ;SKIP TEST
3977  030004  032737  010000  001366          BIT     #BIT12,STAT1    ;LU PRESENT?
3978  030012  001372                          BNE     .-12            ;BR IF NO
3979  030014  004737  035602                  JSR     PC,WROM         ;WRITE MICRO-CODE IN CRAM
3980  030020  004737  036002                  JSR     PC,BASELD               ;LOAD DMC BASE ADDRESS
3981  030024  004537  036272                  JSR     R5,XFRELD       ;LOAD XMIT BA/CC
3982  030030  034714                          TBUF                    ;BA
3983  030032  000044                          44                      ;CC
3984  030034  012700  000010                  MOV     #10,R0          ;R0 = RETRANSMISSION COUNT
3985  030040  012703  000010                  MOV     #10,R3          ;DELAY COUNT
3986  030044  005037  001416                  CLR     TEMP            ;CLEAR DELAY COUNTER
3987  030050  105761  000002          18$:    TSTB    2(R1)           ;IS RDY 0 SET?
3988  030054  100407                          BMI     .+20            ;BR IF SET
3989  030056  005237  001416                  INC     TEMP            ;INC DELAY COUNTER
```

```
3990  030062  001372                          BNE     1$              ;BR IF NOT DONE DELAY
3991  030064  005303                          DEC     R3              ;DEC DELAY COUNT
3992  030066  001370                          BNE     1$              ;BR IF DELAY NOT DONE
3993  030070  104014                          HLT     14              ;ERROR, RDY 0 NOT SET
3994  030072  000427                          BR      10$             ;GET OUT
3995  030074  132761  000001  000002          BITB    #BIT0,2(R1)     ;IS IT CNTL 0?
3996  030102  001002                          BNE     11$             ;BR IF YES
3997  030104  104014                          HLT     14              ;ERROR, NOT CNTL 0
3998  030106  000421                          BR      10$             ;CONTINUE
3999  030110  012705  000004          11$:    MOV     #BIT2,R5        ;PUT "EXPECTED" IN R5
4000  030114  016104  000006                  MOV     6(R1),R4        ;PUT "FOUND" IN R4
4001  030120  020504                          CMP     R5,R4           ;IS ORUN SET?
4002  030122  001404                          BEQ     12$             ;BR IF YES
4003  030124  022704  000001                  CMP     #1,R4           ;DATA CK ERROR?
4004  030130  001411                          BEQ     13$             ;BR IF YES
4005  030132  104015                          HLT     15              ;ERROR, ORUN NOT SET
4006  030134  042761  000207  000002  12$:    BIC     #207,2(R1)      ;CLEAR RDO
4007  030142  005037  001416                  CLR     TEMP            ;RESET DELAY
4008  030146  005300                          DEC     R0              ;DEC RETRANS COUNT
4009  030150  001337                          BNE     1$              ;COUNTINUE
4010  030152  104400                  10$:    SCOPE                   ;SCOPE THIS TEST
4011  030154  042761  000207  000002  13$:    BIC     #207,2(R1)      ;IGNOR THIS ERROR
4012  030162  005037  001416                  CLR     TEMP            ;RESET DELAY
4013  030166  000730                          BR      1$              ;CONTINUE
4014
4015
4016                                          ;************************** TEST 52 **************************
4017                                          ;*LOST DATA TEST
4018                                          ;*IN FREE RUNNING MODE SEND A MESSAGE LONGER THAN THE RECEIVE
4019                                          ;*BUFFER, VERIFY THAT A LOST DATA ERROR OCCURS.
4020                                          ;!!*********************************************************
4021
4022                                          ;  TEST 52
4023                                          ;--------------
4024  030170  012737  000052  001226  TST52:  MOV     #52,TSTNO
4025  030176  012737  030362  001216          MOV     #TST53,NEXT
4026                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
4027  030204  104412                          MSTCLR                  ;MASTER CLEAR DMC11
4028  030206  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT A DMC?
4029  030214  001406                          BEQ     .+16            ;BR IF YES
4030  030216  032737  000001  001372          BIT     #BIT0,STAT3     ;KMC WITH BIT0 SET?
4031  030224  001002                          BNE     .+6             ;BR IF YES
4032  030226  000137  030360                  JMP     10$             ;SKIP TEST
4033  030232  032737  010000  001366          BIT     #BIT12,STAT1    ;LU PRESENT?
4034  030240  001372                          BNE     .-12            ;BR IF NO
4035  030242  004737  035602                  JSR     PC,WROM         ;WRITE MICRO-CODE IN CRAM
4036  030246  004737  036002                  JSR     PC,BASELD               ;LOAD DMC BASE ADDRESS
4037  030252  004537  036240                  JSR     R5,RFRELD       ;LOAD RECEIVE BA/CC
4038  030256  034762                          RBUF                    ;BA
4039  030260  000020                          20                      ;CC
4040  030262  004537  036272                  JSR     R5,XFRELD       ;LOAD XMIT BA/CC
4041  030266  034714                          TBUF                    ;BA
4042  030270  000044                          44                      ;CC
4043  030272  012703  000010                  MOV     #10,R3          ;DELAY COUNT
4044  030276  005037  001416                  CLR     TEMP            ;CLEAR DELAY COUNTER
4045  030302  105761  000002          1$:     TSTB    2(R1)           ;IS RDY 0 SET?
```



```
4046  030306  100407                          BMI     .+20            ;BR IF SET
4047  030310  005237  001416                  INC     TEMP            ;INC DELAY COUNTER
4048  030314  001372                          BNE     1$              ;BR IF NOT DONE DELAY
4049  030316  005303                          DEC     R3              ;DEC DELAY COUNT
4050  030320  001370                          BNE     1$              ;BR IF DELAY NOT DONE
4051  030322  104014                          HLT     14              ;ERROR, RDY 0 NOT SET
4052  030324  000415                          BR      10$             ;GET OUT
4053  030326  132761  000001  000002          BITB    #BIT0,2(R1)     ;IS IT CNTL 0?
4054  030334  001002                          BNE     11$             ;BR IF YES
4055  030336  104014                          HLT     14              ;ERROR NOT CNTL 0
4056  030340  000407                          BR      10$             ;CONTINUE
4057  030342  012705  000020          11$:    MOV     #BIT4,R5        ;PUT "EXPECTED" IN R5
4058  030346  016104  000006                  MOV     6(R1),R4        ;PUT "FOUND" IN R4
4059  030352  020504                          CMP     R5,R4           ;IS LOST DATA SET?
4060  030354  001401                          BEQ     10$             ;BR IF YES
4061  030356  104015                          HLT     15              ;ERROR, LOST DATA NOT SET
4062  030360  104400                  10$:    SCOPE                   ;SCOPE THIS TEST
4063
4064
4065                                          ;************************** TEST 53 **************************
4066                                          ;*TRANSMIT NON-EXISTENT MEMORY TEST
4067                                          ;*IN FREE RUNNING MODE, LOAD A TRANSMIT BA THAT WILL TIME OUT
4068                                          ;*VERIFY THAT A NON-EXISTENT MEMORY ERROR OCCURS
4069                                          ;!!*********************************************************
4070
4071                                          ;  TEST 53
4072                                          ;--------------
4073  030362  012737  000053  001226  TST53:  MOV     #53,TSTNO
4074  030370  012737  030544  001216          MOV     #TST54,NEXT
4075                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
4076  030376  104412                          MSTCLR                  ;MASTER CLEAR DMC11
4077  030400  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT A DMC?
4078  030406  001406                          BEQ     .+16            ;BR IF YES
4079  030410  032737  000001  001372          BIT     #BIT0,STAT3     ;KMC WITH BIT0 SET?
4080  030416  001002                          BNE     .+6             ;BR IF YES
4081  030420  000137  030542                  JMP     10$             ;SKIP TEST
4082  030424  032737  010000  001366          BIT     #BIT12,STAT1    ;LU PRESENT?
4083  030432  001372                          BNE     .-12            ;BR IF NO
4084  030434  004737  035602                  JSR     PC,WROM         ;WRITE MICRO-CODE IN CRAM
4085  030440  004737  036002                  JSR     PC,BASELD               ;LOAD DMC BASE ADDRESS
4086  030444  004537  036272                  JSR     R5,XFRELD       ;LOAD XMIT BA/CC
4087  030450  177320                          177320                  ;BA
4088  030452  140044                          140044                  ;CC
4089  030454  012703  000010                  MOV     #10,R3          ;DELAY COUNT
4090  030460  005037  001416                  CLR     TEMP            ;CLEAR DELAY COUNTER
4091  030464  105761  000002          1$:     TSTB    2(R1)           ;IS RDY 0 SET?
4092  030470  100407                          BMI     .+20            ;BR IF SET
4093  030472  005237  001416                  INC     TEMP            ;INC DELAY COUNTER
4094  030476  001372                          BNE     1$              ;BR IF NOT DONE DELAY
4095  030500  005303                          DEC     R3              ;DEC DELAY COUNT
4096  030502  001370                          BNE     1$              ;BR IF DELAY NOT DONE
4097  030504  104014                          HLT     14              ;ERROR, RDY 0 NOT SET
4098  030506  000415                          BR      10$             ;GET OUT
4099  030510  132761  000001  000002          BITB    #BIT0,2(R1)     ;IS IT CNTL 0?
4100  030516  001002                          BNE     11$             ;BR IF YES
4101  030520  104014                          HLT     14              ;ERROR, NOT CNTL 0
```

```
4102  030522  000407                          BP      10$             ;CONTINUE
4103  030524  012705  000400          11$:    MOV     #BIT8,R5        ;PUT "EXPECTED" IN R5
4104  030530  016104  000006                  MOV     6(R1),R4        ;PUT "FOUND" IN R4
4105  030534  020504                          CMP     R5,R4           ;IS NON-EX-MEM SET?
4106  030536  001401                          BEQ     .+4             ;BR IF YES
4107  030540  104015                          HLT     15              ;ERROR NON-EX-MEM NOT SET
4108  030542  104400                  10$:    SCOPE                   ;SCOPE THIS TEST
4109
4110
4111                                          ;*************************** TEST 54 ***************************
4112                                          ;*RECEIVE NON-EXISTENT MEMORY TEST
4113                                          ;*IN FREE RUNNING MODE, LOAD A RECEIVE BA THAT WILL TIME OUT
4114                                          ;*VERIFY THAT A NON-EXISTENT MEMORY ERROR OCCURS
4115                                          ;!************************************************************
4116
4117                                          ;  TEST 54
4118                                          ;--------------
4119  030544  012737  000054  001226  TST54:  MOV     #54,TSTNO
4120  030552  012737  030736  001216          MOV     #TST55,NEXT
4121                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
4122  030560  104412                          MSTCLR                  ;MASTER CLEAR DMC11
4123  030562  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT A DMC?
4124  030570  001406                          BEQ     .+16            ;BR IF YES
4125  030572  032737  000001  001372          BIT     #BIT0,STAT3     ;KMC WITH BIT0 SET?
4126  030600  001002                          BNE     .+6             ;BR IF YES
4127  030602  000137  030734                  JMP     10$             ;SKIP TEST
4128  030606  032737  010000  001366          BIT     #BIT12,STAT1    ;LU PRESENT?
4129  030614  001372                          BNE     .-12            ;BR IF NO
4130  030616  004737  035602                  JSR     PC,WROM         ;WRITE MICRO-CODE IN CRAM
4131  030622  004737  036002                  JSR     PC,BASELD               ;LOAD DMC BASE ADDRESS
4132  030626  004537  036240                  JSR     R5,RFRELD       ;LOAD RECEIVE BA/CC
4133  030632  177320                          177320                  ;BA
4134  030634  140044                          140044                  ;CC
4135  030636  004537  036272                  JSR     R5,XFRELD       ;LOAD XMIT BA/CC
4136  030642  034714                          TBUF                    ;BA
4137  030644  000044                          44                      ;CC
4138  030646  012703  000010                  MOV     #10,R3          ;DELAY COUNT
4139  030652  005037  001416                  CLR     TEMP            ;CLEAR DELAY COUNTER
4140  030656  105761  000002          1$:     TSTB    2(R1)           ;IS RDY 0 SET?
4141  030662  100407                          BMI     .+20            ;BR IF SET
4142  030664  005237  001416                  INC     TEMP            ;INC DELAY COUNTER
4143  030670  001372                          BNE     1$              ;BR IF NOT DONE DELAY
4144  030672  005303                          DEC     R3              ;DEC DELAY COUNT
4145  030674  001370                          BNE     1$              ;BR IF DELAY NOT DONE
4146  030676  104014                          HLT     14              ;ERROR, RDY 0 NOT SET
4147  030700  000415                          BR      10$             ;GET OUT
4148  030702  132761  000001  000002          BITB    #BIT0,2(R1)     ;IS IT CNTL 0?
4149  030710  001002                          BNE     11$             ;BR IF YES
4150  030712  104014                          HLT     14              ;ERROR, NOT CNTL 0
4151  030714  000407                          BR      10$             ;CONTINUE
4152  030716  012705  000400          11$:    MOV     #BIT8,R5        ;PUT "EXPECTED" IN R5
4153  030722  016104  000006                  MOV     6(R1),R4        ;PUT "FOUND" IN R4
4154  030726  020504                          CMP     R5,R4           ;IS NON-EX-MEM SET?
4155  030730  001401                          BEQ     .+4             ;BR IF YES
4156  030732  104015                          HLT     15              ;ERROR NON-EX-MEM NOT SET
4157  030734  104400                  10$:    SCOPE                   ;SCOPE THIS TEST
```



```
4158
4159
4160                                          ;*************************** TEST 55 ***************************
4161                                          ;*PROCESSOR ERROR TEST
4162                                          ;*IN FREE RUNNING MODE, DO A BASE TRANSFER REQUEST AFTER A
4163                                          ;*BASE HAS BEEN SET UP, VERIFY THAT A PROCESSOR ERROR OCCURS,
4164                                          ;!************************************************************
4165
4166                                          ;  TEST 55
4167                                          ;--------------
4168  030736  012737  000055  001226  TST55:  MOV     #55,TSTNO
4169  030744  012737  031114  001216          MOV     #TST56,NEXT
4170                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
4171  030752  104412                          MSTCLR                  ;MASTER CLEAR DMC11
4172  030754  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT A DMC?
4173  030762  001406                          BEQ     .+16            ;BR IF YES
4174  030764  032737  000001  001372          BIT     #BIT0,STAT3     ;KMC WITH BIT0 SET?
4175  030772  001002                          BNE     .+6             ;BR IF YES
4176  030774  000137  031112                  JMP     10$             ;SKIP TEST
4177  031000  032737  010000  001366          BIT     #BIT12,STAT1    ;LU PRESENT?
4178  031006  001372                          BNE     .-12            ;BR IF NO
4179  031010  004737  035602                  JSR     PC,WROM         ;WRITE MICRO-CODE IN CRAM
4180  031014  004737  036002                  JSR     PC,BASELD       ;LOAD BASE ADDRESS
4181  031020  152711  000043          12$:    BISB    #43,(R1)        ;2ND BASE REQUEST
4182  031024  105711                          TSTB    (R1)            ;RDI SET?
4183  031026  100376                          BPL     .-2             ;BR IF NO
4184  031030  142711  000040                  BICB    #40,(R1)        ;CLEAR PQI
4185  031034  005037  001416                  CLR     TEMP            ;GET SET TO DELAY
4186  031040  105761  000002          13$:    TSTB    2(R1)           ;RDO SET?
4187  031044  100405                          BMI     14$             ;BR IF YES
4188  031046  005237  001416                  INC     TEMP            ;INC DELAY
4189  031052  001372                          BNE     13$             ;BR IF NOT DONE DELAY
4190  031054  104014                          HLT     14              ;ERROR, RDO NOT SET
4191  031056  000770                          BR      13$             ;TRY AGAIN
4192  031060  132761  000001  000002  14$:    BITB    #BIT0,2(R1)     ;IS IS CNTL 0?
4193  031066  001002                          BNE     11$             ;BR IF YES
4194  031070  104014                          HLT     14              ;ERROR NOT CNTL 0
4195  031072  000407                          BR      10$             ;CONTINUE
4196  031074  012705  001000          11$:    MOV     #BIT9,R5        ;PUT "EXPECTED" IN R5
4197  031100  016104  000006                  MOV     6(R1),R4        ;PUT "FOUND" IN R4
4198  031104  020504                          CMP     R5,R4           ;IS PROC ERROR SET?
4199  031106  001401                          BEQ     .+4             ;BR IF YES
4200  031110  104015                          HLT     15              ;ERROR, PROC ERROR NOT SET
4201  031112  104400                  10$:    SCOPE                   ;SCOPE THIS TEST
4202
4203
4204                                          ;*************************** TEST 56 ***************************
4205                                          ;*PROCESSOR ERROR TEST
4206                                          ;*IN FREE RUNNING MODE DO A PQI WITH AN ILLEGAL 10 CODE
4207                                          ;*VERIFY THAT A PROCESSOR ERROR OCCURS
4208                                          ;!************************************************************
4209
4210                                          ;  TEST 56
4211                                          ;--------------
4212  031114  012737  000056  001226  TST56:  MOV     #56,TSTNO
4213  031122  012737  031272  001216          MOV     #TST57,NEXT
```

```
                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
031130  104412                          MSTCLR          ;MASTER CLEAR DMC11
031132  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT A DMC?
031140  001406                          BEQ     .+16            ;BR IF YES
031142  032737  000001  001372          BIT     #BIT0,STAT3     ;KMC WITH BIT0 SET?
031150  001002                          BNE     .+6             ;BR IF YES
031152  000137  031270                  JMP     10$             ;SKIP TEST
031156  032737  010000  001366          BIT     #BIT12,STAT1    ;LU PRESENT?
031164  001372                          BNE     .-12            ;BR IF NO
031166  004737  035602                  JSR     PC,WROM         ;WRITE MICRO-CODE IN CRAM
031172  004737  036002                  JSR     PC,BASELD               ;LOAD DMC BASE ADDRESS
031176  152711  000046                  BISB    #46,(R1)        ;RQI AND ILLEGAL CODE
031202  105711                          TSTB    (R1)            ;WAIT FOR RDI
031204  100376                          BPL     .-2             ;BR IF NO RDI
031206  142711  000040                  BICB    #40,(R1)        ;CLEAR RQI
031212  005037  001416                  CLR     TEMP            ;CLEAR COUNTER
031216  105761  000002          1$:     TSTB    2(R1)           ;RDY O SET?
031222  100405                          BMI     .+14            ;BR IF YES
031224  005237  001416                  INC     TEMP            ;BUMP COUNTER DELAY
031230  001372                          BNE     1$              ;BR IF NOT DONE
031232  104014                          HLT     14              ;ERROR NO RDY O
031234  000770                          BR      1$              ;TRY AGAIN
031236  132761  000001  000002          BITB    #BIT0,2(R1)     ;IS IT CNTL O
031244  001002                          BNE     11$             ;BR IF YES
031246  104014                          HLT     14              ;ERROR, NOT CNTL O
031250  000407                          BR      10$             ;CONTINUE
031252  012705  001000          11$:    MOV     #BIT9,R5        ;PUT "EXPECTED" IN R5
031256  016104  000006                  MOV     6(R1),R4        ;PUT "FOUND" IN R4
031262  020504                          CMP     R5,R4           ;IS PROC ERROR SET?
031264  001401                          BEQ     .+4             ;BR IF YES
031266  104015                          HLT     15              ;ERROR PROC ERROR NOT SET
031270  104400                  10$:    SCOPE                   ;SCOPE THIS TEST


                                        ;************************** TEST 57 **************************
                                        ;*HALF DUPLEX TEST
                                        ;*IN FREE RUNNING MODE, SET HALF DUPLEX AND L U LOOP
                                        ;*SEND A MESSAGE AND VERIFY THAT THERE ARE NO DONES
                                        ;!*********************************************************************


                                        ;  TEST 57
                                        ;--------------
031272  012737  000057  001226  TST57:  MOV     #57,TSTNO
031300  012737  031432  001216          MOV     #TST60,NEXT
                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
031306  104412                          MSTCLR          ;MASTER CLEAR DMC11
031310  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT A DMC?
031316  001406                          BEQ     .+16            ;BR IF YES
031320  032737  000001  001372          BIT     #BIT0,STAT3     ;KMC WITH BIT0 SET?
031326  001002                          BNE     .+6             ;BR IF YES
031330  000137  031424                  JMP     10$             ;SKIP TEST
031334  032737  010000  001366          BIT     #BIT12,STAT1    ;LU PRESENT?
031342  001372                          BNE     .-12            ;BR IF NO
031344  004737  035602                  JSR     PC,WROM         ;WRITE MICRO-CODE
031350  004737  036120                  JSR     PC,BASELH       ;LOAD BASE AND HALF DUPLEX
031354  004537  036240                  JSR     R5,RFRELD       ;LOAD RECEIVE BUFFER
```



```
031360  034762                          RBUF            ;BA
031362  000044                          44              ;CC
031364  004537  036272                  JSR     R5,XFRELD       ;LOAD TRANSMIT BUFFER
031370  034714                          TBUF            ;BA
031372  000044                          44              ;CC
031374  012703  000003                  MOV     #3,R3           ;LOAD DELAY COUNT
031400  005037  001416                  CLR     TEMP            ;CLEAR DELAY
031404  105761  000002          4$:     TSTB    2(R1)           ;IS DONE SET?
031410  100406                          BMI     5$              ;BR IF YES (ERROR)
031412  005237  001416                  INC     TEMP            ;INC DELAY
031416  001372                          BNE     4$              ;BR IF DELAY NOT DONE
031420  005303                          DEC     R3              ;DEC DELAY COUNT
031422  001370                          BNE     4$              ;BR IF DELAY NOT DONE
031424  104400                  10$:    SCOPE                   ;SCOPE THIS TEST
031426  104014                  5$:     HLT     14              ;ERROR DONE WITH HALF-DUPLEX
031430  000775                          BR      10$             ;GET OUT


                                        ;************************** TEST 60 **************************
                                        ;*FREE RUNNING DATA TEST (INTERRUPT DRIVEN EXERCISER)
                                        ;*THIS TEST REPEATEDLY QUEUES UP 7 RECEIVE BUFFERS AND
                                        ;*7 TRANSMIT BUFFERS AND CHECKS DATA WHEN ALL 7 BUFFERS
                                        ;*ARE RECEIVED, TRANSMIT COUNTS RANGE FROM 1 TO 104, ALSO
                                        ;*ODD AND EVEN TRANSMIT AND RECEIVE BA'S ARE USED, DATA
                                        ;*IS A BINARY COUNT PATTERN, THE RESUME FUNCTION IS CHECKED IN THIS TEST
                                        ;!*********************************************************************


                                        ;  TEST 60
                                        ;--------------
031432  012737  000060  001226  TST60:  MOV     #60,TSTNO
031440  012737  003274  001216          MOV     #,EOP,NEXT
                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
031446  104412                          MSTCLR          ;MASTER CLEAR DMC11
031450  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT A DMC?
031456  001406                          BEQ     .+16            ;BR IF YES
031460  032737  000001  001372          BIT     #BIT0,STAT3     ;KMC WITH BIT0 SET?
031466  001002                          BNE     .+6             ;BR IF YES
031470  000137  032434                  JMP     ENDEX1          ;SKIP TEST
031474  032737  010000  001366          BIT     #BIT12,STAT1    ;LU PRESENT?
031502  001372                          BNE     .-12            ;BR IF NO
031504  004737  035602                  JSR     PC,WROM         ;WRITE MICR-CODE
031510  012737  000340  177776          MOV     #340,PS         ;LOCK OUT INTERRUPTS
031516  013700  001366                  MOV     STAT1,R0        ;GET BR LEVEL
031522  006200                          ASR     R0              ;SHIFT RIGHT 4 TIMES
031524  006200                          ASR     R0
031526  006200                          ASR     R0
031530  006200                          ASR     R0
031532  042700  177437                  BIC     #177437,R0      ;PUT BR LEVEL IN R0
031536  012777  032534  147630          MOV     #IISR,@DMRVEC   ;LOAD INPUT VECTOR
031544  010077  147626                  MOV     R0,@DMRLVL      ;LOAD LEVEL
031550  012777  033040  147622          MOV     #OISR,@DMTVEC   ;LOAD OUTPUT VECTOR
031556  010077  147620                  MOV     R0,@DMTLVL      ;LOAD LEVEL


                                        ;INITIALIZE ALL BUFFER LISTS AND COUNT LISTS

031562  012737  000104  034706          MOV     #104,TFLAG      ;TFLAG CONTAINS COUNT
```

```
4326  031570  012700  033454                        MOV     #XMITBA+2,R0    ;R0 POINTS TO BA LIST
4327  031574  012703  033746                        MOV     #RBUFF,R3       ;R3 CONTAINS BUFFER ADDRESS
4328  031600  010320                        1$:     MOV     R3,(R0)+        ;LOAD BA LIST WITH REC BA
4329  031602  062703  000104                        ADD     #104,R3         ;UPDATE BUFFER ADDRESS
4330  031606  022700  033472                        CMP     #XMITBA+20,R0   ;END OF REC BUFFERS?
4331  031612  001372                                BNE     1$              ;NO LOAD NEXT ONE
4332  031614  012720  033510                2$:     MOV     #TBUFF,(R0)+    ;LOAD BA LIST WITH XMIT BA
4333  031620  022700  033510                        CMP     #XMITBA+36,R0   ;END OF XMIT BUFFERS?
4334  031624  001373                                BNE     2$              ;NO LOAD NEXT BUFFER
4335  031626  012700  033622                        MOV     #RCNTAB+2,R0    ;R0 POINTS TO COUNT LIST
4336  031632  013720  034706                3$:     MOV     TFLAG,(R0)+     ;LOAD COUNT OF 104
4337  031636  022700  033640                        CMP     #RCNTAB+20,R0   ;END OF REC COUNT LIST?
4338  031642  001373                                BNE     3$              ;BR IF NO
4339  031644  012737  000006  034704                MOV     #6,FLAG ;LOOP COUNT
4340  031652  012711  040000                        MOV     #BIT14,(R1)     ;SET MASTER CLEAR
4341  031656  032737  100000  001366                BIT     #BIT15,$TAT1    ;IOP?
4342  031664  001402                                BEQ     .+6             ;BR IF NO
4343  031666  012711  100000                        MOV     #BIT15,(R1)     ;SET RUN ON IOP
4344  031672  012700  177777                        MOV     #-1,R0          ;R0 IS INPUT DONE COUNTER
4345  031676  005037  033450                CLRTAB: CLR     RESUME          ;CLEAR RESUME FLAG
4346  031702  012705  033656                        MOV     #RDNTAB,R5      ;GET READY TO CLEAR ALL RECEIVE
4347  031706  005025                        2$:     CLR     (R5)+           ;BUFFERS
4348  031710  022705  034702                        CMP     #RBUFFE,R5      ;END OF BUFFER?
4349  031714  001374                                BNE     2$              ;BR IF NO
4350  031716  005737  034704                        TST     FLAG            ;VARIABLE COUNTS?
4351  031722  100407                                BMI     5$              ;BR IF YES(DON'T CHANGE THEM)
4352  031724  012704  033640                        MOV     #XCNTAB,R4      ;R4 POINTS TO XMIT COUNT LIST
4353  031730  013724  034706                4$:     MOV     TFLAG,(R4)+     ;LOAD XMIT CHAR COUNT
4354  031734  022704  033656                        CMP     #XCNTAB+16,R4   ;DONE?
4355  031740  001373                                BNE     4$              ;BR IF NO
4356  031742  005002                        5$:     CLR     R2              ;R2 IS OUTPUT DONE COUNTER
4357  031744  005004                                CLR     R4              ;R4 IS USED AS INDEX IN OISR
4358  031746  005711                                TST     (R1)            ;IS RUN SET?
4359  031750  100376                                BPL     .-2             ;WAIT FOR RUN
4360  031752  152761  000100  000002                BISB    #BIT6,2(R1)     ;SET IEO
4361  031760  022737  000006  034704                CMP     #6,FLAG ;FIRST TIME?
4362  031766  001003                                BNE     1$              ;BR IF NOT
4363  031770  052711  004143                        BIS     #4143,(R1)      ;SET LU LOOP, IEI,RQI,BASE I
4364  031774  000402                                BR      3$              ;CONTINUE
4365  031776  052711  004144                1$:     BIS     #4144,(R1)      ;SET LU LOOP, IEI, RQI, REC BA/CC
4366  032002  005037  001416                3$:     CLR     TEMP            ;SET UP FOR DELAY COUNT
4367  032006  012737  000022  001250                MOV     #22,TEMP2       ;GET SET FOR DELAY
4368  032014  005037  177776                        CLR     P5              ;ALLOW INTERRUPTS
4369  032020  022737  000001  034704        SCAN:   CMP     #1,FLAG         ;1 BYTE MESS?
4370  032026  001002                                BNE     1$              ;BR IF NO
4371  032030  000137  032472                        JMP     ENDEX3          ;BR IF YES
4372  032034  022700  000020                1$:     CMP     #20,R0          ;INPUT DONE?
4373  032040  001402                                BEQ     SCAN2           ;BR IF YES
4374  032042  000137  032504                        JMP     SCAN1           ;BR IF NO
4375  032046  022702  000034                SCAN2:  CMP     #34,R2          ;XMIT DONE FOR ALL MESSAGES?
4376  032052  001402                                BEQ     8$              ;BR IF YES
4377  032054  000137  032504                        JMP     SCAN1           ;BR IF NO
4378  032060  022704  000034                8$:     CMP     #34,R4          ;REC DONE FOR ALL MESSAGES?
4379  032064  001402                                BEQ     9$              ;BR IF YES
4380  032066  000137  032504                        JMP     SCAN1           ;BR IF NO
4381  032072                                9$:
```



```
4382  032072  012700  033656                        MOV     #RDNTAB,R0      ;GET FIRST REC BUFFER
4383  032076  012002                        5$:     MOV     (R0)+,R2        ;R2 POINTS TO BUFFER
4384  032100  005005                                CLR     R5              ;R5=EXPECTED
4385  032102  005003                                CLR     R3              ;R3 = COUNT
4386  032104  005737  034704                        TST     FLAG            ;CHECK FOR ODD XMIT BA'S
4387  032110  100012                                BPL     6$              ;ONLY FOR VARIABLE COUNTS
4388  032112  022710  000027                        CMP     #27,(R0)        ;IF 27 BUMP DATA BY 1 (ODD XMIT BA)
4389  032116  001406                                BEQ     7$              ;BR IF YES
4390  032120  022710  000042                        CMP     #42,(R0)        ;IF 42 THEN ODD XMIT BA ALSO
4391  032124  001403                                BEQ     7$              ;BR IF YES
4392  032126  022710  000103                        CMP     #103,(R0)       ;IF 103 THEN ODD XMIT BA ALSO
4393  032132  001001                                BNE     6$              ;SKIP IF NOT
4394  032134  005205                        7$:     INC     R5              ;START DATA AT 1 FOR ODD XMIT BA'S
4395  032136  010237  001252                6$:     MOV     R2,TEMP3        ;SAVE ADDRESS FOR TYPEOUT
4396  032142  112204                                MOVB    (R2)+,R4        ;GET RECEIVE DATA
4397  032144  120504                                CMPB    R5,R4           ;IS IT CORRECT?
4398  032146  001401                                BEQ     .+4             ;BR IF YES
4399  032150  104013                                HLT     13              ;DATA ERROR
4400  032152  005205                                INC     R5              ;NEXT CHARACTER
4401  032154  005203                                INC     R3              ;INC COUNT
4402  032156  021003                                CMP     (R0),R3         ;DONE YET?
4403  032160  001366                                BNE     6$              ;BR IF NO
4404  032162  062700  000002                        ADD     #2,R0           ;GET NEXT REC BUFFER
4405  032166  022700  033712                        CMP     #RDNTAB+34,R0   ;DONE YET?
4406  032172  001341                                BNE     5$              ;BR IF NO
4407  032174  012700  000001                        MOV     #1,R0           ;SET R0 TO 1
4408  032200  005737  034704                        TST     FLAG            ;VARIABLE COUNTS?
4409  032204  100004                                BPL     4$              ;BR IF NO
4410  032206  005237  034704                        INC     FLAG            ;FLAG IS NEGITIVE
4411  032212  001231                                BNE     CLRTAB          ;BR IF NOT DONE
4412  032214  000447                                BR      ENDEX           ;ALL DONE
4413  032216  032737  000001  034704        4$:     BIT     #BIT0,FLAG      ;CHANGE CHAR COUNT FOR NEXT LOOP
4414  032224  001003                                BNE     1$              ;BR TO SUB 40
4415  032226  005337  034706                        DEC     TFLAG           ;DEC BY ONE
4416  032232  000403                                BR      2$              ;CONTINUE
4417  032234  162737  000040  034706        1$:     SUB     #40,TFLAG       ;SUBTRACT 40 FROM XMIT COUNT
4418  032242  005337  034704                2$:     DEC     FLAG            ;DEC LOOP COUNT
4419  032246  001213                                BNE     CLRTAB          ;GO DO IT AGAIN
4420  032250  005004                                CLR     R4              ;R4 CONTAINS OFFSET
4421  032252  012702  033642                        MOV     #XCNTAB+2,R2    ;R2 POINTS TO XMIT COUNT LIST
4422  032256  062704  000013                3$:     ADD     #13,R4          ;INCREASE R4 BY 13
4423  032262  060422                                ADD     R4,(R2)+        ;MAKE COUNTS VARIABLE
4424  032264  022702  033656                        CMP     #XCNTAB+16,R2   ;DONE ALL 7?
4425  032270  001372                                BNE     3$              ;BR IF NO
4426  032272  012702  033464                        MOV     #RECBA+12,R2    ;R2 POINTS TO REC BA LIST
4427  032276  005222                                INC     (R2)+           ;MAKE THIS REC BA ODD
4428  032300  005222                                INC     (R2)+           ;MAKE THIS REC BA ODD
4429  032302  005222                                INC     (R2)+           ;MAKE THIS REC BA ODD
4430  032304  062702  000004                        ADD     #4,R2           ;SKIP TO XMIT BA LIST
4431  032310  005222                                INC     (R2)+           ;MAKE THIS XMIT BA ODD
4432  032312  005222                                INC     (R2)+           ;MAKE THIS XMIT BA ODD
4433  032314  062702  000004                        ADD     #4,R2           ;SKIP TO NEXT ODD BA
4434  032320  005222                                INC     (R2)+           ;MAKE THIS XMIT BA ODD
4435  032322  012737  177772  034704                MOV     #-6,FLAG        ;MAKE FLAG NEGITIVE
4436  032330  000137  031676                        JMP     CLRTAB          ;LOOP WITH VARIABLE COUNTS
4437  032334  152711  000146                ENDEX:  BISB    #146,(R1)       ;SHUT DOWN DMC
```

```
 4438  032340  005737  034704                  1$:   TST     FLAG            ;HAS INTERRUPT OCCURED?
 4439  032344  001775                              BEQ     1$              ;BR IF NO
 4440  032346  012700  000003                      MOV     #3,R0           ;BASE ADDRESS OFFSET
 4441  032352  105760  035030                  2$:   TSTB    BASE(R0)        ;CHECK ERROR COUNT
 4442  032356  001027                              BNE     ENDEX2          ;BR IF ERROR
 4443  032360  005200                              INC     R0              ;BUMB INDEX
 4444  032362  022700  000005                      CMP     #5,R0           ;5 = NAKS BAD CRC
 4445  032366  001006                              BNE     3$              ;BR IF NOT 5
 4446  032370  122760  000013  035030              CMPB    #13,BASE(R0)    ;SHOULD BE 13 ERRORS
 4447  032376  001017                              BNE     ENDEX2          ;BECAUSE OF RESUME
 4448  032400  005200                              INC     R0              ;BUMP INDEX
 4449  032402  000763                              BR      2$              ;BR
 4450  032404  022700  000011                  3$:   CMP     #11,R0          ;DONE ALL ERROR COUNTERS YET?
 4451  032410  001360                              BNE     2$              ;BR IF NO
 4452  032412  122760  000013  035030              CMPB    #13,BASE(R0)    ;13 ERRORS BECAUSE OF RESUME
 4453  032420  001006                              BNE     ENDEX2          ;BR IF NOT OK
 4454  032422  005200                              INC     R0              ;NEXT BASE TABLE LOCATION
 4455  032424  122760  000013  035030              CMPB    #13,BASE(R0)    ;13 ERRORS BECAUSE OF RESUME
 4456  032432  001001                              BNE     ENDEX2          ;BR IF NOT OK
 4457  032434  104400                          ENDEX1: SCOPE                   ;SCOPE THIS TEST
 4458  032436  113737  035033  001250          ENDEX2: MOVB    BASE+3,TEMP2    ;SAVE ALL ODD ADDRESSES
 4459  032444  113737  035035  001252                  MOVB    BASE+5,TEMP3    ;FOR TYPEOUT
 4460  032452  113737  035037  001254                  MOVB    BASE+7,TEMP4    ;
 4461  032460  113737  035041  001256                  MOVB    BASE+11,TEMP5
 4462  032466  104017                              HLT     17              ;NON ZERO ERROR COUNT
 4463  032470  000761                              BR      ENDEX1          ;GET OUT
 4464  032472  022700  000017                  ENDEX3: CMP     #17,R0          ;ALL DONE INPUT?
 4465  032476  001002                              BNE     SCAN1           ;BR IF NO
 4466  032500  000137  032046                      JMP     SCAN2           ;BR IF YES
 4467  032504  005337  001416                  SCAN1:  DEC     TEMP            ;DECREMENT DELAY COUNTER
 4468  032510  001402                              BEQ     1$              ;BR IF ZERO
 4469  032512  000137  032020                      JMP     SCAN            ;BR IF NOT DONE DELAY
 4470  032516  005337  001250                  1$:   DEC     TEMP2           ;DEC DELAY COUNT
 4471  032522  001402                              BEQ     2$              ;BR IF DONE DELAY
 4472  032524  000137  032020                      JMP     SCAN            ;BR IF NOT DONE
 4473  032530  104014                          2$:   HLT     14              ;ERROR HUNG
 4474  032532  000740                              BR      ENDEX1          ;GET OUT
 4475
 4476                                              ;INPUT INTERRUPT SERVICE ROUTINE
 4477
 4478  032534  022700  000017                  IISR:   CMP     #17,R0          ;PROC. ERROR DONE?
 4479  032540  001421                              BEQ     12$             ;BR IF YES
 4480  032542  005737  033450                      TST     RESUME          ;IS THIS A RESUME INTERRUPT
 4481  032546  001432                              BEQ     8$              ;BR IF NO
 4482  032550  032711  000002                      BIT     #BIT1,(R1)      ;CNTL OR BASE?
 4483  032554  001407                              BEQ     13$             ;BR IF CNTL I
 4484  032556  012761  035030  000004              MOV     #BASE,4(R1)     ;LOAD BASE ADDRESS
 4485  032564  012761  010000  000006              MOV     #BIT12,6(R1)    ;WITH RESUME BIT SET
 4486  032572  000404                              BR      12$             ;CONTINUE
 4487  032574  005061  000006                  13$:  CLR     6(R1)           ;SELECT FULL DUPLEX
 4488  032600  005037  033450                      CLR     RESUME          ;CLEAR RESUME FLAG
 4489  032604  142711  000040                  12$:  BICB    #40,(R1)        ;CLEAR RQI
 4490  032610  105711                              TSTB    (R1)            ;IS RDI GONE?
 4491  032612  100776                              BMI     .-2             ;BR IF NO
 4492  032614  005737  033450                      TST     RESUME          ;BASE OR CNTL I?
 4493  032620  001403                              BEQ     14$             ;BR IF IT WAS CNTL I
```



```
 4494  032622  152711  000041                      BISB    #41,(R1)        ;ASK FOR CNTL I
 4495  032626  000002                              RTI                     ;RETURN
 4496  032630  105011                  14$:  CLRB    (R1)            ;CLEAR BSEL 0
 4497  032632  000002                              RTI                     ;RETURN
 4498  032634  005700                  8$:   TST     R0              ;FIRST TIME HERE?
 4499  032636  100006                              BPL     7$              ;LOAD BASE IF MINUS
 4500  032640  012761  035030  000004              MOV     #BASE,4(R1)     ;SET UP BASE ADDRESS
 4501  032646  005061  000006                      CLR     6(R1)           ;CLEAR COUNT
 4502  032652  000434                              BR      3$              ;CONTINUE
 4503  032654  001003                  7$:   BNE     1$              ;CNTL I FULL DUPLEX IF 0
 4504  032656  005061  000006                      CLR     6(R1)           ;SELECT FULL DUPLEX
 4505  032662  000430                              BR      3$              ;CONTINUE
 4506  032664  032700  000010                  1$:   BIT     #BIT3,R0        ;XMIT?
 4507  032670  001013                              BNE     2$              ;BR IF YES
 4508  032672  000241                              CLC                     ;CLEAR CARRY
 4509  032674  006100                              ROL     R0              ;MAKE R0 EVEN
 4510  032676  016061  033452  000004              MOV     RECBA(R0),4(R1) ;LOAD REC BUFFER
 4511  032704  016061  033620  000006              MOV     RCNTAB(R0),6(R1);LOAD COUNT
 4512  032712  000241                              CLC                     ;CLEAR CARRY
 4513  032714  006000                              ROR     R0              ;GET R0 BACK
 4514  032716  000412                              BR      3$              ;CONTINUE
 4515  032720  000241                  2$:   CLC                     ;CLEAR CARRY
 4516  032722  006100                              ROL     R0              ;MAKE IT EVEN
 4517  032724  016061  033452  000004              MOV     XMITBA(R0),4(R1);LOAD XMIT BUFFER
 4518  032732  016061  033620  000006              MOV     RCNTAB(R0),6(R1);LOAD COUNT
 4519  032740  000241                              CLC                     ;CLEAR CARRY
 4520  032742  006000                              ROR     R0              ;PUT IT BACK
 4521  032744  142711  000040                  3$:   BICB    #40,(R1)        ;CLEAR RQI
 4522  032750  105711                              TSTB    (R1)            ;WAIT FOR
 4523  032752  100776                              BMI     .-2             ;RDI TO GO AWAY
 4524  032754  005200                              INC     R0              ;INC COUNT
 4525  032756  001003                              BNE     6$              ;IF 0 ASK FOR CNTL I
 4526  032760  152711  000041                      BISB    #41,(R1)        ;ASK FOR CNTL I
 4527  032764  000002                              RTI                     ;RETURN
 4528  032766  022700  000017                  6$:   CMP     #17,R0          ;DONE YET?
 4529  032772  001411                              BEQ     4$              ;BR IF YES
 4530  032774  032700  000010                      BIT     #BIT3,R0        ;XMIT?
 4531  033000  001003                              BNE     5$              ;BR IF YES
 4532  033002  152711  000044                      BISB    #44,(R1)        ;ASK FOR REC BA/CC
 4533  033006  000002                              RTI                     ;RETURN
 4534  033010  152711  000040                  5$:   BISB    #40,(R1)        ;ASK FOR XMIT BA/CC
 4535  033014  000002                              RTI                     ;RETURN
 4536  033016  022737  000001  034704          4$:   CMP     #1,FLAG         ;1 BYTE MESS?
 4537  033024  001403                              BEQ     15$             ;BR IF YES
 4538  033026  152711  000046                      BISB    #46,(R1)        ;FORCE PROC. ERROR
 4539  033032  000002                              RTI                     ;RETURN
 4540  033034  105011                  15$:  CLRB    (R1)            ;CLR SEL0
 4541  033036  000002                              RTI                     ;RETURN
 4542
 4543                                              ;OUTPUT INTERRUPT SERVICE ROUTINE
 4544
 4545  033040  032761  000001  000002          OISR:   BIT     #BIT0,2(R1)     ;IS THIS AN ERROR?
 4546  033046  001461                              BEQ     1$              ;BR IF NO
 4547  033050  005737  034704                      TST     FLAG            ;IS THIS SHUT DOWN INTERRUPT?
 4548  033054  001006                              BNE     9$              ;BR IF NO
 4549  033056  005237  034704                      INC     FLAG            ;YES MAKE FLAG NON-ZERO
```

```
4550  033062  022761  001000  000006          CMP     #BIT9,6(R1)     ;SHUT DOWN BIT SET?
4551  033070  001516                          BEQ     10$             ;YES ALL IS OK
4552  033072  022700  000017          9$:     CMP     #17,R0          ;RESUME INTERRUPT?
4553  033076  001033                          BNE     11$             ;BR IF NO
4554  033100  022761  001000  000006          CMP     #BIT9,6(R1)     ;PROC. ERROR BIT SET?
4555  033106  001027                          BNE     11$             ;BR IF NO
4556  033110  005200                          INC     R0              ;BUMP COUNTER (TO 20)
4557  033112  012711  040000                  MOV     #BIT14,(R1)     ;MASTER CLEAR DEVICE
4558  033116  032737  100000  001366          BIT     #BIT15,STAT1    ;DMC OR KMC?
4559  033124  001405                          BEQ     .+14            ;BR IF DMC
4560  033126  012711  100000                  MOV     #BIT15,(R1)     ;SET RUN ON KMC
4561  033132  105227  000000                  INCB    #0              ;DELAY ON KMC
4562  033136  001375                          BNE     .-4
4563  033140  012737  177777  033450          MOV     #-1,RESUME      ;SET RESUME FLAG
4564  033146  005711                          TST     (R1)            ;RUN SET?
4565  033150  100376                          BPL     .-2             ;BR IF NO
4566  033152  012761  000100  000002          MOV     #BIT6,2(R1)     ;SET IEO
4567  033160  052711  004143                  BIS     #4143,(R1)      ;ASK FOR PORT(BASE REQ)
4568  033164  000002                          RTI                     ;RETURN
4569  033166  016137  000004  001252  11$:     MOV     4(R1),TEMP3     ;SAVE FOR ERROR TYPEOUT
4570  033174  016137  000006  001254          MOV     6(R1),TEMP4     ;SAVE FOR ERROR TYPEOUT
4571  033202  104016                          HLT     16              ;CNTL O ERROR
4572  033204  022626                          CMP     (SP)+,(SP)+     ;ADJUST STACK
4573  033206  000137  032434                  JMP     ENDEX1          ;GET OUT
4574  033212  032761  000004  000002  1$:     BIT     #BIT2,2(R1)     ;RECEIVE?
4575  033220  001046                          BNE     2$              ;BR IF YES
4576  033222  022761  033511  000004          CMP     #TBUFF+1,4(R1)  ;XMIT BA CORRECT?
4577  033230  001405                          BEQ     4$              ;BR IF OK
4578  033232  022761  033510  000004          CMP     #TBUFF,4(R1)    ;XMIT BA CORRECT?
4579  033240  001401                          BEQ     4$              ;BR IF YES
4580  033242  104014                          HLT     14              ;XMIT BA ERROR
4581  033244  005005                  4$:     CLR     R5              ;R5 IS INDEX REG
4582  033246  026561  033640  000006  5$:     CMP     XCNTAB(R5),6(R1);IS CHAR COUNT OK?
4583  033254  001406                          BEQ     6$              ;BR IF YES
4584  033256  062705  000002                  ADD     #2,R5           ;INC INDEX
4585  033262  022705  000016                  CMP     #16,R5          ;DONE LIST YET?
4586  033266  001367                          BNE     5$              ;BR IF NO
4587  033270  104014                          HLT     14              ;XMIT COUNT ERROR
4588  033272  016162  000004  033712  6$:     MOV     4(R1),XDNTAB(R2);STORE XMIT DONE BA
4589  033300  062702  000002                  ADD     #2,R2           ;INC INDEX
4590  033304  016162  000006  033712          MOV     6(R1),XDNTAB(R2);STORE XMIT DONE CC
4591  033312  062702  000002                  ADD     #2,R2           ;INC INDEX
4592  033316  142761  000207  000002          BICB    #207,2(R1)      ;CLEAR RDO
4593  033324  000002                          RTI                     ;RETURN
4594  033326  105011                  10$:    CLRB    (R1)            ;CLEAR SEL0
4595  033330  105061  000002                  CLRB    2(R1)           ;CLEAR SEL2
4596  033334  000002                          RTI                     ;RETURN
4597  033336  012705  000002          2$:     MOV     #2,R5           ;SET UP R5 AS INDEX
4598  033342  026561  033452  000004          CMP     RECBA(R5),4(R1) ;COMPARE WITH LIST OF CORRECT BA'S
4599  033350  001406                          BEQ     3$              ;BR IF OK?
4600  033352  062705  000002                  ADD     #2,R5           ;INCREMENT R5
4601  033356  022705  000020                  CMP     #20,R5          ;END OF LIST?
4602  033362  001367                          BNE     2$+4            ;BR IF NO
4603  033364  104014                          HLT     14              ;REC BA ERROR
4604  033366  005005                  3$:     CLR     R5              ;R5 IS INDEX
4605  033370  026561  033640  000006  7$:     CMP     XCNTAB(R5),6(R1);CHECK FOR CORRECT REC COUNT
```



```
4606  033376  001406                          BEQ     8$              ;BR IF YES
4607  033400  062705  000002                  ADD     #2,R5           ;INCREMENT R5
4608  033404  022705  000016                  CMP     #16,R5          ;END OF LIST?
4609  033410  001367                          BNE     7$              ;BR IF NOT
4610  033412  104014                          HLT     14              ;REC COUNT ERROR
4611  033414  016164  000004  033656  8$:     MOV     4(R1),RDNTAB(R4)  ;STORE REC BA
4612  033422  062704  000002                  ADD     #2,R4           ;INC INDEX
4613  033426  016164  000006  033656          MOV     6(R1),RDNTAB(R4);STORE REC DONE CC
4614  033434  062704  000002                  ADD     #2,R4           ;INC INDEX
4615  033440  142761  000207  000002          BICB    #207,2(R1)      ;CLEAR RDO
4616  033446  000002                          RTI                     ;RETURN
4617
4618
4619                                          ;BUFFERS
4620
4621  033450  000000                  RESUME: 0
4622  033452                          RECBA:
4623  033452  000017                  XMITBA: .BLKW   17      ;REC & XMIT BA LIST
4624
4625  033510                          TBUFF:                  ;TRANSMIT DATA
4626  033510      000     001     002         .BYTE   0,1,2,3,4,5,6,7
4627  033513      003     004     005
4628  033516      006     007
4629  033520      010     011     012         .BYTE   10,11,12,13,14,15,16,17
4630  033523      013     014     015
4631  033526      016     017
4632  033530      020     021     022         .BYTE   20,21,22,23,24,25,26,27
4633  033533      023     024     025
4634  033536      026     027
4635  033540      030     031     032         .BYTE   30,31,32,33,34,35,36,37
4636  033543      033     034     035
4637  033546      036     037
4638  033550      040     041     042         .BYTE   40,41,42,43,44,45,46,47
4639  033553      043     044     045
4640  033556      046     047
4641  033560      050     051     052         .BYTE   50,51,52,53,54,55,56,57
4642  033563      053     054     055
4643  033566      056     057
4644  033570      060     061     062         .BYTE   60,61,62,63,64,65,66,67
4645  033573      063     064     065
4646  033576      066     067
4647  033600      070     071     072         .BYTE   70,71,72,73,74,75,76,77
4648  033603      073     074     075
4649  033606      076     077
4650  033610      100     101     102         .BYTE   100,101,102,103,104,105,106,107
4651  033613      103     104     105
4652  033616      106     107
4653
4654  033620  000010                  RCNTAB: .BLKW   10              ;RECEIVE COUNT TABLE
4655  033640  000007                  XCNTAB: .BLKW   7               ;TRANSMIT COUNT TABLE
4656
4657  033656  000016                  RDNTAB: .BLKW   16              ;RECEIVE DONE TABLE (BA/CC)
4658  033712  000016                  XDNTAB: .BLKW   16              ;XMIT DONE TABLE (BA/CC)
4659
4660  033746                          RBUFF:                  ;RECEIVER BUFFERS
4661  033746  000104                  RBUFF1: .BLKB 104
```

```
4662  034052  000104                          RBUFF2:  .BLKB 104
4663  034156  000104                          RBUFF3:  .BLKB 104
4664  034262  000104                          RBUFF4:  .BLKB 104
4665  034366  000104                          RBUFF5:  .BLKB 104
4666  034472  000104                          RBUFF6:  .BLKB 104
4667  034576  000104                          RBUFF7:  .BLKB 104
4668  034702  000000                          RBUFFE:  0                      ;END OF RECEIVER BUFFERS
4669                                  06900
4670                                  07000
4671                                  07100
4672                                  07200           ;BUFFER AREA
4673                                  07300           ;-----------
4674                                  07400
4675  034704  000000                  07500   FLAG:   0
4676  034706  000000                  07600   TFLAG:  0
4677  034710  000000                  07700   RFLAG:  0
4678  034712  000044                  07800   TCOUNT: 44
4679  034714  041101  042103  043105  07900   TBUF:   .ASCII/ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789/
4680  034722  044107  045111  046113
4681  034730  047115  050117  051121
4682  034736  052123  053125  054127
4683  034744  055131  030460  031462
4684  034752  032464  033466  034470
4685                                  08000   .EVEN
4686  034760  000044                  08100   RCOUNT: 44
4687  034762  035030                  08200   RBUF:   .=.+46
4688                                  08300   .EVEN
4689  035030  035430                  08400   BASE:   .=.+256.
4690                                  00300
4691                                  00400
4692                                  00500   ;SUBROUTINES
4693                                  00600   ;-----------
4694                                  00700
4695  035430                          00800   CLRALL:
4696                                  00900           ;THIS SUBROUTINE CLEARS THE C&Z BITS AND THE BR
4697                                  01000
4698  035430  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4699  035432  000400                  01200   000400                  ;BR_0
4700  035434  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4701  035436  063220                  01400   063220                  ;SP(0)_BR
4702  035440  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4703  035442  060400                  01600   060400                  ;BR_SP(0)+BR
4704  035444  000207                  01700   RTS     PC
4705                                  01800
4706                                  01900
4707  035446                          02000   SETBR0:
4708                                  02100           ;THIS SUBROUTINE SETS BR0 BIT
4709                                  02200
4710  035446  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4711  035450  000401                  02400   000401                  ;BR_001
4712  035452  000207                  02500   RTS     PC
4713                                  02600
4714                                  02700
4715  035454                          02800   SETBR1:
4716                                  02900           ;THIS SUBROUTINE SETS BR1 BIT
4717                                  03000
```



```
4718  035454  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4719  035456  000402                  03200   000402                  ;BR_002
4720  035460  000207                  03300   RTS     PC
4721                                  03400
4722                                  03500
4723  035462                          03600   SETBR4:
4724                                  03700           ;THIS SUBROUTINE SETS BR4 BIT
4725                                  03800
4726  035462  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4727  035464  000420                  04000   000420                  ;BR_020
4728  035466  000207                  04100   RTS     PC
4729                                  04200
4730                                  04300
4731  035470                          04400   SETBR7:
4732                                  04500           ;THIS SUBROUTINE SETS BR7 BIT
4733                                  04600
4734  035470  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4735  035472  000600                  04800   000600                  ;BR_200
4736  035474  000207                  04900   RTS     PC
4737                                  05000
4738                                  05100
4739  035476                          05200   SETC:
4740                                  05300           ;THIS SUBROUTINE SETS THE C BIT
4741                                  05400
4742  035476  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4743  035500  000777                  05600   000777                  ;BR_377
4744  035502  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4745  035504  063220                  05800   063220                  ;SP(0)_BR
4746  035506  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4747  035510  060400                  06000   060400                  ;BR_SP(0)+BR
4748  035512  000207                  06100   RTS     PC
4749                                  06200
4750                                  06300
4751  035514                          06400   SETZ:
4752                                  06500           ;THIS SUBROUTINE SETS THE Z BIT
4753                                  06600
4754  035514  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4755  035516  000777                  06800   000777                  ;BR_377
4756  035520  000207                  06900   RTS     PC
4757                                  07000
4758                                  07100
4759  035522                          07200   ROMDAT:
4760                                  07300           ;THIS SUBROUTINE LOADS R5 WITH EXPECTED ROM CONTENTS
4761                                  07400           ;AND LOADS R4 WITH ACTUAL ROM CONTENTS
4762                                  07500
4763  035522  017500  000000          07600   MOV     @(SP),R0        ;INDEX FOR COMPARE
4764  035526  062716  000002          07700   ADD     #2,(SP)         ;ADJUST STACK
4765  035532  012711  002000          07800   MOV     #BIT10,(R1)     ;SET ROMO
4766  035536  016005  011766          07900   MOV     ROMMAP(R0),R5   ;PUT "EXPECTED" IN R5
4767  035542  016104  000006          08000   MOV     6(R1),R4        ;PUT "FOUND" IN R4
4768  035546  000207                  08100   RTS     PC              ;RETURN
4769                                  08200
4770  035550                          08300   FAMDAT:
4771                                  08400           ;THIS SUBROUTINE LOADS R4 WITH THE LOWEST
4772                                  08500           ;8 BITS OF THE CRAM PC.
                                      08600
```

```
4774  035550  017605  000000          08700          MOV     @(SP),R5        ;GOOD DATA
4775  035554  062716  000002          08800          ADD     #2,(SP)         ;ADJUST STACK
4776  035560  005011                  08900          CLR     (R1)            ;CLEAR BIT10
4777  035562  052711  000400          09000          BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION IN CRAM THAT WAS
4778                                  09100                                  ;JUMPED TO, IT LOADS BR WITH ROM PC
4779  035566  005011                  09200          CLR     (R1)            ;CLR BIT8
4780  035570  104414                                 ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4781  035572  061225                  09400          061225                  ;MOV BR TO PORT 5
4782  035574  116104  000005          09500          MOVB    5(R1),R4        ;PUT "FOUND" IN R4
4783  035600  000207                  09600          RTS     PC              ;RETURN
4784                                  09700
4785  035602                          09800  WROM:
4786                                  09900          ;THIS SUBROUTINE WRITES THE ROMMAP INTO THE CRAM
4787                                  10000
4788  035602  032737  100000  001366  10100          BIT     #BIT15,STAT1    ;BE SURE DMC HAS CRAM
4789  035610  001420                  10200          BEQ     2$              ;SKIP IF NO CRAM
4790  035612  005000                  10300          CLR     R0              ;R0=CRAM ADDRESS
4791  035614  012702  011766          10400          MOV     #ROMMAP,R2      ;R2 POINTS TO ROMMAP
4792  035620  012711  002000          10500  1$:     MOV     #BIT10,(R1)     ;SET ROMO
4793  035624  010061  000004          10600          MOV     R0,4(R1)        ;LOAD CRAM ADDRESS
4794  035630  012261  000006          10700          MOV     (R2)+,6(R1)     ;LOAD WORD TO BE WRITTEN
4795  035634  052711  020000          10800          BIS     #BIT13,(R1)     ;WRITE IT!
4796  035640  005200                  10900          INC     R0              ;NEXT ADDRESS
4797  035642  022700  002000          11000          CMP     #2000,R0        ;DONE YET?
4798  035646  001364                  11100          BNE     1$              ;BR IF NO
4799  035650  005011                  11200          CLR     (R1)            ;CLEAR SEL0
4800  035652  000207                  11300  2$:     RTS     PC              ;RETURN
4801                                  11400
4802                                  11500
4803  035654                          11600  MEMSET:
4804                                  11700          ;THIS SUBROUTINE LOADS CRAM WITH SPECIAL INSTRUCTIONS
4805                                  11800          ;FOR THE CRAM JUMP TEST, ALL CRAM LOCATIONS ARE LOADED
4806                                  11900          ;WITH INSTRUCTIONS THAT MOVE A 37 TO THE BR, EXCEPT THE
4807                                  12000          ;FOLLOWING CRAM ADDRESSES: 0,1,4,7,525,1777, THESE LOCATIONS
4808                                  12100          ;CONTAIN INTRUCTIONS WHICH LOAD THE BR WITH THE LOWEST
4809                                  12200          ;8 BITS OF THAT CRAM ADDRESS.
4810                                  12300
4811  035654  005000                  12400          CLR     R0              ;R0 = CRAM ADDRESS
4812  035656  012711  002000          12500  1$:     MOV     #BIT10,(R1)     ;SET ROMO
4813  035662  010061  000004          12600          MOV     R0,4(R1)        ;LOAD CRAM ADDRESS
4814  035666  012761  000437  000006  12700          MOV     #437,6(R1)      ;LOAD INSTRUCTION
4815  035674  052711  020000          12800          BIS     #BIT13,(R1)     ;WRITE INSTRUCTION IN CRAM
4816  035700  005200                  12900          INC     R0              ;NEXT ADDRESS
4817  035702  022700  002000          13000          CMP     #2000,R0        ;DONE YET?
4818  035706  001363                  13100          BNE     1$              ;BR IF NO
4819  035710  005000                  13200          CLR     R0              ;INDEX REGISTER
4820  035712  012711  002000          13300  2$:     MOV     #BIT10,(R1)     ;SET ROMO
4821  035716  016061  035752  000004  13400          MOV     CRAMA(R0),4(R1) ;LOAD CRAM ADDRESS IN SEL4
4822  035724  016061  035766  000006  13500          MOV     INSTU(R0),6(R1) ;LOAD INSTRUCTIIN TO BE WRITTEN
4823  035732  052711  020000          13600          BIS     #BIT13,(R1)     ;WRITE CRAM!
4824  035736  005720                  13700          TST     (R0)+           ;NEXT
4825  035740  022700  000014          13800          CMP     #14,R0          ;DONE YET?
4826  035744  001362                  13900          BNE     2$              ;BR IF NO
4827  035746  005011                  14000          CLR     (R1)            ;CLEAR ALL BITS
4828  035750  000207                  14100          RTS     PC              ;RETURN
4829                                  14200
```



```
4830  035752  000000  000001  000004  14300  CRAMA:  .WORD   0,1,4,7,1777,525
4831  035760  000007  001777  000525
4832  035766  000400                  14400  INSTU:  000400          ;BR_0
4833  035770  000401                  14500          000401          ;BR_1
4834  035772  000404                  14600          000404          ;BR_4
4835  035774  000407                  14700          000407          ;BR_7
4836  035776  000777                  14800          000777          ;BR_377
4837  036000  000525                  14900          000525          ;BR_125
4838                                  15000
4839                                  15100
4840  036002                          15200  BASELD:
4841                                  15300          ;THIS SUBROUTINE LOADS THE DMC WITH A BASE ADDRESS
4842                                  15400          ;AND PUTS DMC INTO FULL-DUPLEX MODE
4843                                  15500
4844  036002  012711  040000          15600          MOV     #BIT14,(R1)     ;MASTER CLEAR
4845  036006  032737  100000  001366  15700          BIT     #BIT15,STAT1    ;CRAM?
4846  036014  001402                  15800          BEQ     .+6             ;BR IF NO
4847  036016  012711  100000          15900          MOV     #BIT15,(R1)     ;IF CRAM SET RUN
4848  036022  105227  000000          16000          INCB    #0              ;DELAY
4849  036026  001375                  16100          BNE     .-4             ;BR IF NOT DONE DELAY
4850  036030  005711                  16200  1$:     TST     (R1)            ;IS RUN SET?
4851  036032  100376                  16300          BPL     1$              ;BR IF NO
4852  036034  052711  004000          16400          BIS     #BIT11,(R1)     ;SET LU LOOP
4853  036040  152711  000043          16500          BISB    #43,(R1)        ;BASE REQUEST
4854  036044  105711                  16600  2$:     TSTB    (R1)            ;RDY I SET?
4855  036046  100376                  16700          BPL     2$              ;BR IF NO
4856  036050  012761  035030  000004  16800          MOV     #BASE,4(R1)     ;LOAD BASE ADDRESS
4857  036056  005061  000006          16900          CLR     6(R1)           ;CLEAR CC
4858  036062  142711  000040          17000          BICB    #40,(R1)        ;CLEAR RQI
4859  036066  105711                  17100  3$:     TSTB    (R1)            ;RDY I CLEAR?
4860  036070  100776                  17200          BMI     3$              ;BR IF NO
4861  036072  152711  000041                         BISB    #41,(R1)        ;ASK FOR CNTL I
4862  036076  105711                          64$:    TSTB    (R1)            ;WAIT FOR RDI
4863  036100  100376                          BPL     64$             ;BR IF NOT SET Y
4864  036102  005061  000006                  CLR     6(R1)           ;SET FULL DUPLEX
4865  036106  142711  000040                  BICB    #40,(P1)        ;CLEAR RQI
4866  036112  105711                  65$:    TSTB    (R1)            ;RDI UP?
4867  036114  100776                  BMI     65$             ;BR IF YES
4868  036116  000207                  17400          RTS     PC              ;RETURN
4869                                  17500
4870  036120                          17600  BASELH:
4871                                  17700          ;THIS SUBROUTINE LOADS THE DMC WITH A BASE ADDRESS
4872                                  17800          ;AND PUTS DMC INTO HALF-DUPLEX MODE
4873                                  17900
4874  036120  012711  040000          18000          MOV     #BIT14,(R1)     ;MASTER CLEAR
4875  036124  032737  100000  001366  18100          BIT     #BIT15,STAT1    ;CRAM?
4876  036132  001402                  18200          BEQ     .+6             ;BR IF NO
4877  036134  012711  100000          18300          MOV     #BIT15,(R1)     ;IF CRAM SET RUN
4878  036140  105227  000000          18400          INCB    #0              ;DELAY
4879  036144  001375                  18500          BNE     .-4             ;BR IF NOT DONE DELAY
4880  036146  005711                  18600  1$:     TST     (R1)            ;IS RUN SET?
4881  036150  100376                  18700          BPL     1$              ;BR IF NO
4882  036152  052711  004000          18800          BIS     #BIT11,(R1)     ;SET LU LOOP
4883  036156  152711  000043          18900          BISB    #43,(R1)        ;BASE REQUEST
4884  036162  105711                  19000  2$:     TSTB    (R1)            ;RDY I SET?
4885  036164  100376                  19100          BPL     2$              ;BR IF NO
```

```
 4886  036166  012761  035030  000004  19200          MOV     #BASE,4(R1)     ;LOAD BASE ADDRESS
 4887  036174  005061  000006          19300          CLR     6(R1)           ;CLEAR CC
 4888  036200  142711  000040          19400          BICB    #40,(R1)        ;CLEAR RQI
 4889  036204  105711                  19500  3$:     TSTB    (R1)            ;RDY I CLEAR?
 4890  036206  100776                  19600          BMI     3$              ;BR IF NO
 4891  036210  152711  000041                         BISB    #41,(R1)        ;ASK FOR CNTL I
 4892  036214  105711                          64$:    TSTB    (R1)            ;WAIT FOR RDI
 4893  036216  100376                          BPL     64$             ;BR IF NOT SET?
 4894  036220  012761  002000  000006                 MOV     #BIT10,6(R1)    ;SET HALF DUPLEX
 4895  036226  142711  000040                         BICB    #40,(R1)        ;CLEAR RQI
 4896  036232  105711                          65$:    TSTB    (R1)            ;RDI UP?
 4897  036234  100776                          BMI     65$             ;BR IF YES
 4898  036236  000207                  19800          RTS     PC              ;RETURN
 4899                                  19900
 4900  036240                          20000  RFRELD:
 4901                                  20100          ;THIS SUBROUTINE LOADS THE DMC WITH A RECEIVE BA/CC
 4902                                  20200
 4903  036240  152711  000044          20300          BISB    #44,(R1)        ;REC BA/CC REQUEST
 4904  036244  105711                  20400  1$:     TSTB    (R1)            ;RDY I SET?
 4905  036246  100376                  20500          BPL     1$              ;BR IF NO
 4906  036250  012561  000004          20600          MOV     (R5)+,4(R1)     ;LOAD REC BA
 4907  036254  012561  000006          20700          MOV     (R5)+,6(R1)     ;LOAD REC CC
 4908  036260  142711  000040          20800          BICB    #40,(R1)        ;CLEAR RQI
 4909  036264  105711                  20900  2$:     TSTB    (R1)            ;IS RDY I CLEAR
 4910  036266  100776                  21000          BMI     2$              ;BR IF NO
 4911  036270  000205                  21100          RTS     R5              ;RETURN
 4912                                  21200
 4913  036272                          21300  XFRELD:
 4914                                  21400          ;THIS SUBROUTINE LOADS THE DMC WITH A TRANSMIT BA/CC
 4915                                  21500
 4916  036272  152711  000040          21600          BISB    #40,(R1)        ;XMIT BA/CC REQUEST
 4917  036276  105711                  21700  1$:     TSTB    (R1)            ;RDY I SET?
 4918  036300  100376                  21800          BPL     1$              ;BR IF NO
 4919  036302  012561  000004          21900          MOV     (R5)+,4(R1)     ;LOAD XMIT BA
 4920  036306  012561  000006          22000          MOV     (R5)+,6(R1)     ;LOAD XMIT CC
 4921  036312  142711  000040          22100          BICB    #40,(R1)        ;CLEAR RQI
 4922  036316  105711                  22200  2$:     TSTB    (R1)            ;IS RDY I CLEAR
 4923  036320  100776                  22300          BMI     2$              ;BR IF NO
 4924  036322  000205                  22400          RTS     R5              ;RETURN
 4925                                  00300
       036324  041777  040522  020115  00400  EM1:    .ASCIZ  <377>/CRAM DATA ERROR/
       036345     377  051103  046501  00500  EM2:    .ASCIZ  <377>/CRAM DUAL ADDRESSING ERROR/
       036401     377  051103  046517  00600  EM3:    .ASCIZ  <377>/CROM DATA ERROR/
       036422  045377  046525  020120  00700  EM4:    .ASCIZ  <377>/JUMP ERROR/
       036436  047777  052104  042440  00800  EM5:    .ASCIZ  <377>/ODT ERROR IN IBUS* REG10/
       036470  044777  050117  046440  00900  EM6:    .ASCIZ  <377>/IOP MAIN MEMORY TEST/
       036516  044777  050117  046440  01000  EM7:    .ASCIZ  <377>/IOP MAR TEST/
       036534  041377  020122  044522  01100  EM10:   .ASCIZ  <377>/BR RIGHT SHIFT TEST/
       036561     377  042522  042503  01200  EM11:   .ASCIZ  <377>/RECEIVE DATA ERROR/
       036605     377  051106  042505  01300  EM12:   .ASCIZ  <377>/FREE RUNNING ERROR/
       036631     377  047503  052116  01400  EM13:   .ASCIZ  <377>/CONTROL OUT ERROR/
       036654  044777  052116  051105  01500  EM14:   .ASCIZ  <377>/INTERNAL DDCMP ERROR COUNTS NON ZERO/
                                       01600
       036722  042777  050130  041505  01700  DH1:    .ASCIZ  <377>/EXPECTED  FOUND  ADDRESS/
       036754  042777  050130  041505  01800  DH2:    .ASCIZ  <377>/EXPECTED  FOUND/
       036775     377  051440  046105  01900  DH3:    .ASCIZ  <377>/ SEL4      SEL6/
```



```
       037016  041377  051501  025505  02000  DH4:    .ASCIZ  <377>/BASE+3 THRU BASE+12 /
                                       02100          .EVEN
                                       02200
       037044  000003                  02300  DT1:    3
       037046     006      004         02400          .BYTE   6,4
       037050  001764                  02500          SAVR2
       037052     006      004         02600          .BYTE   6,4
       037054  001270                  02700          SAVR4
       037056     004      002         02800          .BYTE   4,2
       037060  001260                  02900          SAVR0
       037062  000003                  03000  DT2:    3
       037064     006      004         03100          .BYTE   6,4
       037066  001272                  03200          SAVR5
       037070     006      004         03300          .BYTE   6,4
       037072  001270                  03400          SAVR4
       037074     004      002         03500          .BYTE   4,2
       037076  001264                  03600          SAVR2
       037100  000003                  03700  DT3:    3
       037102     006      004         03800          .BYTE   6,4
       037104  001272                  03900          SAVR5
       037106     006      004         04000          .BYTE   6,4
       037110  001270                  04100          SAVR4
       037112     004      002         04200          .BYTE   4,2
       037114  001252                  04300          TEMP3
       037116  000002                  04400  DT4:    2
       037120     003      007         04500          .BYTE   3,7
       037122  001272                  04600          SAVR5
       037124     003      002         04700          .BYTE   3,2
       037126  001270                  04800          SAVR4
       037130  000002                  04900  DT5:    2
       037132     006      004         05000          .BYTE   6,4
       037134  001272                  05100          SAVR5
       037136     006      002         05200          .BYTE   6,2
       037140  001270                  05300          SAVR4
       037142  000003                  05400  DT6:    3
       037144     003      010         05500          .BYTE   3,10
       037146  001272                  05600          SAVR5
       037150     003      004         05700          .BYTE   3,4
       037152  001270                  05800          SAVR4
       037154     004      002         05900          .BYTE   4,2
       037156  034704                  06000          FLAG
       037160  000003                  06100  DT7:    3
       037162     003      010         06200          .BYTE   3,10
       037164  001272                  06300          SAVR5
       037166     003      004         06400          .BYTE   3,4
       037170  001270                  06500          SAVR4
       037172     004      002         06600          .BYTE   4,2
       037174  001264                  06700          SAVR2
       037176  000003                  06800  DT10:   3
       037200     003      007         06900          .BYTE   3,7
       037202  001272                  07000          SAVR5
       037204     003      004         07100          .BYTE   3,4
       037206  001270                  07200          SAVR4
       037210     006      002         07300          .BYTE   6,2
       037212  001252                  07400          TEMP3
                                       07500  DT11:   2
```

```
037216      006     004          07600          .BYTE   6,4
037220  001252                   07700          TEMP3
037222      006     002          07800          .BYTE   6,2
037224  001254                   07900          TEMP4
037226  000010                   08000  DT12:   10
037230      003     002          08100          .BYTE   3,2
037232  001250                   08200          TEMP2
037234      003     002          08300          .BYTE   3,2
037236  035034                   08400          BASE+4
037240      003     002          08500          .BYTE   3,2
037242  001252                   08600          TEMP3
037244      003     002          08700          .BYTE   3,2
037246  035036                   08800          BASE+6
037250      003     002          08900          .BYTE   3,2
037252  001254                   09000          TEMP4
037254      003     002          09100          .BYTE   3,2
037256  035040                   09200          BASE+10
037260      003     002          09300          .BYTE   3,2
037262  001256                   09400          TEMP5
037264      003     002          09500          .BYTE   3,2
037266  035042                   09600          BASE+12
                                 09700
037270                           09800  .ERRTAB:
037270  000000                   09900          0
037272  000000                   10000          0
037274  000000                   10100          0
037276  036324                   10200  EM1
037300  036722                   10300  DH1     ;HLT    1
037302  037044                   10400  DT1
037304  036345                   10500  EM2
037306  036722                   10600  DH1     ;HLT    2
037310  037044                   10700  DT1
037312  036324                   10800  EM1
037314  036722                   10900  DH1     ;HLT    3
037316  037062                   11000  DT2
037320  036401                   11100  EM3
037322  036722                   11200  DH1     ;HLT    4
037324  037100                   11300  DT3
037326  036422                   11400  EM4
037330  036754                   11500  DH2     ;HLT    5
037332  037116                   11600  DT4
037334  036422                   11700  EM4
037336  036754                   11800  DH2     ;HLT    6
037340  037130                   11900  DT5
037342  036436                   12000  EM5
037344  036754                   12100  DH2     ;HLT    7
037346  037116                   12200  DT4
037350  036470                   12300  EM6
037352  036722                   12400  DH1     ;HLT    10
037354  037142                   12500  DT6
037356  036516                   12600  EM7
037360  036722                   12700  DH1     ;HLT    11
037362  037160                   12800  DT7
037364  036534                   12900  EM10
037366  036754                   13000  DH2     ;HLT    12
037370  037116                   13100  DT4
```



```
037372  036561                   13200  EM11
037374  036722                   13300  DH1     ;HLT    13
037376  037176                   13400  DT10
037400  036605                   13500  EM12
037402  000000                   13600  0       ;HLT    14
037404  000000                   13700  0
037406  036605                   13800  EM12
037410  036754                   13900  DH2     ;HLT    15
037412  037130                   14000  DT5
037414  036631                   14100  EM13
037416  036775                   14200  DH3     ;HLT    16
037420  037214                   14300  DT11
037422  036654                   14400  EM14
037424  037016                   14500  DH4     ;HLT    17
037426  037226                   14600  DT12
                                 14700
                                 14800
037430                           14900  CORMAX:
        000001                   15400  .END
```

```
ADRCNT= 004301   858*   894*   903#
AUDONE  002734   566    639#
AUSTRT  002446   565#   596    643
AUTO.S  010252   528    1343#
BASE    035030  3841    4441   4446   4452   4455   4458   4459   4460   4461   4484   4500   4689#  4856
                4886    4925
BASELF  036002  3980    4036   4085   4131   4180   4224   4840#
BASELH  036120  4268    4870#
BINWRD  004422   944*   947*   948    985#
BIT0  = 000001    95#   1133   1134   3804   3914   3929   3974   3995   4030   4053   4079   4099   4125
                4148    4174   4192   4218   4236   4262   4305   4413   4545
BIT1  = 000002    94#    533   1127   1133   1134   1503   4482
BIT10 = 002000    85#   1475   1486   1682   1718   1754   1770   1804   2108   4765   4792   4812   4820
                4894
BIT11 = 004000    84#   3833   4852   4882
BIT12 = 010000    83#   1427   1501   3807   3977   4033   4082   4128   4177   4221   4265   4308   4485
BIT13 = 020000    82#   1430   1478   1505   1685   1721   1757   4795   4815   4823
BIT14 = 040000    81#    761   1441   1443   1505   1516   3818   3831   4340   4557   4844   4874
BIT15 = 100000    80#    487    569    572   1414   1481   1677   1712   1750   1797   1832   1878   1927
                1994   2036   2097   2146   2202   2255   2311   2367   2423   2479   2535   2592   2649
                2706   2763   2820   2877   2938   3000   3059   3121   3183   3245   3307   3369   3431
                3493   3555   3617   3679   3741   3802   3819   3821   3972   4028   4077   4123   4172
                4216   4260   4303   4341   4343   4558   4560   4788   4845   4847   4875   4877
BIT2  = 000004    93#    533    692   1134   3908   3999   4574
BIT3  = 000010    92#   1507   1514   4506   4530
BIT4  = 000020    91#   1117   1140   1142   4057
BIT5  = 000040    90#   1618   2047
BIT6  = 000100    89#   1122   1123   1509   2062   4360   4566
BIT7  = 000200    88#   1123   1240   1618
BIT8  = 000400    87#   1485   1496   1498   1511   1513   1519   1522   1579   2045   4103   4152   4777
BIT9  = 001000    86#   1483   1485   1493   1519   1522   1577   1579   4196   4240   4550   4554
BM      006756  1165#   1455
BRLVL   011720  1574    1584   1592   1604#
BRW     003640   698    784#
BRX     003642   699    785#
CHRCNT  004620   942*   945    949    965*   983#   984
CKSWR   007362   514    759    790   1005   1190#
CKSWR1  007426  1200#   1212
CKSWR2  007440  1203#
CKSWR3  007444  1205#
CKSWR4  007450  1206#   1214   1221
CKSWR5  007554  1191   1195   1230#
CLKX    001242   171#
CLRALL  035430  2149   2162   2175   2595   2608   2621   2652   2665   2678   2709   2722   2735   2766
                2779   2792   2823   2836   2849   2880   2893   2906   2942   2955   2968   3435   3448
                3461   3497   3510   3523   3559   3572   3585   3621   3634   3647   3683   3696   3709
                3745   3758   3771   4695#
CLRTAB  031676  4345#  4411   4419   4436
CNERR   007167   607   1165#
CNT.AB  001702   196    365#   486    488    490   1268
CNVRT = 104411   233#   604    718    720    722    724   1039   1041   1100   1203
CONERR  007125   602   1165#
CONN    007016  1165#  1432
CONTAB  002710   611    624#
CONVRT= 104410   231#   545    610   1055   1366
CORMAX  037430  4925#
```



```
CRAM    006510  1165#  1405
CRAMA   035752  4821   4830#
CREAM   001320   195#   485*   1264*  1265   1267*  1271
CSR     006412  1165#  1370
CSRMAP  010254  1345#
CYCLE   007620   700    741    742   1255#
DATABP  005124  1028*  1031   1053   1056#
DATACL= 104415   241#
DATAHD  005112  1027*  1049   1052#
DELAY = 104413   237#
DEVADR  004276   856*   891    901#
DEVTAB  002722   580    629#
DH1     036722  4925#
DH2     036754  4925#
DH3     036775  4925#
DH4     037016  4925#
DISPLA  001200   142#   500*   506*   715*
DISPMP  000174   128#   506
DMACTV  001306   189#   523    656*   657   1255   1269   1351*  1534*  1540*  1541*  1545   1570
DMCM    007210   615   1165#
DMCR00  001500   280#
DMCR01  001510   285#
DMCR02  001520   290#
DMCR03  001530   295#
DMCR04  001540   300#
DMCR05  001550   305#
DMCR06  001560   310#
DMCR07  001570   315#
DMCR10  001600   320#
DMCR11  001610   325#
DMCR12  001620   330#
DMCR13  001630   335#
DMCR14  001640   340#
DMCR15  001650   345#
DMCR16  001660   350#
DMCR17  001670   355#
DMCSP   001404   263#   565*   591    628    745    782   1095   1141   1273*  1282   1331   1638
DMCSRH  001406   264#  1122*  1123*  1127*  1133*  1134*  1140*  1142*  1292*  1283*  1284
DMCTL   001410   265#  1284*  1285*  1286
DMNUM   001310   190#   481    731   1349*  1364*  1526*  1527   1535   1537
DMP04   001412   266#  1111*  1117   1154   1159   1286*  1287*  1288
DMP06   001414   267#  1128*  1288*  1289*
DMRLVL  001376   260#  1291*  1292*  1293   4319#
DMRVEC  001374   259#   748   1274*  1275*  1291   4318#
DMS100  001502   281#
DMS101  001512   286#
DMS102  001522   291#
DMS103  001532   296#
DMS104  001542   301#
DMS105  001552   306#
DMS106  001562   311#
DMS107  001572   316#
DMS110  001602   321#
DMS111  001612   326#
DMS112  001622   331#
```

```
DMS114  001642          341#
DMS115  001652          346#
DMS116  001662          351#
DMS117  001672          356#
DMS200  001504          282#
DMS201  001514          287#
DMS202  001524          292#
DMS203  001534          297#
DMS204  001544          302#
DMS205  001554          307#
DMS206  001564          312#
DMS207  001574          317#
DMS210  001604          322#
DMS211  001614          327#
DMS212  001624          332#
DMS213  001634          337#
DMS214  001644          342#
DMS215  001654          347#
DMS216  001664          352#
DMS217  001674          357#
DMS300  001506          283#
DMS301  001516          288#
DMS302  001526          293#
DMS303  001536          298#
DMS304  001546          303#
DMS305  001556          308#
DMS306  001566          313#
DMS307  001576          318#
DMS310  001606          323#
DMS311  001616          328#
DMS312  001626          333#
DMS313  001636          338#
DMS314  001646          343#
DMS315  001656          348#
DMS316  001666          353#
DMS317  001676          358#
DMTLVL  001402          262#   1295*   1296*   4321*
DMTVEC  001400          261#   1293*   1294*   1295    4320*
DM.END  001700          360#   1347
DM.MAP  001500          195     279*    485     538     548     642    1265    1267    1345    1350    1562
DT1     037044         4925#
DT10    037176         4925#
DT11    037214         4925#
DT12    037226         4925#
DT2     037062         4925#
DT3     037100         4925#
DT4     037116         4925#
DT5     037130         4925#
DT6     037142         4925#
DT7     037160         4925#
EM1     036324         4925#
EM10    036534         4925#
EM11    036561         4925#
EM12    036605         4925#
EM13    036631         4925#
EM14    036654         4925#
```



```
EM2     036345         4925#
EM3     036401         4925#
EM4     036422         4925#
EM5     036436         4925#
EM6     036470         4925#
EM7     036516         4925#
ENDEX   032334         4412    4437#
ENDEX1  032434         4307    4457#   4463    4474    4573
ENDEX2  032436         4442    4447    4453    4456    4458#
ENDEX3  032472         4371    4464#
ERCT00  001704          367#
ERCT01  001710          370#
ERCT02  001714          373#
ERCT03  001720          376#
ERCT04  001724          379#
ERCT05  001730          382#
ERCT06  001734          385#
ERCT07  001740          388#
ERCT10  001744          391#
ERCT11  001750          394#
ERCT12  001754          397#
ERCT13  001760          400#
ERCT14  001764          403#
ERCT15  001770          406#
ERCT16  001774          409#
ERCT17  002000          412#
ERR     002612          589     599#    603
ERRCNT  001232          163#    727     754    1066#   1280#
ERRFLG  001325          202#    483#    713#    775#   1016#   1029    1043#   1101#
ERRMSG  005100         1026#   1044    1047#
ERRPC   002702          605     621#
ERTAB0  005224         1041    1075#
EXIT  = 000205           96#
EXITER  005160         1061    1066#
FLAG    034704         1834#   1838    1839    1860#   1861    1880#   1885    1886    1908#   1909    1929#   1930    1933
                       1934    1952#   1953    1956#   1957    1960    1961    1976#   1977    4339#   4350    4361    4369
                       4386    4408    4410#   4413    4418#   4435#   4438    4536    4547    4549#   4675#   4925
FLOAT   002536          582#    588
FY      002562          590#    594
HALTS   005130         1012    1058#
HILIM   004274          855#    882     900#
ICOUNT  001222          159#    773     778#
IISP    032534         4318    4478#
INBUF   007256          825     861    1180#
INCHAR  007560         1206    1234#
INTFLG  001324          201#    509     529     536#
INSTAB= 104404          223#    876
INSTR = 104403          221#   1304    1356    1369    1378    1445    1454
INSTR2  004074          832     844#
INSTU   035766         4822    4832#
INTTY   011734         1389    1406    1417    1433    1612#
KMCM    007220          618    1165#
LIMITS  004222          871     882#
LINE    006720         1165#   1446
LOBITS  004300          857#    886     902#    903
LOCK    001220          158#    777#    793     795    1035    1675#   1709#   1747#   1768#   1794#   1829#   1875#   1924#
```

```
                    1955*  2033*  2059*  2094*  2143*  2160*  2173*  2199*  2215*  2227*  2252*  2269*  2282*
                    2308*  2325*  2338*  2364*  2381*  2394*  2420*  2437*  2450*  2476*  2493*  2506*  2532*
                    2549*  2562*  2589*  2606*  2619*  2646*  2663*  2676*  2703*  2720*  2733*  2760*  2777*
                    2790*  2817*  2834*  2847*  2874*  2891*  2904*  2935*  2953*  2966*  2997*  3014*  3026*
                    3056*  3074*  3087*  3118*  3136*  3149*  3180*  3198*  3211*  3242*  3260*  3273*  3304*
                    3322*  3335*  3366*  3384*  3397*  3428*  3446*  3459*  3490*  3508*  3521*  3552*  3570*
                    3583*  3614*  3632*  3645*  3676*  3694*  3707*  3738*  3756*  3769*
LOKFLG  001326       203#
LOLIM   004272       854*   884    899#
LPCNT   001224       160#   772*   773    776*
LSTERR  001234       164#   492*   712*   1013   1015*  1102*
MASKX   001244       172#
MASTER  006044      1037   1165#
MCRLF   005574       810    933    1033   1034   1042   1165#  1303   1365
MCSPX   005774       717   1165#
MDATA   007320       963    973    1182#
MEMLIM  001304       188#   680*
MEMSET  035654      2940   3002   3061   3123   3185   3247   3309   3371   3433   3495   3557   3619   3681
                    3743   4803#
MEPASS  005635       716   1165#
MERRPC  006121      1040   1165#
MERRX   006021       723   1165#
MERR2   005662      1165#  1547
MERR3   005707       653   1165#
MILK    001322       196#   486*   725    1263*  1268*  1272
MLOCK   005745       694   1165#
MNEW    006046       648   1165#
MODU    006606      1165#  1416
MPASSX  006010       721   1165#
MPFAIL  005577      1099   1165#
MOM     005570       840   1165#  1327   1402   1412   1425   1439
MR      005657       703   1165#  1321
MRESET= 004000        96#
MSTCLP= 104412       235#  1104   1637   1711   1749   1796   1831   1877   1926   1993   2035   2096   2145
                    2201   2254   2310   2366   2422   2478   2534   2591   2648   2705   2762   2819   2876
                    2937   2999   3058   3120   3182   3244   3306   3368   3430   3492   3554   3616   3678
                    3740   3801   3971   4027   4076   4122   4171   4215   4259   4302
MTITLE  001000       136#   513
MTSTN   006032      1038   1165#  1305
MTSTPC  005733      1165#
MVECX   006002       719   1165#
NEXT    001216       157#   779    1071   1635*  1674*  1708*  1746*  1793*  1828*  1874*  1923*  1991*  2032*
                    2093*  2142*  2198*  2251*  2307*  2363*  2419*  2475*  2531*  2588*  2645*  2702*  2759*
                    2816*  2873*  2934*  2996*  3055*  3117*  3179*  3241*  3303*  3365*  3427*  3489*  3551*
                    3613*  3675*  3737*  3799*  3969*  4025*  4074*  4120*  4169*  4213*  4257*  4300*
NOACT   007056       525   1165#  1257
NODEV   002606       574    597#
NUM     006352      1165#  1357
OISR    033040      4320   4545#
OK      002600       570    573    592    595#   620
ONE     001302       187#
PACT00  001702       366#
PACT01  001706       369#
PACT02  001712       372#
PACT03  001716       375#
PACT04  001722       378#
```



```
PACT05  001726       381#
PACT06  001732       384#
PACT07  001736       387#
PACT10  001742       390#
PACT11  001746       393#
PACT12  001752       396#
PACT13  001756       399#
PACT14  001762       402#
PACT15  001766       405#
PACT16  001772       408#
PACT17  001776       411#
PARAM = 104405       225#  1306   1358   1371   1380   1447   1456
PARAM1  004142       860#   877
PARBIT= 040000        96#
PARERR  004216       863    865    867    876#   883    885    887
PASCNT  001230       162#   714*   715    726    751   1279*
PERFOR= 004537        96#
PFTAB   005332      1100   1106#
POPRO = 012600        72#  1065
POP1SP= 005726        70#
POP2SP= 022626        74#   781
PRIO    006451      1165#  1388
PS    = 177776        63#   478*   691*  1574*  1584*  4311*  4368*
PUSHRO= 010046        71#  1062
PUSH1S= 005746        69#
PUSH2S= 024646        73#
QV.FLG  001327       204#   484*   730*   770
RAMDAT  035550      2947   2960   2973   3008   3020   3032   3068   3081   3094   3130   3143   3156   3192
                    3205   3218   3254   3267   3280   3316   3329   3342   3378   3391   3404   3440   3453
                    3466   3502   3515   3528   3564   3577   3590   3626   3639   3652   3688   3701   3714
                    3750   3763   3776   4770#
RBUF    034762      3812   3871   3932   3940   4038   4270   4687#
RBUFF   033746      4327   4660#
RBUFFE  034702      4348   4668#
RBUFF1  033746      4661#
RBUFF2  034052      4662#
RBUFF3  034156      4663#
RBUFF4  034262      4664#
RBUFF5  034366      4665#
RBUFF6  034472      4666#
RBUFF7  034576      4667#
RCNTAB  033620      4335   4337   4511   4518   4654#
RCOUNT  034760      3810   3872   3935   3938   4686#
RDNTAB  033656      4346   4382   4405   4611*  4613*  4657#
RECBA   033452      4426   4510   4598   4622#
RESREG  005126      1054   1057#
RESTAB  005252      1086   1092#
RESTRT  003444       729    733    741#
RESUME  033450      4345*  4480   4488*  4492   4563*  4621#
RES05 = 104407       229#  1057
RETURN  001214       156#   494*   700*   704    741*   779*   783   1071*  1073   1105   1320*  1330*  1332
RFLAG   034710      3817*  3925   3928*  3953   4677#
RFREE   036240      4037   4132   4269   4900#
ROMCLK= 104414       239#  1112   1115   1152   1157   1642   1644   1646   1654   1656   1840   1842   1845
                    1847   1849   1887   1889   1892   1894   1896   1935   1937   1940   1942   1944   1962
                    1964   1966   1968   1997   2000   2006   2009   2039   2042   2054   2060   2068   2070
```

```
                    2072    2106    2150    2152    2163    2165    2176    2178    2205    2207    2217    2219    2229
                    2231    2259    2261    2272    2274    2285    2287    2315    2317    2328    2330    2341    2343
                    2371    2373    2384    2386    2397    2399    2427    2429    2440    2442    2453    2455    2483
                    2485    2496    2498    2509    2511    2539    2541    2552    2554    2565    2567    2596    2598
                    2609    2611    2622    2624    2653    2655    2666    2668    2679    2681    2710    2712    2723
                    2725    2736    2738    2767    2769    2780    2782    2793    2795    2824    2826    2837    2839
                    2850    2852    2881    2883    2894    2896    2907    2909    2943    2945    2956    2958    2969
                    2971    3004    3006    3016    3018    3028    3030    3064    3066    3077    3079    3090    3092
                    3126    3128    3139    3141    3152    3154    3188    3190    3201    3203    3214    3216    3250
                    3252    3263    3265    3276    3278    3312    3314    3325    3327    3338    3340    3374    3376
                    3387    3389    3400    3402    3436    3438    3449    3451    3462    3464    3498    3500    3511
                    3513    3524    3526    3560    3562    3573    3575    3586    3588    3622    3624    3635    3637
                    3648    3650    3684    3686    3697    3699    3710    3712    3746    3748    3759    3761    3772
                    3774    4698    4700    4702    4710    4718    4726    4734    4742    4744    4746    4754    4780
ROMDAT  035522      2154    2167    2180    2209    2221    2233    2263    2276    2289    2319    2332    2345    2375
                    2388    2401    2431    2444    2457    2487    2500    2513    2543    2556    2569    2600    2613
                    2626    2657    2670    2683    2714    2727    2740    2771    2784    2797    2828    2841    2854
                    2885    2898    2911    4759#
ROMMAP  011766      1622#   1801    2100    4766    4791
RUN     001316       193#    487*   1261*   1262*   1269
SAVACT  001312       191#    651    1545*
SAVNUM  001314       192#    481*    728*    731*   1538*
SAVPC   001276       185#    603*    623     908*   1077
SAVR0   001260       178#    917*    922    4925
SAVR1   001262       179#    609*    626     916*    923
SAVR2   001264       180#    915*    924    4925
SAVR3   001266       181#    914*    925
SAVR4   001270       182#    913*    926    4925
SAVR5   001272       183#    912*    927    4925
SAVSP   001274       184#
SAVO5 = 104406       227#   1017
SCAN    032020      4369#   4469    4472
SCAN1   032504      4374    4377    4380    4465   4467#
SCAN2   032046      4373    4375#   4466
SCOPE = 104400       215#   1663    1697    1734    1780    1817    1863    1911    1979    2019    2080    2129    2186
                    2239    2295    2351    2407    2463    2519    2575    2632    2689    2746    2803    2860    2917
                    2979    3038    3100    3162    3224    3286    3348    3410    3472    3534    3596    3658    3720
                    3782    3957    4010    4062    4108    4157    4201    4245    4283    4457
SCOP1 = 104401       217#   1690    1726    1763    1776    1812    1856    1903    1951    1975    2053    2120    2159
                    2172    2185    2214    2226    2238    2268    2281    2294    2324    2337    2350    2380    2393
                    2406    2436    2449    2462    2492    2505    2518    2548    2561    2574    2605    2618    2631
                    2662    2675    2688    2719    2732    2745    2776    2789    2802    2833    2846    2859    2890
                    2903    2916    2952    2965    2978    3013    3025    3037    3073    3086    3099    3135    3148
                    3161    3197    3210    3223    3259    3272    3285    3321    3334    3347    3383    3396    3409
                    3445    3458    3471    3507    3520    3533    3569    3582    3595    3631    3644    3657    3693
                    3706    3719    3755    3768    3781
SETBP0  035446      2370    2383    2396    3187    3200    3213   4707#
SETBP1  035454      2426    2439    2452    3249    3262    3275   4715#
SETBP4  035462      2482    2495    2508    3311    3324    3337   4723#
SETBP7  035470      2538    2551    2564    3373    3386    3399   4731#
SETC    035476      2258    2271    2284    3063    3076    3089   4739#
SET7    035514      2314    2327    2340    3125    3138    3151   4751#
SOFTSW  007612      1204    1243#
SPACNT= 004621       943*    967     970*    984#
STACK = 001200        64#    479     670    1072    1094
STAT    001240       170#
```



```
STAT1   001366       252#   1276*   1677    1712    1750    1797    1832    1878    1927    1994    2036    2097    2146
                    2202    2255    2311    2367    2423    2479    2535    2592    2649    2706    2763    2820    2877
                    2938    3000    3059    3121    3183    3245    3307    3369    3431    3493    3555    3617    3679
                    3741    3802    3807    3819    3831    3972    3977    4028    4033    4077    4082    4123    4128
                    4172    4177    4216    4221    4260    4265    4303    4308    4312    4341    4558    4788    4845
                    4875
STAT2   001370       253#   1277*
STAT3   001372       254#   1278*   3804    3974    4030    4079    4125    4174    4218    4262    4305
STRTSW  001236       169#    515*    518*    519     521     531     533     646     692     701    1298   1322*   1352
                    1558
SV05    004310       912#
SWFLG   007556       482*    804    1199*   1226*   1232#
SWMES   007107      1165#   1202
SWMES1  007117      1165#   1205
SWR     001202       143#    499*    501     505*    515     651     656     761     768     791     806    1006    1011
                    1060    1067    1069    1130    1190   1225*
SWREG   000176       129#    505    1190    1245
SW00  = 000001        45#    519    1352    1558
SW01  = 000002        44#    701    1298    1322
SW02  = 000004        43#
SW03  = 000010        42#    646
SW04  = 000020        41#
SW05  = 000040        40#
SW06  = 000100        39#   1130
SW07  = 000200        38#
SW08  = 000400        37#   1067
SW09  = 001000        36#    791
SW10  = 002000        35#   1069
SW11  = 004000        34#    768
SW12  = 010000        33#    806    1006
SW13  = 020000        32#   1011
SW14  = 040000        31#
SW15  = 100000        30#
TBUF    034714      3887    3917    3939    3982    4041    4136    4273   4679#
TBUFF   033510      4332    4576    4578   4625#
TCOUNT  034712      3888    3920   4678#
TEMP    001416       272#    950    1096*   1097*   1138*   1143*   1149*   1161*   3824*   3827*   3835*   3838*   3844*
                    3847*   3865*   3868*   3874*   3877*   3881*   3884*   3890*   3893*   3896*   3900*   3986*   3989*
                    4007*   4012*   4044*   4047*   4090*   4093*   4139*   4142*   4185*   4188*   4229*   4232*   4276*
                    4279*   4366*   4467*
TEMP1   001246       173#    539*   1167    1570*   1571*   3897*   3902*
TEMP2   001250       174#    540*   1169    3905*   4367*   4458*   4470*   4925
TEMP3   001252       175#    542*    563*    599     608*   1171    1361    1364    1464*   2113*   2115*   2116*   2117*
                    2118*   3851*   3941*   4395*   4459*   4569*   4925
TEMP4   001254       176#    543*   1173    1374    1377    1383    1386    1450    1453    1459    1462   3852*   4460*
                    4570*   4925
TEMP5   001256       177#    544*   1175    1355*   1368*   1556    4461*   4925
TFLAG   034706      3816*   3910    3913*   3955    4325*   4336    4353    4415*   4417*   4676#
TIMFP = 104416       243#
TKCSR   001204       148#    764     827    1234    1612
TKDBR   001206       149#    766     829     835    1192    1194    1236    1614
TLAST = 031432      1325    4690#
TPCSR   001210       150#    811     833    1008    1237    1615
TPDBR   001212       151#    813*    835*   1010*   1239*   1617*
TRPCK   004636       996#
TSTNO   001226       161#    493*   1080    1108    1309    1316    1318    1634*   1673*   1707*   1745*   1792*   1827*
```

```
                    1873*  1922*  1990*  2031*  2092*  2141*  2197*  2250*  2306*  2362*  2418*  2474*  2530*
                    2587*  2644*  2701*  2758*  2815*  2872*  2933*  2995*  3054*  3116*  3178*  3240*  3302*
                    3364*  3426*  3488*  3550*  3612*  3674*  3736*  3798*  3968*  4024*  4073*  4119*  4168*
                    4212*  4256*  4299*
TST1     015766     1312   1330   1634#
TST10    017216     1874   1922#
TST11    017516     1923   1990#
TST12    017632     1991   2031#
TST13    020040     2032   2092#
TST14    020230     2093   2141#
TST15    020420     2142   2197#
TST16    020574     2198   2250#
TST17    020764     2251   2306#
TST2     016100     1635   1673#
TST20    021154     2307   2362#
TST21    021344     2363   2418#
TST22    021534     2419   2474#
TST23    021724     2475   2530#
TST24    022114     2531   2587#
TST25    022304     2588   2644#
TST26    022474     2645   2701#
TST27    022664     2702   2758#
TST3     016214     1674   1707#
TST30    023054     2759   2815#
TST31    023244     2816   2872#
TST32    023434     2873   2933#
TST33    023630     2934   2995#
TST34    024010     2996   3054#
TST35    024204     3055   3116#
TST36    024400     3117   3178#
TST37    024574     3179   3240#
TST4     016336     1708   1745#
TST40    024770     3241   3302#
TST41    025164     3303   3364#
TST42    025360     3365   3426#
TST43    025554     3427   3488#
TST44    025750     3489   3550#
TST45    026144     3551   3612#
TST46    026340     3613   3674#
TST47    026534     3675   3736#
TST5     016516     1746   1792#
TST50    026730     3737   3798#
TST51    027742     3799   3968#
TST52    030170     3969   4024#
TST53    030362     4025   4073#
TST54    030544     4074   4119#
TST55    030736     4120   4168#
TST56    031114     4169   4212#
TST57    031272     4213   4256#
TST6     016636     1793   1827#
TST60    031432     4257   4299#  4690
TST7     017024     1828   1873#
TTST     003522      695*   696*   698*   699*   762#
TWOSYM=  010000       96#
TYPDAT   005114     1032   1050   1053#
TYPE   = 104402      219#   513    525    537    601    606    614    617    648    653    694    703    716
```



```
                     717    719    721    723    810    823    840    933    973   1033   1034   1037   1038
                    1040   1042   1046   1051   1099   1202   1205   1257   1303   1321   1327   1365   1387
                    1401   1404   1411   1415   1424   1431   1438   1547
TYPMSG   005014     1030   1033#
VEC      006430     1165#  1379
VECMAP   011456     1546   1558#
WHICH    011450     1367   1554#
WRDCNT   004616      941*   974*   982#
WRKO,P   005102     1045   1048#
WROM     035602     1800   3809   3979   4035   4084   4130   4179   4223   4267   4310   4785#
XBX      004706     1007   1009   1011#
XCNTAB   033640     4352   4354   4421   4424   4582   4605   4655#
XCSP     003456      718    743#
XDNTAB   033712     4588*  4590*  4658#
XERR     003500      724    752#
XFRELO   036272     3981   4040   4086   4135   4272   4913#
XHEAD    006126      537   1165#
XMITBA   033452     4326   4330   4333   4517   4623#
XPASS    003472      722    749#
XSTATO   007230      546   1165#
XTSTN    005232     1039   1078#
XVEC     003464      720    746#
X0     = 000110     4626#  4629#  4632#  4635#  4638#  4641#  4644#  4647#  4650#  4653#
X1     = 000101     4626#  4629#  4632#  4635#  4638#  4641#  4644#  4647#  4650#
X2     = 000102     4626#  4629#  4632#  4635#  4638#  4641#  4644#  4647#  4650#
X3     = 000103     4626#  4629#  4632#  4635#  4638#  4641#  4644#  4647#  4650#
X4     = 000104     4626#  4629#  4632#  4635#  4638#  4641#  4644#  4647#  4650#
X5     = 000105     4626#  4629#  4632#  4635#  4638#  4641#  4644#  4647#  4650#
X6     = 000106     4626#  4629#  4632#  4635#  4638#  4641#  4644#  4647#  4650#
X7     = 000107     4626#  4629#  4632#  4635#  4638#  4641#  4644#  4647#  4650#
ZERO     001300      186#
$COD   = ****** U     1
$CRAP  = 177777       1#   1624#  1627   1630#  1664#  1667   1669#  1698#  1701   1703#  1735#  1738   1741#
                    1781#  1784   1788#  1818#  1821   1823#  1864#  1867   1869#  1912#  1915   1918#  1980#
                    1983   1986#  2020#  2023   2027#  2081#  2084   2088#  2130#  2133   2137#  2187#  2190
                    2193#  2240#  2243   2246#  2296#  2299   2302#  2352#  2355   2358#  2408#  2411   2414#
                    2464#  2467   2470#  2520#  2523   2526#  2576#  2579   2583#  2633#  2636   2640#  2690#
                    2693   2697#  2747#  2750   2754#  2804#  2807   2811#  2861#  2864   2868#  2918#  2921
                    2929#  2980#  2983   2991#  3039#  3042   3050#  3101#  3104   3112#  3163#  3166   3174#
                    3225#  3228   3236#  3287#  3290   3298#  3349#  3352   3360#  3411#  3414   3422#  3473#
                    3476   3484#  3535#  3538   3546#  3597#  3600   3608#  3659#  3662   3670#  3721#  3724
                    3732#  3783#  3786   3794#  3958#  3961   3964#  4014#  4017   4020#  4063#  4066   4069#
                    4109#  4112   4115#  4158#  4161   4164#  4202#  4205   4208#  4246#  4249   4252#  4286#
                    4289   4295#
$ENDAD   003432      123    511    735#   1058
$N     = 000060       1#   1624   1630   1632   1637#  1664   1669   1671   1677#  1698   1703   1705   1711
                    1712#  1735   1741   1743   1749   1750#  1781   1788   1790   1796   1797#  1818   1823
                    1825   1831   1832#  1864   1869   1871   1877   1878#  1912   1918   1920   1926   1927#
                    1980   1986   1988   1993   1994#  2020   2027   2029   2035   2036#  2081   2088   2090
                    2096   2097#  2130   2137   2139   2145   2146#  2187   2193   2195   2201   2202#  2240
                    2246   2248   2254   2255#  2296   2302   2304   2310   2311#  2352   2358   2360   2366
                    2367#  2408   2414   2416   2422   2423#  2464   2470   2472   2478   2479#  2520   2526
                    2528   2534   2535#  2576   2583   2585   2591   2592#  2633   2640   2642   2648   2649#
                    2690   2697   2699   2705   2706#  2747   2754   2756   2762   2763#  2804   2811   2813
                    2819   2820#  2861   2868   2870   2876   2877#  2918   2929   2931   2937   2938#  2980
                    2991   2993   2999   3000#  3039   3050   3052   3058   3059#  3101   3112   3114   3120
```

```
                    3121#  3163   3174   3176   3182   3183#  3225   3236   3238   3244   3245#  3287   3298
                    3300   3306   3307#  3349   3360   3362   3368   3369#  3411   3422   3424   3430   3431#
                    3473   3484   3486   3492   3493#  3535   3546   3548   3554   3555#  3597   3608   3610
                    3616   3617#  3659   3670   3672   3678   3679#  3721   3732   3734   3740   3741#  3783
                    3794   3796   3801   3802#  3958   3964   3966   3971   3972#  4014   4020   4022   4027
                    4028#  4063   4069   4071   4076   4077#  4109   4115   4117   4122   4123#  4158   4164
                    4166   4171   4172#  4202   4208   4210   4215   4216#  4246   4252   4254   4259   4260#
                    4286   4295   4297   4302   4303#  4690#
$$      = 000062      1#   1635   1637#  1674   1677#  1708   1712#  1746   1750#  1793   1797#  1828   1832#
                    1874   1878#  1923   1927#  1991   1994#  2032   2036#  2093   2097#  2142   2146#  2198
                    2202#  2251   2255#  2307   2311#  2363   2367#  2419   2423#  2475   2479#  2531   2535#
                    2588   2592#  2645   2649#  2702   2706#  2759   2763#  2816   2820#  2873   2877#  2934
                    2938#  2996   3000#  3055   3059#  3117   3121#  3179   3183#  3241   3245#  3303   3307#
                    3365   3369#  3427   3431#  3489   3493#  3551   3555#  3613   3617#  3675   3679#  3737
                    3741#  3799   3802#  3969   3972#  4025   4028#  4074   4077#  4120   4123#  4169   4172#
                    4213   4216#  4257   4260#  4303#
$Y      = 000017      1#   207#   215    217#   219#   221#   223#   225#   227#   229#   231#   233#   235#
                    237#   239#   241#   243#   245#
.       = 037430    108#   109    112#   119#   124#   127#   131#   135#   137#   189#   190#   191#   192#
                    273#   278#   280#   281#   282#   283#   285#   286#   287#   288#   290#   291#   292#
                    293#   295#   296#   297#   298#   300#   301#   302#   303#   305#   306#   307#   308#
                    310#   311#   312#   313#   315#   316#   317#   318#   320#   321#   322#   323#   325#
                    326#   327#   328#   330#   331#   332#   333#   335#   336#   337#   338#   340#   341#
                    342#   343#   345#   346#   347#   348#   350#   351#   352#   353#   355#   356#   357#
                    358#   517    527    638#   655    1088   1098   1146#  1181#  1183#  1235   1238   1259
                   1353   1397   1500   1504   1550   1581   1613   1616   2046   2051   2066   2078   3803
                   3805   3808   3820   3823   3826   3832   3837   3867   3876   3883   3892   3906   3911
                   3915   3918   3921   3926   3930   3933   3936   3945   3950   3973   3975   3978   3988
                   4029   4031   4034   4046   4078   4080   4083   4092   4106   4124   4126   4129   4141
                   4155   4173   4175   4178   4183   4199   4217   4219   4222   4227   4231   4243   4261
                   4263   4266   4304   4306   4309   4342   4359   4398   4491   4523   4559   4562   4565
                   4623#  4654#  4655#  4657#  4658#  4661#  4662#  4663#  4664#  4665#  4666#  4667#  4687#
                   4689#  4846   4849   4876   4879
.BEGIN  003062    670#
.CNVRT  004400    234    934#
.CONVB  004374    232    933#
.DATAC  005454    242   1137#
.DELAY  005340    238   1110#
.EOP    003274    711#  4300
.ERRTA  037270   1025  4925#
.HLT    004656    115   1005#
.INSTB  004062    224    840#
.INSTR  003756    222    819#
.INST1  003776    823#   843
.MSG    004000    821#   824#
.MSTCL  005370    236   1121#
.PARAM  004102    226    851#
.PFAIL  005240    113    480   1085#  1093
.RESOS  004342    230    922#
.ROMCL  005406    240   1126#
.SAVCS  004302    228    908#
.SCOPE  003506    216    759#
.SCOP1  003644    218    790#
.START  002002    132    478#   494   1222
.TIMER  005520    244   1148#
.TRPSE  004624    117    993#
```



```
.TRPIA  001330    214#   998
.TYPE   003674    220    801#
```

```
DMEND       1#    705
DMFRNT      1#
HLT        75#   1652   1662   1689   1725   1762   1775   1809   1855   1902   1950   1974   2015   2052   2067
         2079   2119   2158   2171   2184   2213   2225   2237   2267   2280   2293   2323   2336   2349   2379
         2392   2405   2435   2448   2461   2491   2504   2517   2547   2560   2573   2604   2617   2630   2661
         2674   2687   2718   2731   2744   2775   2788   2801   2832   2845   2858   2889   2902   2915   2951
         2964   2977   3012   3024   3036   3072   3085   3098   3134   3147   3160   3196   3209   3222   3258
         3271   3284   3320   3333   3346   3382   3395   3408   3444   3457   3470   3506   3519   3532   3568
         3581   3594   3630   3643   3656   3692   3705   3718   3754   3767   3780   3829   3840   3853   3855
         3870   3879   3886   3895   3904   3907   3912   3916   3919   3922   3927   3931   3934   3937   3946
         3951   3993   3997   4005   4051   4055   4061   4097   4101   4107   4146   4150   4156   4190   4194
         4200   4234   4238   4244   4284   4399   4462   4473   4571   4580   4587   4603   4610
$AUTO       1#    549
$BRRSH      1#   1624
$BUFFF      1#   1177
$BYTE       1#   4626   4629   4632   4635   4638   4641   4644   4647   4650
$CKDAT      1#   4382
$COMP       1#
$CRAM       1#   1664   1698
$CRAMD      1#   1735
$CYCLE      1#   1246
$DATAF      1#   3783
$EOP        1#    705
$EXER       1#   4286
$FD         1#   3857   4861
$FINI       1#   4690
$GETPA      1#
$HALF       1#   4246
$HD         1#   4891
$HEADF      1#
$IOPOP      1#   2020
$JUMP       1#   2130   2187   2240   2296   2352   2408   2464   2520   2576   2633   2690   2747   2804   2861
         2918   2980   3039   3101   3163   3225   3287   3349   3411   3473   3535   3597   3659   3721
$LSTDA      1#   4014
$MARHI      1#
$MEMFI      1#   1832   1878
$MEM0       1#   1864
$MEM1       1#   1918
$MEM2       1#   1912
$MEM3       1#   1980
$MOCK       1#
$MSG        1#   1165
$NONEX      1#   4063   4109
$ORUN       1#   3958
$PFAIL      1#   1081
$PROC       1#   4158
$PROC1      1#   4202
$QUEST      1#   1356   1369   1378   1445   1454
$RAMCI      1#   1109
$RCLK       1#   1112   1115   1152   1157   1642   1644   1646   1653   1656   1840   1842   1845   1847   1849
         1887   1889   1892   1894   1896   1935   1937   1940   1942   1944   1962   1964   1966   1968   1997
         2000   2006   2009   2039   2042   2054   2060   2068   2070   2072   2106   2150   2152   2163   2165
         2176   2178   2205   2207   2217   2219   2229   2231   2259   2261   2272   2274   2285   2287   2315
         2317   2328   2330   2341   2343   2371   2373   2384   2386   2397   2399   2427   2429   2440   2442
         2453   2455   2483   2485   2496   2498   2509   2511   2539   2541   2552   2554   2565   2567   2596
         2598   2609   2611   2622   2624   2653   2655   2666   2668   2679   2681   2710   2712   2723   2725
```



```
         2736   2738   2767   2769   2780   2782   2793   2795   2824   2826   2837   2839   2850   2852   2881
         2883   2894   2896   2907   2909   2943   2945   2956   2958   2969   2971   3004   3006   3016   3018
         3028   3030   3064   3066   3077   3079   3090   3092   3126   3128   3139   3141   3152   3154   3188
         3190   3201   3203   3214   3216   3250   3252   3263   3265   3276   3278   3312   3314   3325   3327
         3338   3340   3374   3376   3387   3389   3400   3402   3436   3438   3449   3451   3462   3464   3498
         3500   3511   3513   3524   3526   3560   3562   3573   3575   3586   3588   3622   3624   3635   3637
         3648   3650   3684   3686   3697   3699   3710   3712   3746   3748   3759   3761   3772   3774   4698
         4700   4702   4710   4718   4726   4734   4742   4744   4746   4754   4780
$RDROM      1#   1781
$ROMRD      1#   2081
$SCOPE      1#    755
$SETUP      1#   3972   4028   4077   4123
$SIMRC      1#
$SKIPT      1#   3802   3972   4028   4077   4123   4172   4216   4260   4303
$SOFTC      1#   1185
$TRPDF      1#    215    217    219    221    223    225    227    229    231    233    235    237    239    241
          243
$TSTN       1#   1632   1671   1705   1743   1790   1825   1871   1920   1988   2029   2090   2139   2195   2248
         2304   2360   2416   2472   2528   2585   2642   2699   2756   2813   2870   2931   2993   3052   3114
         3176   3238   3300   3362   3424   3486   3548   3610   3672   3734   3796   3966   4022   4071   4117
         4166   4210   4254   4297
$VARIA      1#    134
$XZ         1#   1624   1630   1664   1669   1698   1703   1735   1741   1781   1788   1818   1823   1864   1869
         1912   1918   1980   1986   2020   2027   2081   2088   2130   2137   2187   2193   2240   2246   2296
         2302   2352   2358   2408   2414   2464   2470   2520   2526   2576   2583   2633   2640   2690   2697
         2747   2754   2804   2811   2861   2868   2918   2929   2980   2991   3039   3050   3101   3112   3163
         3174   3225   3236   3287   3298   3349   3360   3411   3422   3473   3484   3535   3546   3597   3608
         3659   3670   3721   3732   3783   3794   3958   3964   4014   4020   4063   4069   4109   4115   4158
         4164   4202   4208   4246   4252   4286   4295

. ABS.  037430     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DZDMH,DZDMH/SOL/CRF_IPLUTL,DZDMH
 RUN-TIME: 51 72 5 SECONDS
 RUN-TIME RATIO: 259/130=1.9
 CORE USED:  29K  (57 PAGES)
```