DMC11

CROM AND JUMP TESTS

MD-11-DZDMG-C

EP-DZDMG-C-DL-A

FICHE 1 OF 2

AUG 1977

digital

MADE IN USA

DMC11

CROM AND JUMP TESTS

MD-11-DZDMG-C

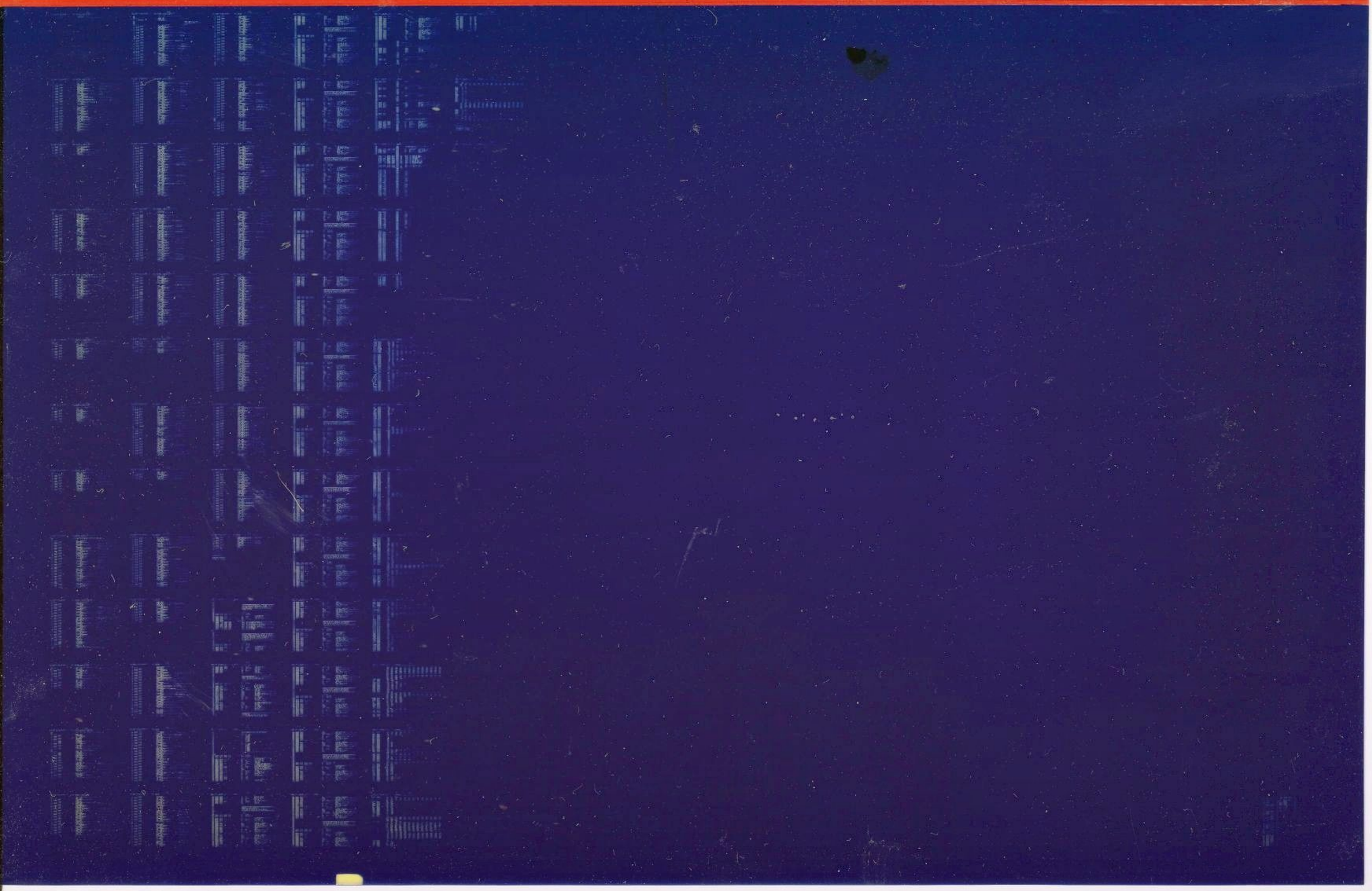
EP-DZDMG-C-DL-A

FICHE 2 OF 2

AUG 1977

digital

MADE IN USA



IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZDMG-C-D
PRODUCT NAME: DMC11 CROM AND JUMP TESTS
DATE: MAY 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.

DZDMG tests the DMC11-AR and DMC11-AL micro-processors (M8200-YA M8200-YB). It performs jump tests on the micro-processor and verifies the control ROM of the M8200. This diagnostic will not run on a KMC (M8204), however it is possible to load the KMC CRAM with the DMC micro-code. See test 2 for details.

Currently there are five 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 five 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] Jump and CROM tests
5. DZDMH [REV] Free-running tests (Heat test tape)

2. REQUIREMENTS

2.1 EQUIPMENT

Any PDP11 family CPU (except an LSI-11) with minimum 8k memory
ASR 33 (or equilivalent)
DMC11-AR (M8200-YA) or an DMC11-AL (M8200-YB)

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 strongly 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.

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 DZDMG 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
BIT1=0 DMC11-AR (LOW SPEED)
BIT1=1 DMC11-AL (HIGH SPEED)

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 626 or 16520 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, a 626 indicates a DMC11-AL, and a 16520 indicates a DMC11-AL. 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
****
DZDMG LST
****

```
6       MAINDEC-11-DZDMG-C   DMC11 CROM AND JUMP TESTS
        COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
        ------------------------------------------------------------

1675    ************************** TEST 1 ****************************
        THIS IS A SPECIAL TEST WHICH IS ONLY EXECUTED  ONE TIME,
        THE FIRST PASS AFTER THE DIAGNOSTIC IS LOADED. IT TYPES ON
        THE CONSOLE THE PART NUMBERS OF THE CROMS WHICH THIS
        REVISION SUPPORTS. TO FORCE A TYPE OUT PATCH LOCATION
        ROMNUM: TO A ZERO.

1696    ************************** TEST 2 ****************************
        THIS IS A SPECIAL TEST WHICH WILL RUN ON A KMC (DMC WITH
        WRITTABLE CONTROL STORE) TO LOAD THE CRAM WITH THE DDCMP
        MICRO-CODE. FIRST BE SURE BIT1 OF STAT3 IS SET UP AS FOLLOWS
        1=LOCAL HIGH SPEED CODE,  0=REMOTE LOW SPEED CODE  THE STATUS
        OF STAT3 BIT1 DETERMINES WHICH MICRO-CODE WILL
        BE LOADED IN THE KMC. LOOP ON THIS TEST FOR A FEW SECONDS
        TO LOAD THE KMC.

1727    ************************** TEST 3 ****************************
        TEST OF BR RIGHT SHIFT
        VERIFY THAT A DEST OF BR RSH (011) OF A MICRO-INSTRUCTION
        SHIFTS THE RESULTING BR DATA RIGHT ONCE.

1768    ************************** TEST 4 ****************************
        CROM READ TEST
        THIS TEST READS EACH ROM LOCATION AND COMPARES
        IT TO A SOFTWARE DUPLICATE OF THE CROM. THIS TEST
        ALSO TESTS THE JUMP(I) MICRO-PROCESSOR INSTRUCTION.

1773    IF THIS TEST FAILS CHECK YOUR CROM PART NUMBERS.
        DZDMG-C SUPPORTS THE FOLLOWING PART NUMBERS:

        DMC11-AR  (M8200-YA)
                23-414A9
                23-415A9
                23-416A9
                23-417A9
                23-418A9
                23-419A9
                23-420A9
                23-421A9

        DMC11-AL  (M8200-YB)
                23-392A9
                23-393A9
                23-394A9
                23-395A9
                23-396A9
                23-397A9
                23-398A9
                23-399A9
```

```
1840    ************************** TEST 5 ****************************
        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).

1898    ************************** TEST 6 ****************************
        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

1952    ************************** TEST 7 ****************************
        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

2009    ************************** TEST 10 ***************************
        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

2066    ************************** TEST 11 ***************************
        CROM TEST OF JUMP(I) ON BRO SET MICRO-PROCESSOR INSTRUCTION.
        SET THE BRO BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2123    ************************** TEST 12 ***************************
        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

2180    ************************** TEST 13 ***************************
        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

2237    ************************** TEST 14 ***************************
        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

2294    ************************** TEST 15 ***************************
        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).

2352    ************************** TEST 16 ***************************
        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).
```

```
2410    ************************** TEST 17 ****************************
        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).

2468    ************************** TEST 20 ****************************
        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).

2526    ************************** TEST 21 ****************************
        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).

2584    ************************** TEST 22 ****************************
        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).
```

```
                        ;*MAINDEC-11-DZDMG-C   DMC11 CROM AND JUMP 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-DZDMG-C   DMC11 CROM AND JUMP TESTS"
                        ;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".
                                ;-----------------------------------------------------------------
                                ;;*****************************************************************

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

                                .=24
000024  005336                          .PFAIL                  ;POWER FAIL HANDLER
000026  000340                          340                     ;SERVICE AT LEVEL 7
000030  004750                          .HLT                    ;ERROR HANDLER
000032  000340                          340                     ;SERVICE AT LEVEL 7
000034  004716                          .TRPSRV                 ;GENERAL HANDLER DISPATCH SERVICE
000036  000340                          340                     ;SERVICE AT LEVEL 7
                                .=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  003522                          $ENDAD                  ;FOR USE WITH ACT-11 OR XXDP
                                .=52
000052  000000                          0                       ;ACT-11 PROGRAM CHARACTERISTICS

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

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

                                .=1000
001000  005377  040515  047111  MTITLE: .ASCII  <377><12>/MAINDEC-11-DZDMG-C/<377>
(2) 001025  104  041515  030461          .ASCIZ  /DMC11 CROM AND JUMP TESTS/<377>
(2)
                                .=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  003576                          .SCOPE
        104401                  SCOP1=TRAP+1            ;CALL TO LOOP ON CURRENT DATA HANDLER
001332  003736                          .SCOP1
        104402                  TYPE=TRAP+2             ;CALL TO TELETYPE OUTPUT ROUTINE
001334  003766                          .TYPE
        104403                  INSTR=TRAP+3            ;CALL TO ASCII STRING INPUT ROUTINE
001336  004050                          .INSTR
        104404                  INSTER=TRAP+4           ;CALL TO INPUT ERROR HANDLER
001340  004154                          .INSTER
        104405                  PARAM=TRAP+5            ;CALL TO NUMERICAL DATA INPUT ROUTINE
001342  004174                          .PARAM
        104406                  SAVOS=TRAP+6            ;CALL TO REGISTER SAVE ROUTINE
001344  004374                          .SAVOS
        104407                  RESOS=TRAP+7            ;CALL TO REGISTER RESTORE ROUTINE
001346  004434                          .RESOS
        104410                  CONVRT=TRAP+10          ;CALL TO DATA OUTPUT ROUTINE
001350  004466                          .CONVRT
        104411                  CNVRT=TRAP+11           ;CALL TO DATA OUTPUT ROUNTINE WITHOUT CR/LF.
001352  004472                          .CNVRT
        104412                  MSTCLR=TRAP+12          ;CALL TO ISUE A MASTER CLEAR
001354  005466                          .MSTCLR
        104413                  DELAY=TRAP+13           ;CALL TO DELAY
001356  005436                          .DELAY
        104414                  ROMCLK=TRAP+14          ;CALL TO CLOCK ROM ONCE
001360  005504                          .ROMCLK
        104415                  DATACLK=TRAP+15         ;CALL TO CLK DATA
001362  005552                          .DATACLK
        104416                  TIMER=TRAP+16           ;CALL TO DELAY A CLOCK TICK
001364  005616                          .TIMER

                                ;-----------------------------------------------------------------
                                ;:*****************************************************************
```

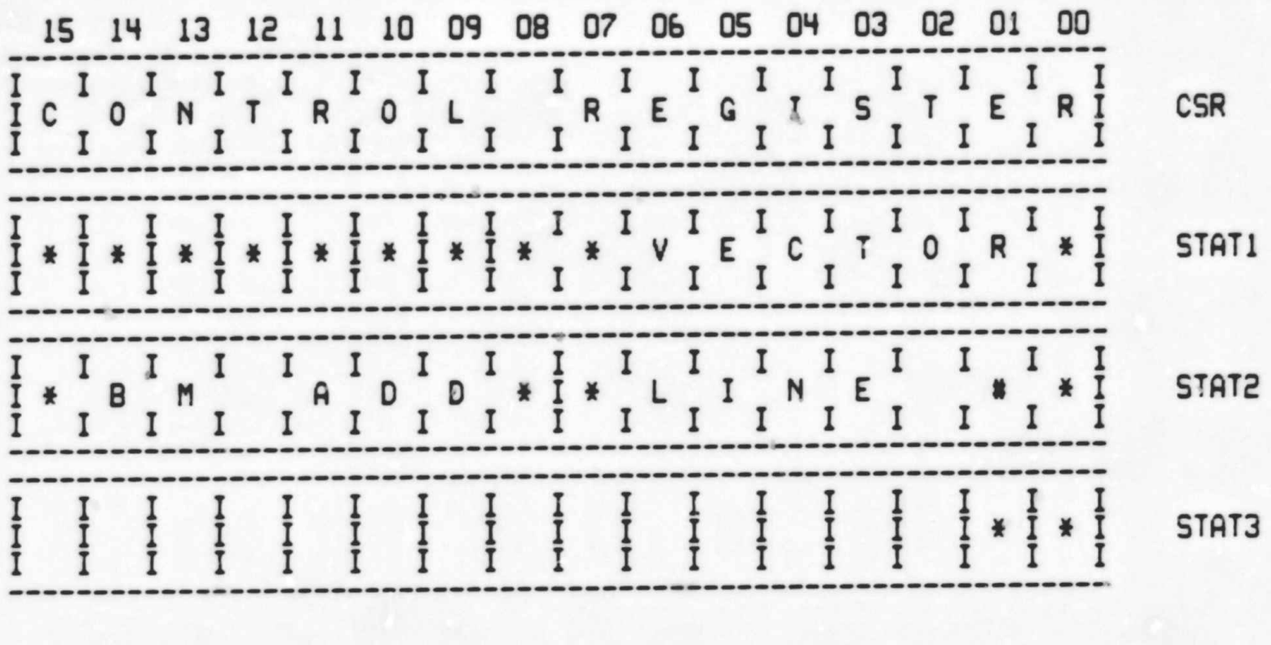
```
                                        ;DMC11 CONTROL INDICATORS FOR CURRENT DMC11 UNDER TEST
                                        ;----------------------------------------------------

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                          DMP04:  0               ;POINTER TO DMC11 PORT REGISTER(SEL 4)
001414  000000                          DMP06:  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 REGISTER 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
```

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

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

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

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

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

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

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

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

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

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

                                DMCR17: .BLKW   1               ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 17
                                DMS117: .BLKW   1               ;VECTOR FOR DMC11 NUMBER 17
                                DMS217: .BLKW   1               ;DDCMP LINE# FOR DMC11 NUMBER 17
                                DMS317: .BLKW   1               ;3RD STATUS WORD

                                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
```

413

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 [PROGRAM DZDMI ONLY])
KMC MUST HAVE MICRO-CODE WRITTEN FROM RUNNING
DZDMG TEST 2 FIRST
BIT1=1 DMC11-AL LOCAL HIGH SPEED MICRO-CODE
BIT1=0 DMC11-AR REMOTE LOW SPEED MICRO-CODE

```
                                                ;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  005336  000024                MOV     #.PFAIL,@#24    ;SET UP POWER FAIL VECTOR
002022  013737  001310  001314                MOV     DMNUM,SAVNUM    ;SAVE NUMBER OF DEVICES IN SYSTEM.
002030  005037  010016                        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$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
002170  012737  000176  001202                MOV     #SWREG,SWR      ;POINTER TO SOFT SWR
002176  012737  000174  001200                MOV     #DISPREG,DISPLAY;POINTER TO SOFT DISPLAY REG
002204  012637  000004                7$:     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  003522  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  007606                20$:    JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
002242  017737  176734  001236                MOV     @SWR,STRTSW     ;STORE STARTING SWITCHES
002250  005737  000042                        TST     @#42            ;IS IT RUNNING IN AUTO MODE?
002254  001402                                BEQ     .+6             ;BR IF NO
002256  005037  001236                        CLR     STRTSW          ;IF YES, CLEAR SWITCHES
002262  032737  000001  001236                BIT     #SW00,STRTSW    ;IF SW00=1, QUESTIONS ARE ASKED.
002270  001012                                BNE     17$             ;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  007154                        TYPE,   NOACT           ;NO DEVICES SELECTED.
```

```
                                                HALT                    ;STOP THE SHOW
                                                BR      .-2             ;DISQUALIFY CONTINUE SWITCH
                                        17$:    JSR     PC,AUTO.SIZE    ;GO DO THE AUTO SIZE
                                        16$:    TSTB    INIFLG          ;FIRST TIME?
                                                BEQ     21$             ;BR IF YES
                                                TSTB    STRTSW          ;IF USING SAME PARAMETERS DONT TYPE MAP
                                                BMI     1$
                                                BIT     #BIT1!BIT2,STRTSW;IS TEST NO. OR LOCK SELECTED
                                                BEQ     24$             ;IF NO THEN TYPE STATUS
                                                BR      1$              ;IF YES DO NOT TYPE STATUS
                                        21$:    COM     INIFLG          ;SET FLAG
                                        24$:    TYPE    ,XHEAD          ;TYPE HEADER
                                                MOV     #DM.MAP,R4      ;SET POINTER
                                        5$:     MOV     R4,TEMP1        ;SET ADDRESS
                                                MOV     (R4)+,TEMP2     ;SET CSR
                                                BEQ     1$              ;ALL DONE IF ZERO
                                                MOV     (R4)+,TEMP3     ;SET STAT1
                                                MOV     (R4)+,TEMP4     ;SET STAT2
                                                MOV     (R4)+,TEMP5     ;SET STAT3
                                                CONVRT                  ;TYPE OUT STATUS MAP
                                                XSTATQ                  ;
                                                BR      5$
                                        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. THIS TEST MAY REQUIRE 2 OR MORE ATTEMPTS TO GET THE
;;*RIGHT ADDRESSES. AFTER YOU HAVE CHANGED THE ADDRESS TO WHAT IT TOLD
;;*YOU THE FIRST TIME, IT MAY COME BACK AND TELL YOU A DIFFERENT ADDRESS
;;*THE NEXT TIME YOU RUN IT. PLEASE HAVE PATIENCE, THE FINAL ADDRESS
;;*WILL BE CORRECT (AS LONG AS ALL DEVICES IN FRONT OF THE DMC'S ARE
;;*CORRECT).
;;*************************************************************************

                                                MOV     @#4,-(SP)       ;SAVE LOC 4
                                                MOV     @#6,-(SP)       ;SAVE LOC 6
                                                CLR     @#6             ;CLEAR VEC+2
                                                CLR     TEMP3           ;CLEAR FLAG
                                                CLR     R5              ;R5=0=DMC, R5=-1=KMC
                                        AUSTRT: MOV     (R0),DMCSR      ;GET NEXT DMC CSR
                                                BEQ     AUDONE          ;BR IF DONE
                                                TST     R5              ;DMC OR KMC?
                                                BNE     1$              ;BR IF KMC
                                                BIT     #BIT15,2(R0)    ;CHECK FOR DMC CSR
                                                BNE     SKIP            ;SKIP IF NOT DMC
                                                BR      2$              ;ITS A DMC SO CONTINUE
                                        1$:     BIT     #BIT15,2(R0)    ;CHECK FOR KMC CSR
                                                BEQ     SKIP            ;SKIP IF NOT KMC
                                        2$:     MOV     #NODEV,@#4      ;SET UP FOR TIMEOUT
                                                TST     R5              ;DMC OR KMC?
                                                BNE     3$              ;BR IF KMC
```

```
  580  002514  012703  000006                    MOV     #6,R3           ;R3 IS COUNT OF DEVICES BEFORE DMC
  581  002520  000402                            BR      4$              ;GO ON
  582  002522  012703  000010              3$:   MOV     #10,R3          ;R3 IS COUNT OF DEVICES BEFORE KMC
  583  002526  012702  003010              4$:   MOV     #DEVTAB,R2      ;R2 IS DEVICE TABLE PONTER
  584  002532  012701  160010                    MOV     #160010,R1      ;START WITH ADDRESS 160010
  585  002536  005711                      FLOAT: TST     (R1)            ;CHECK ADDRESS IN R1
  586  002540  111204                            MOVB    (R2),R4         ;IF NO TIMEOUT, GET NEXT ADDRESS
  587  002542  060401                            ADD     R4,R1           ;IN R1
  588  002544  005201                            INC     R1              ;
  589  002546  040401                            BIC     R4,R1           ;
  590  002550  005703                            TST     R3              ;ANY MORE DEVICES TO CHECK FOR?
  591  002552  001371                            BNE     FLOAT           ;BR IF YES
  592  002554  012737  002700  000004            MOV     #ERR,@#4        ;OK ONLY DMC'S ARE LEFT, SET UP FOR TIMEOUT
  593  002562  010137  003022                    MOV     R1,XLOC         ;SAVE FIRST DMC/KMC ADDRESS
  594  002566  005705                      FY:   TST     R5              ;DMC OR KMC?
  595  002570  001005                            BNE     1$              ;BR IF KMC
  596  002572  032760  100000  000002            BIT     #BIT15,2(R0)    ;CHECK FOR DMC CSR
  597  002600  001014                            BNE     SKIP            ;SKIP IF NOT DMC
  598  002602  000404                            BR      2$              ;ITS A DMC SO CONTINUE
  599  002604  032760  100000  000002      1$:   BIT     #BIT15,2(R0)    ;CHECK FOR KMC CSR
  600  002612  001407                            BEQ     SKIP            ;SKIP IF NOT KMC
  601  002614  005711                      2$:   TST     (R1)            ;CHECK DMC ADDRESS
  602  002616  020137  001404                    CMP     R1,DMCSR        ;DOES IT MATCH
  603  002622  001411                            BEQ     OK              ;BR IF YES
  604  002624  062701  000010                    ADD     #10,R1          ;GET NEXT DMC ADDRESS
  605  002630  000756                            BR      FY              ;DO IT AGAIN
  606  002632  062700  000010              SKIP: ADD     #10,R0          ;SKIP TO NEXT CSR IN TABLE
  607  002636  011037  001404                    MOV     (R0),DMCSR      ;GET NEXT CSR
  608  002642  001470                            BEQ     AUDONE          ;BR IF DONE
  609  002644  000750                            BR      FY              ;ELSE CONTINUE
  610  002646  062700  000010              OK:   ADD     #10,R0          ;SKIP TO NEXT DMC CSR
  611  002652  062737  000010  003022            ADD     #10,XLOC        ;UPDATE EXPECTED DMC/KMC ADDRESS
  612  002660  011037  001404                    MOV     (R0),DMCSR      ;GET NEXT DMC/KMC CSR
  613  002664  001457                            BEQ     AUDONE          ;BR IF DONE
  614  002666  013701  003022                    MOV     XLOC,R1         ;GET EXPECTED DMC/KMC ADDRESS
  615  002672  000735                            BR      FY              ;CONTINUE
  616  002674  122243                      NODEV: CMPB    (R2)+,-(R3)     ;ON TIMEOUT, INC R2, DEC R3
  617  002676  000002                            RTI                     ;RETURN
  618  002700  005737  001252              ERR:  TST     TEMP3           ;CHECK FLAG IF = 0 TYPE HEADER
  619  002704  001014                            BNE     1$              ;SKIP HEADER
  620  002706  104402                            TYPE                    ;TYPEOUT HEADER MESSAGE
  621  002710  007223                            CONERR                  ;CONFIGURATION ERROR!!!!
  622  002712  012737  002700  001276            MOV     #ERR,SAVPC      ;SAVE PC FOR TYPEOUT
  623  002720  104411                            CNVRT                   ;TYPE OUT ERROR PC
  624  002722  002770                            ERRPC                   ;
  625  002724  104402                            TYPE                    ;TYPE REST OF HEADER
  626  002726  007277                            CNERR                   ;
  627  002730  012737  177777  001252            MOV     #-1,TEMP3       ;SET FLAG SO IT ONLY GETS TYPED ONCE
  628  002736  010137  001262              1$:   MOV     R1,SAVR1        ;SAVE R1 FOR TYPEOUT
  629  002742  104410                            CONVRT
  630  002744  002776                            CONTAB                  ;TYPE CSR VALUES
  631  002746  005705                            TST     R5              ;DMC OR KMC ?
  632  002750  001003                            BNE     3$              ;BR IF KMC
  633  002752  104402                            TYPE
  634  002754  007320                            DMCM
  635  002756  000402                            BR      4$              ;CONTINUE
```

```
   636  002760  104402                        3$:     TYPE
   637  002762  007330                                KMCM
   638  002764  022626                        4$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
   639  002766  000727                                BR      OK              ;BR TO GET OUT
   640  002770  000001                        ERRPC:  1
   641  002772     006      002                        .BYTE   6,2
   642  002774  001276                                SAVPC
   643  002776  000002                        CONTAB: 2
   644  003000     006      004                        .BYTE   6,4
   645  003002  003022                                XLOC
   646  003004     006      002                        .BYTE   6,2
   647  003006  001404                                DMCSR
   648  003010     007                         DEVTAB: .BYTE   7               ;DJ
   649  003011     017                                 .BYTE   17              ;DH
   650  003012     007                                 .BYTE   7               ;DQ
   651  003013     007                                 .BYTE   7               ;DU
   652  003014     007                                 .BYTE   7               ;DUP
   653  003015     007                                 .BYTE   7               ;LK
   654  003016     007                                 .BYTE   7               ;DMC
   655  003017     007                                 .BYTE   7               ;DZ
   656  003020     007                                 .BYTE   7               ;KMC
   657     003022                             .EVEN
   658  003022  000000                        XLOC:   0
   659  003024  005705                        AUDONE: TST     R5              ;DMC?
   660  003026  001005                                BNE     1$              ;BR IF KMC AND ALL DONE
   661  003030  012705  177777                        MOV     #-1,R5          ;SET R5 TO -1 (KMC)
   662  003034  012700  001500                        MOV     #DM.MAP,R0      ;RESET R0 TO START OF TABLE
   663  003040  000602                                BR      AUSTRT          ;GO DO KMC'S
   664  003042  012637  000006                1$:     MOV     (SP)+,@#6       ;RESTORE LOC 6
   665  003046  012637  000004                        MOV     (SP)+,@#4       ;RESTORE LOC 4
   666  003052  032737  000010  001236                BIT     #SW03,STRTSW    ;SELECT SPECIFIC DEVICES??
   667  003060  001422                                BEQ     3$              ;BR IF NO.
   668  003062  104402  006144                        TYPE    ,MNEW           ;TYPE THE MESSAGE.
   669  003066  005000                                CLR     R0              ;ZERO DATA LIGHTS
   670  003070  000000                                HALT                    ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
   671  003072  027737  176104  001312                CMP     @SWR,SAVACT     ;IS THE NUMBER VALID?
   672  003100  101404                                BLOS    2$              ;BR IF NUMBER IS OK.
   673  003102  104402  006005                        TYPE    ,MERR3          ;TELL USER OF INVALID NUMBER.
   674  003106  000000                                HALT                    ;STOP EVERY THING.
   675  003110  000776                                BR      .-2             ;RESTART THE PROGRAM AGAIN.
   676  003112  017737  176064  001306        2$:     MOV     @SWR,DMACTV     ;GET NEW DEVICE PATTERN
   677  003120  013700  001306                        MOV     DMACTV,R0       ;SHOW THE USER WHAT HE SELECTED.
   678  003124  000000                                HALT                    ;CONTINUE DYNAMIC SWITCHES.
   679  003126  012700  000300                3$:     MOV     #300,R0         ;PREPARE TO CLEAR THE FLOATING
   680  003132  012701  000302                        MOV     #302,R1         ;VECTOR AREA. 300-776
   681  003136  010120                        4$:     MOV     R1,(R0)+        ;START PUTTING "PC+2 - HALT"
   682  003140  005021                                CLR     (R1)+           ;IN VECTOR AREA.
   683  003142  022021                                CMP     (R0)+,(R1)+     ;POP POINTERS
   684  003144  022700  001000                        CMP     #1000,R0        ;ALL DONE??
   685  003150  001372                                BNE     4$              ;BR IF NO.
   686
   687                                     ;TEST START AND RESTART
   688                                     ;----------------------
   689
   690  003152  012706  001200                .BEGIN: MOV     #STACK,SP       ;SET UP STACK
   691  003156  013746  000006                        MOV     @#6,-(SP)       ;SAVE LOC 6
```

```
 692  003162  013746  000004                        MOV     @#4,-(SP)       ;SAVE LOC 4
 693  003166  005000                                CLR     R0              ;START AT 0
 694  003170  012737  003234  000004                MOV     #2$,@#4         ;SET UP FOR TIME OUT
 695  003176  005037  000006                        CLR     @#6             ;TO AUTOSIZE MEMORY
 696  003202  005720                        6$:     TST     (R0)+           ;CHECK ADDRESS IN R0
 697  003204  022700  157776                        CMP     #157776,R0      ;IS IT AT LEAST 28K
 698  003210  001374                                BNE     6$              ;BR IF NO
 699  003212  162700  007776                        SUB     #7776,R0        ;SAVE 2K FOR MONITORS
 700  003216  010037  001304                7$:     MOV     R0,MEMLIM       ;STORE MEMORY LIMIT
 701  003222  012637  000004                        MOV     (SP)+,@#4       ;RESTORE LOC 4
 702  003226  012637  000006                        MOV     (SP)+,@#6       ;RESTORE LOC 6
 703  003232  000413                                BR      10$             ;CONTINUE
 704  003234  022626                        2$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
 705  003236  162700  000004                        SUB     #4,R0           ;GET LAST GOOD ADDRESS
 706  003242  162700  007776                        SUB     #7776,R0        ;SAVE 2K FOR MONITORS
 707  003246  022700  030000                        CMP     #30000,R0       ;IS IT 8K?
 708  003252  001361                                BNE     7$              ;BR IF NO
 709  003254  012700  037400                        MOV     #37400,R0       ;IF 8K DON'T SAVE 2K
 710  003260  000756                                BR      7$              ;
 711  003262  012737  000340  177776        10$:    MOV     #340,PS         ;LOCK OUT INTERRUPTS
 712  003270  032737  000004  001236                BIT     #BIT2,STRTSW    ;CHECK FOR LOCK ON TEST
 713  003276  001411                                BEQ     1$              ;BR IF NO LOCK DESIRED.
 714  003300  104402  006043                        TYPE    ,MLOCK          ;TYPE LOCK SELECTED.
 715  003304  012737  000240  003612                MOV     #NOP,TTST       ;ADJUST SCOPE ROUTINE.
 716  003312  012737  000240  003614                MOV     #NOP,TTST+2     ;SET UP TO LOCK
 717  003320  000406                                BR      3$              ;CONTINUE ALONG.
 718  003322  013737  003730  003612        1$:     MOV     BRW,TTST        ;PREPARE NORMAL SCOPE ROUTINE
 719  003330  013737  003732  003614                MOV     BRX,TTST+2      ;LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
 720  003336  012737  010060  001214        3$:     MOV     #CYCLE,RETURN   ;START AT "CYCLE" FIND WHICH DEVICE TO TEST
 721  003344  032737  000002  001236        4$:     BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED?
 722  003352  001002                                BNE     5$              ;BR IF YES
 723  003354  104402  005755                        TYPE    ,MR             ;TYPE R
 724  003360  000177  175630                5$:     JMP     @RETURN         ;START TESTING
```

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

003364  000005                                .EOP:   RESET                   ;MAKE THE  WORLD CLEAN AGAIN.
003366  005037  001234                                CLR     LSTERR          ;CLEAR LAST ERROR PC
003372  105037  001325                                CLRB    ERRFLG          ;CLEAR ERROR FLAG
003376  005237  001230                                INC     PASCNT          ;UPDATE PASS COUNT
003402  013777  001230  175570                        MOV     PASCNT,@DISPLAY ;DISPLAY PASS COUNT
003410  104402  005733                                TYPE    ,MEPASS         ;TYPE END PASS
003414  104402  006072                                TYPE    ,MCSRX          ;TYPE CSR
003420  104411  003546                                CNVRT   ,XCSR           ;SHOW IT
003424  104402  006100                                TYPE    ,MVECX          ;TYPE VECTOR
003430  104411  003554                                CNVRT   ,XVEC           ;SHOW IT
003434  104402  006106                                TYPE    ,MPASSX         ;TYPE PASSES
003440  104411  003562                                CNVRT   ,XPASS          ;SHOW IT
003444  104402  006117                                TYPE    ,MERRX          ;TYPE ERRORS
003450  104411  003570                                CNVRT   ,XERR           ;SHOW IT
003454  013700  001322                                MOV     MILK,R0         ;GET POINTER TO PASS COUNT
003460  013720  001230                                MOV     PASCNT,(R0)+    ;STORE PASS COUNT FOR THIS DMC11
003464  013720  001232                                MOV     ERRCNT,(R0)+    ;STORE ERROR COUNT FOR THIS DMC11
003470  005337  001314                                DEC     SAVNUM          ;ARE ALL DEVICES TESTED?
003474  001017                                        BNE     RESTRT          ;BR IF NO.
003476  112737  000377  001327                        MOVB    #377,QV.FLG     ;SET THE QUICK VERIFY FLAG.
003504  013737  001310  001314                        MOV     DMNUM,SAVNUM    ;RESTORE THE COUNT
003512  013701  000042                                MOV     @#42,R1         ;CHECK FOR ACT-11 OR DDP
003516  001406                                        BEQ     RESTRT          ;IF NOT, CONTINUE TESTING
003520  000005                                        RESET                   ;STOP THE SHOW--CLEAR THE WORLD
003522                                        $ENDAD:
003522  004711                                        JSR     PC,(R1)
003524  000240                                        NOP
003526  000240                                        NOP
003530  000240                                        NOP
003532  000240                                        NOP
003534  012737  010060  001214                RESTRT: MOV     #CYCLE,RETURN
003542  000137  010060                                JMP     CYCLE
003546  000001                                XCSR:   1
003550     006     002                                .BYTE   6,2
003552  001404                                        DMCSR
003554  000001                                XVEC:   1
003556     004     002                                .BYTE   4,2
003560  001374                                        DMRVEC
003562  000001                                XPASS:  1
003564     006     002                                .BYTE   6,2
003566  001230                                        PASCNT
003570  000001                                XERR:   1
003572     006     002                                .BYTE   6,2
003574  001232                                        ERRCNT

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

003576  004737  007606                        .SCOPE: JSR     PC,CKSWR        ;CKECK FOR SOFT SWR
003602  010016                                        MOV     R0,(SP)         ;SAVE R0 ON THE STACK
```

```
                                                        BIT     #BIT14,@SWR     ;"LOOP ON THIS TEST"?
                                                TTST:   BEQ     1$              ;BR IF NO.  (IF LOCK SW01=1; THIS LOC =240)
                                                        BR      3$              ;GOTO 3$    (IF LOCK SW01=1; THIS LOC =240)
                                                        TST     DONE            ;WAS TKCSR DONE SET?
                                                        BEQ     3$              ;BR IF NO (LOCKED ON TEST)
                                                        CLR     DONE            ;YES, CLEAR FLAG
                                                        BR      2$              ;GO TO NEXT TEST
                                                1$:     BIT     #SW11,@SWR      ;DELETE ITERATION?  (QUICK PASS)
                                                        BNE     2$              ;BR IF YES
                                                        TSTB    QV.FLG          ;HAVE PASSES BEECOMPLETED?
                                                        BEQ     2$              ;BR IF QUICK PASS.
                                                        INC     LPCNT           ;UPDATE ITERATION COUNTER
                                                        CMP     LPCNT,ICOUNT    ;ARE ALL ITERATIONS DONE??
                                                        BLOS    3$              ;BR IF NOT YET
                                                2$:     CLRB    ERRFLG          ;PREPARE FOR NEW TEST
                                                        CLR     LPCNT           ;START ICOUNTER AT 0
                                                        CLR     LOCK
                                                        MOV     #20,ICOUNT      ;RESET ITERATIONS
                                                        MOV     NEXT,RETURN     ;GET NEXT TEST
                                                3$:     MOV     (SP),R0         ;POP R0 OFF OF THE STACK
                                                        POP2SP                  ;FAKE AN "RTI"
                                                        MOV     DMCSR,R1        ;R1 CONTAINS BASE DMC ADDRESS
                                                        JMP     @RETURN         ;GO DO THE TEST
                                                BRW:    1407
                                                BRX:    437
                                                DONE:   0

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

003736  004737  007606                          .SCOP1: JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
003742  032777  001000  175232                          BIT     #SW09,@SWR      ;IS SW09=1(SET)?
003750  001405                                          BEQ     1$              ;BR IF NOT SET.
003752  005737  001220                                  TST     LOCK
003756  001402                                          BEQ     1$
003760  013716  001220                                  MOV     LOCK,(SP)       ;GOTO THE ADDRESS IN LOCK.
003764  000002                                  1$:     RTI                     ;GO BACK.

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

003766  010546                                  .TYPE:  MOV     R5,-(SP)        ;SAVE R5 ON THE STACK.
003770  017605  000002                                  MOV     @2(SP),R5       ;GET ADDRESS OF MESSAGE.
003774  062766  000002  000002                          ADD     #2,2(SP)        ;POP OVER ADDRESS.
004002  005737  010016                          4$:     TST     SWFLG           ;SOFT SWR MESSAGE?
004006  001004                                          BNE     1$              ;IF YES TYPE IT OUT REGARDLESS OF SW12
004010  032777  010000  175164                          BIT     #SW12,@SWR      ;INHIBIT ALL PRINT OUT??
004016  001012                                          BNE     3$              ;BR IF NO PRINT OUT WANTED (SW12=1)
004020  105715                                  1$:     TSTB    (R5)            ;IS NUMBER MINUS? (MSB=1(BIT7))
004022  100002                                          BPL     2$              ;BR IF NUMBER IS PLUS
004024  104402  005672                                  TYPE    ,MCRLF          ;TYPE A CR/LF!
004030  105777  175154                          2$:     TSTB    @TPCSR          ;TTY READY?
004034  100375                                          BPL     2$              ;BR IF NO.
004036  112577  175150                                  MOVB    (R5)+,@TPDBR    ;PRINT CURRENT CHAR.
004042  001357                                          BNE     4$              ;IF NOT ZERO KEEP PRINTING!
004044  012605                                  3$:     MOV     (SP)+,R5        ;END OF OUTPUT. RESTORE R5
```

```
                                        RTI                     ;GO HOME
                                        ;-----------------------
                                .INSTR: MOV     R3,-(SP)        ;SAVE R3 ON STACK
                                        MOV     R4,-(SP)        ;SAVE R4 ON STACK
                                        MOV     @4(SP),.MSG
                                        ADD     #2,4(SP)
                                .INST1: TYPE
                                .MSG:   0
                                        MOV     #INBUF,R4
                                        MOV     #7,R3
                                1$:     TSTB    @TKCSR
                                        BPL     1$
                                        MOVB    @TKDBR,(R4)
                                        BICB    #200,(R4)
                                        CMPB    (R4)+,#15
                                        BEQ     INSTR2
                                2$:     TSTB    @TPCSR
                                        BPL     2$
                                        MOV     @TKDBR,@TPDBR
                                        DEC     R3
                                        BNE     1$
                                        MOV     (SP)+,R4
                                        MOV     (SP)+,R3
                                .INSTE: TYPE    ,MQM
                                        MOV     R3,-(SP)
                                        MOV     R4,-(SP)
                                        BR      .INST1
                                INSTR2: MOV     (SP)+,R4        ;RESTORE R4
                                        MOV     (SP)+,R3        ;RESTORE R3
                                        RTI

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

                                .PARAM: MOV     R5,-(SP)
                                        MOV     R4,-(SP)
                                        MOV     4(SP),R5
                                        MOV     (R5)+,LOLIM
                                        MOV     (R5)+,HILIM
                                        MOV     (R5)+,DEVADR
                                        MOVB    (R5)+,LOBITS
                                        MOVB    (R5)+,ADRCNT
                                        MOV     R5,4(SP)
                                PARAM1: CLR     R5
                                        MOV     #INBUF,R4
                                        CMPB    #15,(R4)
                                        BEQ     PARERR
                                1$:     CMPB    (R4),#60
                                        BLT     PARERR
                                        CMPB    (R4),#67
                                        BGT     PARERR
                                        BICB    #60,(R4)
                                        BISB    (R4)+,R5
                                        CMPB    #15,(R4)
                                        BEQ     LIMITS
```

```
                        ASL     R5
                        ASL     R5
                        ASL     R5
                        BR      1$
PARERR:                 INSTER
                        BR      PARAM1

                        ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
                        ;--------------------------------------

LIMITS:                 CMP     R5,HILIM
                        BHI     PARERR
                        CMP     R5,LOLIM
                        BLO     PARERR
                        BITB    LOBITS,R5
                        BNE     PARERR

                        ;STORE NUMBER AT SPECIFIED ADDRESS

                        MOV     DEVADR,R4
1$:                     MOV     R5,(R4)+
                        ADD     #2,R5
                        DECB    ADRCNT
                        BNE     1$
                        MOV     (SP)+,R4
                        MOV     (SP)+,R5
                        RTI
LOLIM:                  0
HILIM:                  0
DEVADR:                 0
LOBITS:                 0
ADRCNT=LOBITS+1

                        ;SAVE PC OF TEST THAT FAILED AND R0-R5
                        ;-------------------------------------

.SAV05:                 MOV     4(SP),SAVPC     ;SAVE R7 (PC)

                        ;SAVE R0-R5

SV05:                   MOV     R5,SAVR5        ;SAVE R5
                        MOV     R4,SAVR4        ;SAVE R4
                        MOV     R3,SAVR3        ;SAVE R3
                        MOV     R2,SAVR2        ;SAVE R2
                        MOV     R1,SAVR1        ;SAVE R1
                        MOV     R0,SAVR0        ;SAVE R0
                        RTI                     ;LEAVE.

                        ;RESTORE R0-R5

.RES05:                 MOV     SAVR0,R0        ;RESTORE R0
                        MOV     SAVR1,R1        ;RESTORE R1
                        MOV     SAVR2,R2        ;RESTORE R2
                        MOV     SAVR3,R3        ;RESTORE R3
                        MOV     SAVR4,R4        ;RESTORE R4
                        MOV     SAVR5,R5        ;RESTORE R5
```

```
 949  004464  000002                          RTI                     ;LEAVE
 950
 951                                          ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
 952                                          ;-----------------------------------------------------
 953
 954  004466  104402  005672          .CONVR: TYPE    ,MCRLF
 955  004472  010046                  .CNVRT: MOV     R0,-(SP)
 956  004474  010146                          MOV     R1,-(SP)
 957  004476  010346                          MOV     R3,-(SP)
 958  004500  010446                          MOV     R4,-(SP)
 959  004502  010546                          MOV     R5,-(SP)
 960  004504  017601  000012                  MOV     @12(SP),R1
 961  004510  062766  000002  000012          ADD     #2,12(SP)
 962  004516  012137  004710                  MOV     (R1)+,WRDCNT
 963  004522  112137  004712          1$:     MOVB    (R1)+,CHRCNT
 964  004526  112137  004713                  MOVB    (R1)+,SPACNT
 965  004532  013137  004714                  MOV     @(R1)+,BINWRD
 966  004536  122737  000003  004712          CMPB    #3,CHRCNT
 967  004544  001003                          BNE     2$
 968  004546  042737  177400  004714          BIC     #177400,BINWRD
 969  004554  013704  004714          2$:     MOV     BINWRD,R4
 970  004560  113705  004712                  MOVB    CHRCNT,R5
 971  004564  012700  001416                  MOV     #TEMP,R0
 972  004570  010403                  3$:     MOV     R4,R3
 973  004572  042703  177770                  BIC     #177770,R3
 974  004576  062703  000060                  ADD     #060,R3
 975  004602  110320                          MOVB    R3,(R0)+
 976  004604  000241                          CLC
 977  004606  006004                          ROR     R4
 978  004610  000241                          CLC
 979  004612  006004                          ROR     R4
 980  004614  000241                          CLC
 981  004616  006004                          ROR     R4
 982  004620  005305                          DEC     R5
 983  004622  001362                          BNE     3$
 984  004624  012703  007544                  MOV     #MDATA,R3
 985  004630  114023                  4$:     MOVB    -(R0),(R3)+
 986  004632  105337  004712                  DECB    CHRCNT
 987  004636  001374                          BNE     4$
 988  004640  105737  004713                  TSTB    SPACNT
 989  004644  001405                          BEQ     6$
 990  004646  112723  000040          5$:     MOVB    #040,(R3)+
 991  004652  105337  004713                  DECB    SPACNT
 992  004656  001373                          BNE     5$
 993  004660  105013                  6$:     CLRB    (R3)
 994  004662  104402  007544                  TYPE    ,MDATA
 995  004666  005337  004710                  DEC     WRDCNT
 996  004672  001313                          BNE     1$
 997  004674  012605                          MOV     (SP)+,R5
 998  004676  012604                          MOV     (SP)+,R4
 999  004700  012603                          MOV     (SP)+,R3
1000  004702  012601                          MOV     (SP)+,R1
1001  004704  012600                          MOV     (SP)+,R0
1002  004706  000002                          RTI
1003  004710  000000          WRDCNT: 0
1004  004712  000000          CHRCNT: 0
```

```
1005            004713                          SPACNT=CHRCNT+1
1006  004714    000000                          BINWRD: 0
1007
1008
1009                                            ;TRAP DISPATCH SERVICE
1010                                            ;ARGUMENT OF TRAP IS EXTRACTED
1011                                            ;AND USED AS OFFSET TO OBTAIN POINTER
1012                                            ;TO SELECTED SUBROUTINE
1013
1014  004716    011646                  .TRPSR: MOV     (SP),-(SP)      ;GET PC OF RETURN
1015  004720    162716  000002                  SUB     #2,(SP)         ;=PC OF TRAP
1016  004724    017616  000000                  MOV     @(SP),(SP)      ;GET TRP
1017  004730    006316                  TRPOK:  ASL     (SP)            ;MULTIPLY TRAP ARG BY 2
1018  004732    042716  177001                  BIC     #177001,(SP)    ;CLEAR UNWANTED BITS
1019  004736    062716  001330                  ADD     #.TRPTAB,(SP)   ;POINTER TO SUBROUTINE ADDRESS
1020  004742    017616  000000                  MOV     @(SP),(SP)      ;SUBROUTINE ADDRESS
1021  004746    000136                          JMP     @(SP)+          ;GO TO SUBROUTINE
1022
1023                                            ;ERROR HANDLER
1024                                            ;-------------
1025
1026  004750    004737  007606          .HLT:   JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
1027  004754    032777  010000  174220          BIT     #SW12,@SWR      ;BELL ON ERROR?
1028  004762    001406                          BEQ     XBX             ;BR IF NO BELL
1029  004764    105777  174220                  TSTB    @TPCSR          ;TTY READY.
1030  004770    100003                          BPL     XBX             ;DON'T WAIT IF TTY NOT READY.
1031  004772    112777  000207  174212          MOVB    #207,@TPDBR     ;PUSH A BELL AT THE TTY.
1032  005000    032777  020000  174174  XBX:    BIT     #SW13,@SWR      ;DELETE ERROR PRINT OUT?
1033  005006    001105                          BNE     HALTS           ;BR IF NO PRINT OUT WANTED.
1034  005010    021637  001234                  CMP     (SP),LSTERR     ;WAS THIS ERROR FOUND LAST TIME?
1035  005014    001404                          BEQ     1$              ;BR IF YES
1036  005016    011637  001234                  MOV     (SP),LSTERR     ;RECORD BEING HERE
1037  005022    105037  001325                  CLRB    ERRFLG          ;PREPARE HEADER
1038  005026    104406                  1$:     SAVOS                   ;SAVE ALL PROC REGISTERS
1039  005030    011605                          MOV     (SP),R5         ;GET THE PC OF ERROR
1040  005032    162705  000002                  SUB     #2,R5           ;GET ADDRESS OF TRAP CALL
1041  005036    011504                          MOV     (R5),R4         ;GET HLT INSTRUCTION
1042  005040    006304                          ASL     R4              ;MULT BY TWO
1043  005042    061504                          ADD     (R5),R4         ;DOUBLE IT
1044  005044    006304                          ASL     R4              ;MULT AGAIN
1045  005046    042704  177001                  BIC     #177001,R4      ;CLEAR JUNK
1046  005052    062704  027602                  ADD     #.ERRTAB,R4     ;GET POINTER
1047  005056    012437  005172                  MOV     (R4)+,ERRMSG    ;GET ERROR MESSAGE
1048  005062    012437  005204                  MOV     (R4)+,DATAHD    ;GET DATA HEADRER
1049  005066    011437  005216                  MOV     (R4),DATABP     ;GET DATA TABLE
1050  005072    105737  001325                  TSTB    ERRFLG          ;TYPE HEADREER
1051  005076    001403                          BEQ     TYPMSG          ;BR IF YES
1052  005100    005737  005216                  TST     DATABP          ;DOES DATA TABLE EXIST?
1053  005104    001040                          BNE     TYPDAT          ;BR IF YES.
1054  005106    104402  005672  TYPMSG: TYPE    ,MCRLF
1055  005112    104402  005672          TYPE    ,MCRLF
1056  005116    005737  001220          TST     LOCK
1057  005122    001402                  BEQ     1$
1058  005124    104402  006142          TYPE    ,MASTEK
1059  005130    104402  006130  1$:     TYPE    ,MTSTN
1060  005134    104411  005330          CNVRT   ,XTSTN           ;SHOW IT
```

```
1061  005140  104402  006217                    TYPE    ,MERRPC         ;TYPE PC.
1062  005144  104411  005322                    CNVRT   ,ERTABO         ;SHOW IT
1063  005150  104402  005672                    TYPE    ,MCRLF          ;GIVE A CR/LF
1064  005154  112737  177777  001325            MOVB    #-1,ERRFLG      ;NO MORE HEADER UNLESS NO DATA TABLE.
1065  005162  005737  005172                    TST     ERRMSG          ;IS THERE AN ERROR MESSAGE?
1066  005166  001402                            BEQ     WRKO.FM         ;BR IF NO.
1067  005170  104402                            TYPE                    ;TYPE
1068  005172  000000                ERRMSG:     0                       ;    ERROR MESSAGE
1069  005174                         WRKO.FM:                            ;
1070  005174  005737  005204                    TST     DATAHD          ;DATA HEADER?
1071  005200  001402                            BEQ     TYPDAT          ;BR IF NO
1072  005202  104402                            TYPE                    ;TYPE
1073  005204  000000                DATAHD:     0                       ;    DATA HEADER
1074  005206  005737  005216        TYPDAT:     TST     DATABP          ;DATA TABLE?
1075  005212  001402                            BEQ     RESREG          ;BR IF NO.
1076  005214  104410                            CONVRT                  ;SHOW
1077  005216  000000                DATABP:     0                       ;    DATA TABLE
1078  005220  104407                RESREG:     RESCS                   ;RESTORE PROC REGISTERS
1079  005222  022737  003522  000042 HALTS:     CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, HALT!!
1080  005230  001403                            BEQ     1$
1081  005232  005777  173744                    TST     @SWR            ;HALT ON ERROR?
1082  005236  100005                            BPL     EXITER          ;BR IF NO HALT ON ERROR
1083  005240  010046                1$:         PUSHRO                  ;SAVE R0
1084  005242  016600  000002                    MOV     2(SP),R0        ;SHOW ERROR PC IN DATA LIGHTS
1085  005246  000000                            HALT                    ;HALT
1086  005250  012600                            POPRO                   ;GET R0
1087  005252  005237  001232        EXITER:     INC     ERRCNT          ;UPDATE ERROR COUNT
1088  005256  032777  000400  173716            BIT     #SW08,@SWR      ;GOTO TOP OF TEST?
1089  005264  001007                            BNE     1$              ;BR IF YES
1090  005266  032777  002000  173706            BIT     #SW10,@SWR      ;GOTO NEXT TEST?
1091  005274  001411                            BEQ     2$              ;BR IF NO
1092  005276  013737  001216  001214            MOV     NEXT,RETURN     ;SET FOR NEXT TEST
1093  005304  012706  001200        1$:         MOV     #STACK,SP       ;RESET SP
1094  005310  013701  001404                    MOV     DMCSR,R1        ;SET UP R1
1095  005314  000177  173674                    JMP     @RETURN         ;GOTO SPECIFIED TEST
1096  005320  000002                2$:         RTI                     ;RETURN
1097  005322  000001                ERTABO:     1
1098  005324     006     002                    .BYTE   6,2
1099  005326  001276                            SAVPC
1100  005330  000001                XTSTN:      1
1101  005332     003     002                    .BYTE   3,2
1102  005334  001226                            TSTNO
1103                                            ;ENTER HERE ON POWER FAILURE
1104                                            ;--------------------------
1105
1106
1107  005336                         .PFAIL:
1108  005336  012737  005350  000024            MOV     #RESTART,24             ;SET UP FOR POWER UP TRAP
1109  005344  000000                            HALT                            ;HALT ON POWER DOWN NORMAL
1110  005346  000777                            BR      .
1111
1112                                            ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
1113
1114  005350                         RESTAR:
1115  005350  012737  005336  000024            MOV     #.PFAIL,24      ;SET UP FOR POWER FAILURE
1116  005356  012706  001200                    MOV     #STACK,SP       ;RESET THE STACK POINTER
```

```
1117  005362  013701  001404                    MOV     DMCSR,R1        ;RESTORE R1
1118  005366  005037  001416                    CLR     TEMP            ;READY FOR TIMMER
1119  005372  005237  001416                    INC     TEMP            ;PLUS ONE TO THE TIMER!
1120  005376  001375                            BNE     .-4             ;BR IF MORE TO GO
1121  005400  104402  005675                    TYPE    ,MPFAIL         ;TYPE THE MESSAGE
1122  005404  104411  005430                    CNVRT   ,PFTAB          ;TELL WHAT TEST TO RETURN TO.
1123  005410  105037  001325                    CLRB    ERRFLG          ;START CLEAN
1124  005414  005037  001234                    CLR     LSTERR          ;
1125  005420  005011                            CLR     (R1)            ;CLEAR MAINT BITS
1126  005422  104412                            MSTCLR                  ;START CLEAN UP OF DEVICE
1127  005424  000177  173564                    JMP     @RETURN         ;START DOING THAT TEST AGAIN.
1128  005430  000001                    PFTAB:  1
1129  005432     003     002             .BYTE   3,2
1130  005434  001226                            TSTNO
1131
1132  005436                            .DELAY:
1133  005436  012777  000020  173746            MOV     #20,@DMP04
1134  005444  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1135  005446  121111                            121111                  ;POKE CLOCK DELAY BIT
1136  005450                            1$:
1137  005450  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1138  005452  121224                            121224                  ;PORT4←IBUS*11
1139  005454  032777  000020  173730            BIT     #BIT4,@DMP04    ;IS CLOCK BIT SET?
1140  005462  001772                            BEQ     1$              ;BR IF NO
1141  005464  000002                            RTI
1142
1143  005466                            .MSTCLR:
1144  005466  152777  000100  173712            BISB    #BIT6,@DMCSRH   ;SET MASTER CLEAR
1145  005474  142777  000300  173704            BICB    #BIT6!BIT7,@DMCSRH  ;CLEAR MASTER CLEAR AND RUN
1146  005502  000002                            RTI                     ;RETURN
1147
1148  005504                            .ROMCLK:
1149  005504  152777  000002  173674            BISB    #BIT1,@DMCSRH   ;SET ROMI
1150  005512  013677  173676                    MOV     @(SP)+,@DMP06   ;LOAD INSTRUCTION IN SEL6
1151  005516  062746  000002                    ADD     #2,-(SP)        ;ADJUST STACK
1152  005522  032777  000100  173452            BIT     #SW06,@SWR      ;HALT IF SW06 =1
1153  005530  001401                            BEQ     1$              ;BR IF SW06 =0
1154  005532  000000                            HALT                    ;HALT BEFORE CLOCKING INSTRUCTION
1155  005534  152777  000003  173644    1$:     BISB    #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
1156  005542  142777  000007  173636            BICB    #BIT2!BIT1!BIT0,@DMCSRH  ;CLEAR ROMO, ROMI, STEP
1157  005550  000002                            RTI
1158
1159  005552                            .DATACLK:
1160  005552  013637  001416                    MOV     @(SP)+,TEMP     ;PUT TICK COUNT IN TEMP
1161  005556  062746  000002                    ADD     #2,-(SP)        ;ADJUST STACK
1162  005562  152777  000020  173616    1$:     BISB    #BIT4,@DMCSRH   ;SET STEP LU
1163  005570  027777  173610  173606            CMP     @DMCSR,@DMCSR   ;WASTE TIME
1164  005576  142777  000020  173602            BICB    #BIT4,@DMCSRH   ;CLEAR STEP LU
1165  005604  005337  001416                    DEC     TEMP            ;DEC TICK COUNT
1166  005610  001364                            BNE     1$              ;BR IF NOT DONE
1167  005612  000002                            RTI                     ;RETURN
1168  005614  000001                    3$:     .BLKW 1
1169
1170  005616                            .TIMER:
1171  005616  013637  001416                    MOV     @(SP)+,TEMP     ;MOVE COUNT TO TEMP
1172  005622  062746  000002                    ADD     #2,-(SP)        ;ADJUST STACK
```

```
1173  005626
1174  005626  104414                          1$:
1175  005630  021364                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1176  005632  032777  000002  173552                021364                  ;PORT4←IBUS* REG11
1177  005640  001772                               BIT     #2,@DMP04       ;IS PGM CLOCK BIT CLEAR?
1178  005642                                        BEQ     1$              ;BR IF YES
1179  005642  104414                          2$:
1180  005644  021364                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1181  005646  032777  000002  173536                021364                  ;PORT4←IBUS* REG11
1182  005654  001372                               BIT     #2,@DMP04       ;IS PGM CLOCK BIT SET?
1183  005656  005337  001416                       BNE     2$              ;BR IF YES
1184  005662  001361                               DEC     TEMP            ;DEC COUNT
1185  005664  000002                               BNE     1$              ;BR IF NOT DONE
1186                                                RTI                     ;RETURN
1187  005666  020040  000077           MQM:    .ASCIZ  /  ?/
 (2)  005672  005015     000           MCRLF:  .ASCIZ  <15><12>
 (2)  005675     377  053520  020122   MPFAIL: .ASCIZ  <377>/PWR FAILED. RESTART AT TEST /
 (2)  005733     377  047105  020104   MEPASS: .ASCIZ  <377>/END PASS DZDMG  /
 (2)  005755     377  000122           MR:     .ASCIZ  <377>/R/
 (2)  005760  047377  020117  042504   MERR2:  .ASCIZ  <377>/NO DEVICES PRESENT./
 (2)  006005     377  047111  052523   MERR3:  .ASCIZ  <377>/INSUFFICIENT DATA!/
 (2)  006031     377  042524  052123   MTSTPC: .ASCIZ  <377>/TEST PC-/
 (2)  006043     377  047514  045503   MLOCK:  .ASCIZ  <377>/LOCK ON SELECTED TEST/
 (2)  006072  051503  035122  000040   MCSRX:  .ASCIZ   /CSR: /
 (2)  006100  042526  035103  000040   MVECX:  .ASCIZ  /VEC: /
 (2)  006106  040520  051523  051505   MPASSX: .ASCIZ  /PASSES: /
 (2)  006117     105  051122  051117   MERRX:  .ASCIZ  /ERRORS: /
 (2)  006130  042524  052123  047040   MTSTN:  .ASCIZ  /TEST NO: /
 (2)  006142  000052                   MASTEK: .ASCIZ  /*/
 (2)  006144  051777  052105  051440   MNEW:   .ASCIZ  <377>/SET SWITCH REG TO DMC11'S DESIRED ACTIVE./
 (2)  006217     120  035103  000040   MERRPC: .ASCIZ  /PC: /
 (2)  006224  020212  020040  020040   XHEAD:  .ASCII  <212>/           MAP OF DMC11 STATUS/
 (2)  006263     377  020040  020040           .ASCII  <377>/           --------------------/
 (2)  006322  020212  050040  020103           .ASCII  <212>/  PC      CSR     STAT1    STAT2    STAT3/
 (2)  006374  026777  026455  026455           .ASCIZ  <377>/------   ------   ------   ------   ------/
 (2)  006450  044377  053517  046440   NUM:    .ASCIZ  <377>/HOW MANY DMC11'S TO BE TESTED?/
 (2)  006510  041777  051123  040440   CSR:    .ASCIZ  <377>/CSR ADDRESS?/
 (2)  006526  053377  041505  047524   VEC:    .ASCIZ  <377>/VECTOR ADDRESS?/
 (2)  006547     377  051102  050040   PRIO:   .ASCIZ  <377>/BR PRIORITY LEVEL? (4,5,6,7)?/
 (2)  006606  044777  020106  046504   CRAM:   .ASCIZ  <377>/IF DMC HAS CRAM (M8204) TYPE "Y", IF CROM (M8200) TYPE "N"
 (2)  006704  053777  044510  044103   MODU:   .ASCIZ  <377>/WHICH LINE UNIT? IF NONE TYPE "N", IF M8201 TYPE "1", IF M
 (2)  007016  051777  044527  041524   LINE:   .ASCIZ  <377>/SWITCH PAC#1 (DDCMP LINE #)?/
 (2)  007054  051777  044527  041524   BM:     .ASCIZ  <377>/SWITCH PAC#2 (BM873 BOOT ADD)?/
 (2)  007114  044777  020123  044124   CONN:   .ASCIZ  <377>/IS THE LOOP BACK CONNECTOR ON?/
 (2)  007154  047377  020117  042504   NOACT:  .ASCIZ  <377>/NO DEVICES ARE SELECTED/
 (2)  007205     377  051412  051127   SWMES:  .ASCIZ  <377><12>/SWR= /
 (2)  007215     116  053505  020077   SWMES1: .ASCIZ  /NEW? /
 (2)  007223     377  042377  041515   CONERR: .ASCIZ  <377><377>/DMC11 FOUND AT NON-STANDARD ADDRESS  PC: /
 (2)  007277     377  054105  042520   CNERR:  .ASCIZ  <377>/EXPECTED  FOUND/
 (2)  007320  024040  046504  024503   DMCM:   .ASCIZ  / (DMC) /
 (2)  007330  024040  046513  024503   KMCM:   .ASCIZ  / (KMC) /
 (2)  007340  042377  041515  030461   SPEED:  .ASCIZ  <377>/DMC11-AR(REMOTE,LOW SPEED) OR DMC11-AL(LOCAL,HIGH SPEED) T
 (2)                                    .EVEN
 (2)  007454  000005                   XSTATQ: 5
1188  007456     006     003                   .BYTE   6,3
1189  007460  001246                           TEMP1
```

```
1190  007462  006  003                          .BYTE   6,3
1191  007464  001250                           TEMP2
1192  007466  006  003                          .BYTE   6,3
1193  007470  001252                           TEMP3
1194  007472  006  003                          .BYTE   6,3
1195  007474  001254                           TEMP4
1196  007476  006  002                          .BYTE   6,2
1197  007500  001256                           TEMP5
1198                                  .EVEN
1199
1200                                          ;BUFFERS FOR INPUT-OUTPUT
1201
1202  007502  000000                  INBUF:  0
1203          007544                  .=.+40
1204  007544  000000                  MDATA:  0
1205          007606                  .=.+40
1206
1207
1208                                          ;ROUTINE USED TO CHANGE SOFTWARE SWITCH
1209                                          ;REGISTER USING THE CONSOLE TERMINAL
1210                                          ;-------------------------------------
1211
1212  007606  022737  000176  001202  CKSWR:  CMP     #SWREG,SWR      ;IS THE SOFT SWR BEING USED?
1213  007614  001077                          BNE     CKSWR5          ;BR IF NO
1214  007616  105777  171362                  TSTB    @TKCSR          ;IS DONE SET?
1215  007622  100003                          BPL     2$              ;GO ON IF NOT SET
1216  007624  012737  177777  003734          MOV     #-1,DONE        ;IF DONE SET, SET FLAG
1217  007632  022777  000007  171346  2$:     CMP     #7,@TKDBR       ;WAS CTRL G TYPED? (7 BIT ASCII)
1218  007640  001404                          BEQ     1$              ;BR IF YES
1219  007642  022777  000207  171336          CMP     #207,@TKDBR     ;WAS CTRL G TYPED? (8 BIT ASCII)
1220  007650  001061                          BNE     CKSWR5          ;BR IF NO
1221  007652  010246                  1$:     MOV     R2,-(SP)        ;STORE R2
1222  007654  010346                          MOV     R3,-(SP)        ;STORE R3
1223  007656  010446                          MOV     R4,-(SP)        ;STORE R4
1224  007660  012737  177777  010016          MOV     #-1,SWFLG       ;SET SOFT TYPE OUT FLAG
1225  007666  005002          CKSWR1: CLR     R2                      ;CLEAR NEW SWR CONTENTS
1226  007670  012704  177777                  MOV     #-1,R4          ;SET FLAG TO ALL ONES
1227  007674  104402  007205                  TYPE    ,SWMES          ;TYPE "SWR= "
1228  007700  104411          CKSWR2: CNVRT                           ;TYPE OUT PRESENT CONTENTS
1229  007702  010052                          SOFTSW                  ;OF SOFT SWITCH REGISTER
1230  007704  104402  007215  CKSWR3: TYPE    ,SWMES1                 ;TYPE "NEW? "
1231  007710  004737  010020  CKSWR4: JSR     PC,INCHAR               ;GET RESPONSE
1232  007714  022703  000015                  CMP     #15,R3          ;WAS IT A CR?
1233  007720  001424                          BEQ     5$              ;BR IF YES
1234  007722  022703  000012                  CMP     #12,R3          ;WAS IT A LF?
1235  007726  001416                          BEQ     4$              ;BR IF YES
1236  007730  022703  000025                  CMP     #25,R3          ;WAS IT CTRL U?
1237  007734  001754                          BEQ     CKSWR1          ;BR IF YES(START OVER)
1238  007736  022703  000007                  CMP     #7,R3           ;IF CNTL G GET NEXT CHAR
1239  007742  001762                          BEQ     CKSWR4
1240  007744  005004                          CLR     R4              ;IT MUST BE A DIGIT SO CLR FLAG
1241  007746  042703  177770                  BIC     #177770,R3      ;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
1242  007752  006302                          ASL     R2              ;SHIFT R2 3 TIMES
1243  007754  006302                          ASL     R2
1244  007756  006302                          ASL     R2
1245  007760  050302                          BIS     R3,R2           ;ADD LAST DIGIT
```

```
1246  007762  000752                          BR      CKSWR4          ;GET NEXT CHARACTER
1247  007764  012766  002002  000006  4$:     MOV     #.START,6(SP)   ;LF WAS TYPED SO GO TO START
1248  007772  005704          5$:     TST     R4              ;IS FLAG CLEAR?
1249  007774  001002                  BNE     6$              ;IF NOT DON'T CHANGE SOFT SWR
1250  007776  010277  171200          MOV     R2,@SWR         ;IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
1251  010002  005037  010016  6$:     CLR     SWFLG           ;CLEAR TYPEOUT FLAG
1252  010006  012604                  MOV     (SP)+,R4        ;RESTORE R4
1253  010010  012603                  MOV     (SP)+,R3        ;RESTORE R3
1254  010012  012602                  MOV     (SP)+,R2        ;RESTORE R2
1255  010014  000207          CKSWR5: RTS     PC              ;RETURN
1256
1257  010016  000000          SWFLG:  0
1258
1259  010020  105777  171160  INCHAR: TSTB    @TKCSR
1260  010024  100375                  BPL     .-4
1261  010026  017703  171154          MOV     @TKDBR,R3
1262  010032  105777  171152          TSTB    @TPCSR
1263  010036  100375                  BPL     .-4
1264  010040  010377  171146          MOV     R3,@TPDBR
1265  010044  042703  000200          BIC     #BIT7,R3
1266  010050  000207                  RTS     PC
1267
1268  010052  000001          SOFTSW: 1
1269  010054     006     002          .BYTE   6,2
1270  010056  000176                  SWREG
```

```
1271
1272                                                    ;
1273                                                    ;ROUTINE USED TO "CYCLE" THROUGH UP TO 16 DMC11'S
1274                                                    ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
1275                                                    ;AND RUNS THE SPECIFIED DMC11'S.   THIS ROUTINE *MUST*
1276                                                    ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
1277                                                    ;SETUP NECESSARY.
1278                                                    ;
1279
1280  010060  005737  001306              CYCLE:  TST     DMACTV          ;ARE ANY DMC11'S TO BE TESTED?
1281  010064  001004                              BNE     1$              ;BR IF OK.
1282  010066  104402  007154                      TYPE    ,NOACT          ;NO DMC11'S SELECTED!!
1283  010072  000000                              HALT                    ;STOP THE SHOW.
1284  010074  000776                              BR      .-2             ;DISQUALIFY CONT. SW.
1285  010076  000241                      1$:     CLC                     ;CLEAR PROC. CARRY BIT.
1286  010100  006137  001316                      ROL     RUN             ;UPDATE POINTER
1287  010104  005537  001316                      ADC     RUN             ;CATCH CARRY FROM RUN
1288  010110  062737  000004  001322              ADD     #4,MILK         ;UPDATE POINTER
1289  010116  062737  000010  001320              ADD     #10,CREAM       ;UPDATE ADDRESS POINTER.
1290  010124  022737  001700  001320              CMP     #DM.MAP+200,CREAM
1291  010132  001006                              BNE     2$              ;KEEP GOING; NOT ALL TESTED FOR.
1292  010134  012737  001500  001320              MOV     #DM.MAP,CREAM   ;RESET ADDRESS POINTER.
1293  010142  012737  001702  001322              MOV     #CNT.MAP,MILK   ;RESET PASS COUNT POINTER
1294  010150  033737  001316  001306      2$:     BIT     RUN,DMACTV      ;IS THIS ONE ACTIVE?
1295  010156  001747                              BEQ     1$              ;BR IF NO
1296  010160  013700  001320                      MOV     CREAM,R0        ;GET ADDRESS POINTER
1297  010164  013702  001322                      MOV     MILK,R2         ;GET PASS COUNT POINTER
1298  010170  012037  001404                      MOV     (R0)+,DMCSR     ;LOAD SYSTEM CTRL. REG
1299  010174  011037  001374                      MOV     (R0),DMRVEC     ;LOAD VECTOR
1300  010200  042737  177000  001374              BIC     #177000,DMRVEC  ;CLEAR UNWANTED BITS
1301  010206  012037  001366                      MOV     (R0)+,STAT1     ;LOAD STAT1
1302  010212  012037  001370                      MOV     (R0)+,STAT2     ;LOAD STAT2
1303  010216  012037  001372                      MOV     (R0)+,STAT3     ;LOAD STAT3
1304  010222  012237  001230                      MOV     (R2)+,PASCNT    ;LOAD PASS COUNT
1305  010226  012237  001232                      MOV     (R2)+,ERRCNT    ;LOAD ERROR COUNT
1306  010232  012700  000002                      MOV     #2,R0           ;SAVE CORE THIS WAY!
1307  010236  013737  001404  001406              MOV     DMCSR,DMCSRH
1308  010244  005237  001406                      INC     DMCSRH
1309  010250  013737  001406  001410              MOV     DMCSRH,DMCTL
1310  010256  005237  001410                      INC     DMCTL
1311  010262  013737  001410  001412              MOV     DMCTL,DMP04
1312  010270  060037  001412                      ADD     R0,DMP04
1313  010274  013737  001412  001414              MOV     DMP04,DMP06
1314  010302  060037  001414                      ADD     R0,DMP06
1315
1316  010306  013737  001374  001376              MOV     DMRVEC,DMRLVL   ;PTY LVL
1317  010314  060037  001376                      ADD     R0,DMRLVL       ;
1318  010320  013737  001376  001400              MOV     DMRLVL,DMTVEC   ;TX VEC
1319  010326  060037  001400                      ADD     R0,DMTVEC       ;
1320  010332  013737  001400  001402              MOV     DMTVEC,DMTLVL   ;TX LVL
1321  010340  060037  001402                      ADD     R0,DMTLVL
1322
1323  010344  032737  000002  001236              BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED
1324  010352  001450                              BEQ     7$              ;BR IF NO
1325  010354                              4$:
1326  010354  005737  000042                      TST     @#42            ;RUNNING IN AUTO MODE?
```

```
1327  010360  001045                        BNE     7$              ;BR IF YES
1328  010362  104402  005672                TYPE    ,MCRLF
1329  010366  104403                        INSTR                   ;GET TEST NO.
1330  010370  006130                        MTSTN
1331  010372  104405                        PARAM
1332  010374  000001                        1
1333  010376  001000                        1000
1334  010400  001226                        TSTNO
1335  010402     000              .BYTE   0
1336  010403     001              .BYTE   1
1337  010404  012700  022322                MOV     #TST1,R0
1338  010410  022710                5$:     CMP     (PC)+,(R0)      ;CMP FIRST WORD TO 12737
1339  010412  012737                        MOV     (PC)+,@(PC)+
1340  010414  001020                        BNE     6$              ;BR IF NOT SAME
1341  010416  023760  001226  000002        CMP     TSTNO,2(R0)     ;DOES TSTNO MATCH?
1342  010424  001014                        BNE     6$              ;BR IF NO
1343  010426  022760  001226  000004        CMP     #TSTNO,4(R0)    ;IS LAST WORD OK?
1344  010434  001010                        BNE     6$              ;BR IF NO
1345  010436  010037  001214                MOV     R0,RETURN       ;IT IS A LEGAL TEST SO DO IT
1346  010442  104402  005755                TYPE    ,MR
1347  010446  042737  000002  001236        BIC     #SW01,STRTSW
1348  010454  000412                        BR      8$
1349  010456  005720                6$:     TST     (R0)+           ;POP R0
1350  010460  020027  026472                CMP     R0,#TLAST+10    ;AT END YET?
1351  010464  001351                        BNE     5$              ;BR IF NO
1352  010466  104402  005666                TYPE    ,MQM            ;YES ILLEGAL TEST NO.
1353  010472  000730                        BR      4$              ;TRY AGAIN
1354
1355  010474  012737  022322  001214  7$:     MOV     #TST1,RETURN    ;PREPARE RETURN ADDRESS
1356  010502  013701  001404        8$:     MOV     DMCSR,R1        ;R1 = BASE DMC11 ADDRESS
1357  010506  000177  170502                JMP     @RETURN         ;GO START TESTING.
1358
1359
1360                                    ;ROUTINE USED TO "AUTO SIZE" THE DMC11
1361                                    ;CSR AND VECTOR.
1362                                    ;NOTE:   THE CSR MAY BE ANY WHERE IN THE FLOATING
1363                                    ;        ADDRESS RANGE (160000:164000)
1364                                    ;        AND THE VECTOR MAY BE ANY WHERE IN THE
1365                                    ;        FLOATING VECTOR RANGE (300:770)
1366                                    ;
1367
1368  010512                    AUTO.SIZE:
1369  010512  000005                        RESET                   ;INSURE A BUS INIT.
1370  010514  012702  001500        CSRMAP: MOV     #DM.MAP,R2      ;LOAD MAP POINTER.
1371  010520  005022                1$:     CLR     (R2)+           ;ZERO ENTIRE MAP
1372  010522  022702  001700                CMP     #DM.END,R2      ;ALL DONE?
1373  010526  001374                        BNE     1$              ;BR IF NO
1374  010530  005037  001310                CLR     DMNUM           ;SET OCTAL NUMBER OF DMC11'S TO 0
1375  010534  012702  001500                MOV     #DM.MAP,R2      ;R2 POINTS TO DMC MAP
1376  010540  005037  001306                CLR     DMACTV          ;CLEAR ACTIVE
1377  010544  032737  000001  001236        BIT     #SW00,STRTSW    ;QUESTIONS?
1378  010552  001002                        BNE     .+6             ;BR IF YES
1379  010554  000137  011252                JMP     7$              ;IF NO SKIP QUESTIONS
1380  010560  012737  000001  001256        MOV     #1,TEMP5        ;START WITH 1
1381  010566  104403                        INSTR
1382  010570  006450                        NUM
```

```
1383  010572  104405                                  PARAM
1384  010574  000001                                  1
1385  010576  000020                                  16.
1386  010600  001252                                  TEMP3
1387  010602     000                                  .BYTE   0
1388  010603     001                                  .BYTE   1
1389  010604  013737  001252  001310                  MOV     TEMP3,DMNUM     ;DMNUM = HOW MANY
1390  010612  104402  005672                  12$:    TYPE    ,MCRLF
1391  010616  104410                                  CONVRT                  ;TYPE WHICH DMC IS BEING DONE
1392  010620  012002                                  WHICH                   ;TEMP5 IS WHICH DMC
1393  010622  005237  001256                          INC     TEMP5
1394  010626  104403                                  INSTR
1395  010630  006510                                  CSR
1396  010632  104405                                  PARAM
1397  010634  160000                                  160000
1398  010636  164000                                  164000
1399  010640  001254                                  TEMP4
1400  010642     000                                  .BYTE   0
1401  010643     001                                  .BYTE   1
1402  010644  013722  001254                          MOV     TEMP4,(R2)+     ;STORE CSR IN MAP
1403  010650  104403                                  INSTR
1404  010652  006526                                  VEC
1405  010654  104405                                  PARAM
1406  010656  000000                                  0
1407  010660  000776                                  776
1408  010662  001254                                  TEMP4
1409  010664     000                                  .BYTE   0
1410  010665     001                                  .BYTE   1
1411  010666  013712  001254                          MOV     TEMP4,(R2)      ;STORE VECTOR IN MAP
1412  010672  104402                          10$:    TYPE
1413  010674  006547                                  PRIO                    ;ASK WHAT BR LEVEL
1414  010676  004737  012266                          JSR     PC,INTTY        ;GET RESPONSE
1415  010702  022703  000024                          CMP     #24,R3          ;
1416  010706  101014                                  BHI     50$             ;BR IF LESS THAN 4
1417  010710  022703  000027                          CMP     #27,R3          ;
1418  010714  103411                                  BLO     50$             ;BR IF GREATER THAN 7
1419  010716  012704  000011                          MOV     #11,R4          ;R4 = NUMBER OF SHIFTS
1420  010722  006303                                  ASL     R3              ;SHIFT R3 LEFT
1421  010724  005304                                  DEC     R4              ;DEC SHIFT COUNT
1422  010726  001375                                  BNE     .-4             ;BR IF NOT DONE
1423  010730  042703  170777                          BIC     #170777,R3      ;BIC UNWANTED BITS
1424  010734  050312                                  BIS     R3,(R2)         ;PUT BR LEVEL IN STATUS MAP
1425  010736  000403                                  BR      8$              ;CONTINUE
1426  010740  104402                          50$:    TYPE
1427  010742  005666                                  MQM                     ;RESPONSE IS OUT OF LIMITS
1428  010744  000752                                  BR      10$             ;TRY AGAIN
1429  010746  104402                          8$:     TYPE
1430  010750  006606                                  CRAM                    ;DOES DMC HAVE CRAM?
1431  010752  004737  012266                          JSR     PC,INTTY        ;GET REPLY
1432  010756  022703  000131                          CMP     #131,R3
1433  010762  001427                                  BEQ     9$              ;YES
1434  010764  022703  000116                          CMP     #116,R3         ;NO
1435  010770  001403                                  BEQ     40$             ;NOT A Y OR N
1436  010772  104402                                  TYPE
1437  010774  005666                                  MQM                     ;TYPE "?"
1438  010776  000763                                  BR      8$              ;ASK AGAIN
```

```
1439  011000  104402                        40$:    TYPE
1440  011002  007340                                SPEED                   ;DMC11-AR OR DMC11-AL?
1441  011004  004737  012266                        JSR     PC,INTTY        ;GET RESPONSE
1442  011010  022703  000122                        CMP     #122,R3         ;IS IT R
1443  011014  001414                                BEQ     16$             ;BR IF REMOTE
1444  011016  022703  000114                        CMP     #114,R3         ;IS IT L
1445  011022  001403                                BEQ     41$             ;BR IF LOCAL
1446  011024  104402                                TYPE
1447  011026  005666                                MQM
1448  011030  000763                                BR      40$             ;TRY AGAIN
1449  011032  052762  000002  000004        41$:    BIS     #BIT1,4(R2)     ;SET BIT1 IN STAT3
1450  011040  000402                                BR      16$             ;CONTINUE
1451  011042  052712  100000                9$:     BIS     #BIT15,(R2)     ;SET BIT 15 IF CRAM
1452  011046  104402                        16$:    TYPE
1453  011050  006704                                MODU                    ;ASK WHICH LINE UNIT
1454  011052  004737  012266                        JSR     PC,INTTY        ;GET REPLY
1455  011056  022703  000021                        CMP     #21,R3          ;"1"
1456  011062  001417                                BEQ     30$
1457  011064  022703  000022                        CMP     #22,R3          ;"2"
1458  011070  001412                                BEQ     31$
1459  011072  022703  000116                        CMP     #116,R3         ;"N"
1460  011076  001403                                BEQ     32$
1461  011100  104402                                TYPE
1462  011102  005666                                MQM                     ;IF NOT A 1,2 OR N TYPE "?"
1463  011104  000760                                BR      16$             ;TRY AGIAN
1464  011106  052722  010000                32$:    BIS     #BIT12,(R2)+    ;SET BIT 12 IN STAT2 IF NO LU
1465  011112  022222                                CMP     (R2)+,(R2)+     ;POP OVER STAT2 AND STAT3
1466  011114  000447                                BR      33$
1467  011116  052712  020000                31$:    BIS     #BIT13,(R2)     ;SET BIT 13 IN STAT2 IF M8202
1468  011122  104402                        30$:    TYPE
1469  011124  007114                                CONN                    ;ASK IF LOOP-BACK IS ON
1470  011126  004737  012266                        JSR     PC,INTTY        ;GET REPLY
1471  011132  022703  000131                        CMP     #131,R3         ;Y
1472  011136  001406                                BEQ     17$
1473  011140  022703  000116                        CMP     #116,R3         ;N
1474  011144  001406                                BEQ     18$
1475  011146  104402                                TYPE
1476  011150  005666                                MQM                     ;IF NOT Y OR N TYPE "?"
1477  011152  000763                                BR      30$             ;TRY AGAIN
1478  011154  052722  040000                17$:    BIS     #BIT14,(R2)+    ;TURNAROUND IS CONNECTED
1479  011160  000402                                BR      19$
1480  011162  042722  040000                18$:    BIC     #BIT14,(R2)+    ;NO TURNAROUND
1481  011166                                19$:
1482  011166  104403                                INSTR
1483  011170  007016                                LINE
1484  011172  104405                                PARAM
1485  011174  000000                                0
1486  011176  000377                                377
1487  011200  001254                                TEMP4
1488  011202     000                                .BYTE   0
1489  011203     001                                .BYTE   1
1490  011204  113722  001254                        MOVB    TEMP4,(R2)+     ;STORE SWITCH PAC IN MAP
1491  011210  104403                                INSTR
1492  011212  007054                                BM
1493  011214  104405                                PARAM
1494  011216  000000                                0
```

```
1495  011220  000377                                    377
1496  011222  001254                                    TEMP4
1497  011224     000                                    .BYTE   0
1498  011225     001                                    .BYTE   1
1499  011226  113722  001254                           MOVB    TEMP4,(R2)+     ;STORE SWITCH PAC IN MAP
1500  011232  005722                                    TST     (R2)+           ;POP OVER STAT3
1501  011234  005337  001252                 33$:      DEC     TEMP3           ;DEC DMC COUNT
1502  011240  001402                                    BEQ     34$             ;BR IF DONE
1503  011242  000137  010612                           JMP     12$             ;JUMP IF NOT
1504  011246  000137  011702                 34$:      JMP     13$             ;CONTINUE
1505  011252  012701  160000                 7$:       MOV     #160000,R1      ;SET FOR FIRST ADDRESS TO BE TESTED
1506  011256  012737  011774  000004                   MOV     #6$,@#4         ;SET FOR NON-EXISTANT DEVICE TIME OUT
1507  011264  005011                          2$:       CLR     (R1)            ;CLEAR SEL0
1508  011266  005711                                    TST     (R1)            ;IF DMC11 DMCSR S/B 0
1509  011270  001172                                    BNE     3$              ;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO DMC1
1510  011272  005061  000006                           CLR     6(R1)           ;CLEAR SEL6
1511  011276  005761  000006                           TST     6(R1)           ;IF DMC11 THEN DMRIC S/B =0!
1512  011302  001165                                    BNE     3$              ;BR IF NOT DMC11
1513  011304  012711  002000                           MOV     #BIT10,(R1)     ;SET ROM0
1514  011310  005061  000004                           CLR     4(R1)           ;CLEAR SEL4
1515  011314  012761  125252  000006                   MOV     #125252,6(R1)   ;WRITE THIS TO SEL6
1516  011322  052711  020000                           BIS     #BIT13,(R1)     ;WRITE IT!
1517  011326  022761  125252  000004                   CMP     #125252,4(R1)   ;WAS IT WRITTEN?
1518  011334  001004                                    BNE     21$             ;IF NO IT IS NOT CRAM
1519  011336  052762  100000  000002                   BIS     #BIT15,2(R2)    ;SET BIT15 IF CRAM
1520  011344  000431                                    BR      22$
1521  011346  012711  001000                 21$:      MOV     #BIT9,(R1)      ;SET ROMI
1522  011352  012761  100417  000006                   MOV     #100417,6(R1)   ;PUT INSTRUCTION IN SEL6
1523  011360  012711  001400                           MOV     #BIT9!BIT8,(R1) ;CLOCK INSTRUCTION (MICRO PROC PC TO 0)
1524  011364  012711  002000                           MOV     #BIT10,(R1)     ;SET ROM0
1525  011370  022761  000626  000006                   CMP     #626,6(R1)      ;IS IT LOCAL CROM
1526  011376  001411                                    BEQ     23$             ;BR IF YES
1527  011400  022761  016520  000006                   CMP     #16520,6(R1)    ;IS IT REMOTE CROM?
1528  011406  001410                                    BEQ     22$             ;BR IF YES
1529  011410  022761  177777  000006                   CMP     #-1,6(R1)       ;NO CROM?
1530  011416  001404                                    BEQ     22$             ;BR IF YES
1531  011420  000516                                    BR      3$              ;NOT A DMC
1532  011422  052762  000002  000006          23$:      BIS     #BIT1,6(R2)     ;SET BIT 1 IN STAT3
1533                                           ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DMC11 CSR ADDRESS.
1534  011430  010122                          22$:      MOV     R1,(R2)+        ;STORE CSR IN CORE TABLE.
1535  011432  012711  001000                 15$:      MOV     #BIT9,(R1)      ;CLEAR LINE UNIT LOOP
1536  011436  005061  000004                           CLR     4(R1)           ;CLEAR PORT4
1537  011442  012761  122113  000006                   MOV     #122113,6(R1)   ;LOAD INSTRUCTION (CLR DTR)
1538  011450  052711  000400                           BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
1539  011454  012761  021264  000006                   MOV     #021264,6(R1)   ;LOAD INSTRUCTION
1540  011462  052711  000400                           BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
1541  011466  122761  000377  000004                   CMPB    #377,4(R1)      ;IS IT ALL ONES?
1542  011474  001003                                    BNE     .+10            ;BR IF NO
1543  011476  052712  010000                           BIS     #BIT12,(R2)     ;IF YES, NO LINE UNIT, SET STATUS BIT
1544  011502  000436                                    BR      20$
1545  011504  032761  000002  000004                   BIT     #BIT1,4(R1)     ;IS SWITCH A ONE?
1546  011512  001403                                    BEQ     .+10            ;BR IF M8201
1547  011514  052712  060000                           BIS     #BIT13!BIT14,(R2) ;M8202 ASSUME CONNECTOR
1548  011520  000427                                    BR      20$             ;CONNECTOR ON)
1549  011522  032761  000010  000004                   BIT     #BIT3,4(R1)     ;IS MRDY SET
1550  011530  001023                                    BNE     20$             ;BR IF M8201 NO CONNECTOR (ON LINE)
```

```
1551  011532  012761  000100  000004          MOV     #BIT6,4(R1)     ;LOAD PORT4
1552  011540  012761  122113  000006          MOV     #122113,6(R1)   ;LOAD INSTRUCTION
1553  011546  052711  000400                  BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(SET DTR)
1554  011552  012761  021264  000006          MOV     #021264,6(R1)   ;LOAD INSTRUCTION
1555  011560  052711  000400                  BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(READ MODEM REG)
1556  011564  032761  000010  000004          BIT     #BIT3,4(R1)     ;IS MRDY SET NOW?
1557  011572  001402                          BEQ     20$             ;BR IF NO CONNECTOR
1558  011574  052712  040000                  BIS     #BIT14,(R2)     ;SET STATUS BIT FOR CONNECTOR
1559  011600  005722                  20$:    TST     (R2)+           ;POP POINTER
1560  011602  012761  021324  000006          MOV     #021324,6(R1)   ;PUT INSTRUCTION IN PORT6
1561  011610  012711  001400                  MOV     #BIT9!BIT8,(R1) ;PORT4←LU 15
1562  011614  156122  000004                  BISB    4(R1),(R2)+     ;STORE DDCMP LINE # IN TABLE
1563  011620  012761  021344  000006          MOV     #021344,6(R1)   ;PORT6←INSTRUCTION
1564  011626  012711  001400                  MOV     #BIT8!BIT9,(R1) ;CLOCK INSTR.
1565  011632  156122  000004                  BISB    4(R1),(R2)+     ;STORE BM873 ADD IN TABLE
1566  011636  005722                          TST     (R2)+           ;POP OVER STAT3
1567  011640  005011                          CLR     (R1)            ;CLEAR ROMI
1568  011642  005237  001310                  INC     DMNUM           ;UPDATE DEVICE COUNTER
1569  011646  022737  000020  001310          CMP     #20,DMNUM       ;ARE MAX. NO. OF DEV FOUND?
1570  011654  001412                          BEQ     13$             ;YES DON'T LOOK FOR ANY MORE.
1571  011656  005011                  3$:     CLR     (R1)            ;CLEAR BIT 10
1572  011660  005061  000006                  CLR     6(R1)           ;CLEAR SEL 6
1573  011664  062701  000010          14$:    ADD     #10,R1          ;UPDATE CSR POINTER ADDRESS
1574  011670  022701  164000                  CMP     #164000,R1
1575  011674  001402                          BEQ     13$             ;BR IF DONE
1576  011676  000137  011264                  JMP     2$              ;JUMP IF NOT
1577  011702  005037  001306          13$:    CLR     DMACTV
1578  011706  005737  001310                  TST     DMNUM           ;WERE ANY DMC11'S FOUND AT ALL?
1579  011712  001423                          BEQ     5$              ;ERROR AUTO SIZER FOUND NO DMC11'S IN THIS SYS.
1580  011714  013701  001310                  MOV     DMNUM,R1
1581  011720  010137  001314                  MOV     R1,SAVNUM       ;SAVE NUMBER OF DEVICES
1582  011724  000241                  4$:     CLC
1583  011726  006137  001306                  ROL     DMACTV          ;GENERATE ACTIVE REGISTER OF DEVICES.
1584  011732  005237  001306                  INC     DMACTV          ;SET THE BIT
1585  011736  005301                          DEC     R1
1586  011740  001371                          BNE     4$              ;BR IF MORE TO GENERATE
1587  011742  012737  000006  000004          MOV     #6,@#4          ;RESTORE TRAP VECTOR
1588  011750  013737  001306  001312          MOV     DMACTV,SAVACT   ;SAVE ACTIVE REGISTER
1589  011756  000137  012010                  JMP     VECMAP          ;GO FIND THE VECTOR NOW.
1590  011762  104402  005760          5$:     TYPE    ,MERR2          ;NOTIFY OPR THAT NO DMC11'S FOUND.
1591  011766  005000                          CLR     R0              ;MAKE DATA LIGHTS ZERO
1592  011770  000000                          HALT                    ;STOP THE SHOW
1593  011772  000776                          BR      .-2             ;DISABLE CONT. SW.
1594  011774  012716  011664          6$:     MOV     #14$,(SP)       ;ENTERED BY NON-EXISTANT TIME-OUT.
1595  012000  000002                          RTI                     ;RETURN TO MAINSTREAM
1596
1597  012002  000001                  WHICH:  1
1598  012004     002     002                  .BYTE   2,2
1599  012006  001256                          TEMP5
1600
1601  012010  032737  000001  001236  VECMAP: BIT     #SW00,STRTSW
1602  012016  001114                          BNE     5$
1603  012020  012737  000340  000022          MOV     #340,@#22       ;SET IOT TRAP PRIO TO 7
1604  012026  012737  012202  000020          MOV     #4$,@#20        ;SET IOT TRAP VECTOR
1605  012034  012702  001500                  MOV     #DM.MAP,R2      ;SET SOFTWARE POINTER
1606  012040  012700  000300                  MOV     #300,R0         ;FLOATING VECTORS START HERE.
```

```
1607  012044  012701  000302                    MOV     #302,R1          ;PC OF IOT INSTR.
1608  012050  010120                    1$:     MOV     R1,(R0)+         ;START FILLING VECTOR AREA
1609  012052  012721  000004                    MOV     #4,(R1)+         ;WITH .+2; IOT
1610  012056  022021                            CMP     (R0)+,(R1)+      ;ADD 2 TO R0 +R1
1611  012060  020127  001000                    CMP     R1,#1000
1612  012064  101771                            BLOS    1$               ;BR IF MORE TO FILL
1613  012066  013737  001306  001246            MOV     DMACTV,TEMP1     ;STORE TEMPORALLY
1614  012074  006037  001246            2$:     ROR     TEMP1            ;BRING OUT A BIT
1615  012100  103063                            BCC     5$               ;BR IF ALL DONE
1616  012102  012704  000012                    MOV     #12,R4           ;R4 IS INDEX REGISTER
1617  012106  016437  012252  177776            MOV     BRLVL(R4),PS     ;SET PS TO 7
1618  012114  011201                            MOV     (R2),R1
1619  012116  012761  000200  000004            MOV     #200,4(R1)
1620  012124  012711  001000                    MOV     #BIT9,(R1)       ;SET ROMI
1621  012130  012761  121111  000006            MOV     #121111,6(R1)    ;PUT INSTRUCTION IN PORT6
1622  012136  012711  001400                    MOV     #BIT9!BIT8,(R1)  ;FORCE AN INTERRUPT
1623  012142  105200                    7$:     INCB    R0               ;STALL
1624  012144  001376                            BNE     .-2              ;FOR TIME TO INTERUPT
1625  012146  162704  000002                    SUB     #2,R4            ;GET NEXT LOWEST PS LEVEL
1626  012152  001404                            BEQ     6$               ;BR IF R4 = 0
1627  012154  016437  012252  177776            MOV     BRLVL(R4),PS     ;MOVE NEXT LOWER LEVEL IN PS
1628  012162  000767                            BR      7$               ;BR TO DELAY
1629  012164  052762  005300  000002    6$:     BIS     #5300,2(R2)      ;NO INTERUPT ASSUME 300 AT LEVEL 5 AND FIX DMC11
1630  012172  005011                    3$:     CLR     (R1)             ;CLEAR ROMI
1631  012174  062702  000010                    ADD     #10,R2           ;POP SOFTWARE POINTER
1632  012200  000735                            BR      2$               ;KEEP GOING
1633  012202  051662  000002            4$:     BIS     (SP),2(R2)       ;GET VECTOR ADDRESS
1634  012206  042762  000007  000002            BIC     #7,2(R2)         ;CLEAR JUNK
1635  012214  016405  012254                    MOV     BRLVL+2(R4),R5   ;GET BR LEVEL OF DMC11
1636  012220  006305                            ASL     R5               ;SHIFT LEVEL 4 PLACES
1637  012222  006305                            ASL     R5               ;TO THE LEFT FOR THE
1638  012224  006305                            ASL     R5               ;STATUS TABLE
1639  012226  006305                            ASL     R5
1640  012230  042705  170777                    BIC     #170777,R5       ;CLEAR UNWANTED BITS
1641  012234  050562  000002                    BIS     R5,2(R2)         ;PUT BR LEVEL IN STATUS TABLE
1642  012240  022626                            CMP     (SP)+,(SP)+      ;POP IOT JUNK OFF STACK
1643  012242  012716  012172                    MOV     #3$,(SP)         ;SET FOR RETURN
1644  012246  000002                            RTI
1645  012250  000207                    5$:     RTS     PC               ;ALL DONE WITH "AUTO SIZING"
1646
1647  012252  000000                    BRLVL:  0       ;LEVEL 0
1648  012254  000000                            0       ;LEVEL 0
1649  012256  000200                            200     ;LEVEL 4
1650  012260  000240                            240     ;LEVEL 5
1651  012262  000300                            300     ;LEVEL 6
1652  012264  000340                            340     ;LEVEL 7
1653
1654
1655  012266  105777  166712            INTTY:  TSTB    @TKCSR           ;WAIT FOR DONE
1656  012272  100375                            BPL     .-4
1657  012274  017703  166706                    MOV     @TKDBR,R3        ;PUT CHAR IN R3
1658  012300  105777  166704                    TSTB    @TPCSR           ;WAIT UNTIL PRINTER IS READY
1659  012304  100375                            BPL     .-4
1660  012306  010377  166700                    MOV     R3,@TPDBR        ;ECHO CHAR
1661  012312  042703  000240                    BIC     #BIT7!BIT5,R3    ;MASK OFF LOWER CASE
1662  012316  000207                            RTS     PC               ;RETURN
```

```
1663
1664                                    01000
1665  012320  000000                    01100   ROMMAP: 0                       ;POINTER TO HI OR LO SPEED MICRO-CODE
1666                                    01200
1667  012322                            01300   LOMAP:                          ;LOW SPEED (REMOTE) MICRO-CODE
```

```
                        .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
004000          PAGE1=4000
010000          PAGE2=10000
014000          PAGE3=14000
001000  CCOND=1000             ;CONDITION C
001400  ZCOND=1400             ;CONDITION Z
000400  ALCOND=400             ;ALWAYS
002000  BRDCON=2000            ;CONDITION BRD
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
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                        .SBTTL          INDEXED BRANCH INSTRUCTIONS
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                        .SBTTL          MOVE INSTRUCTIONS
                ;
        000000  MOVE=0          ;MOVE OPCODE
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
```

```
                        .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                  ORCVCO=12               ;RCVR CONTROL
000013                  OMODEM=13               ;MODEM CONTROL
000014                  SYNC=14                 ;SYN REGISTER
000017                  OLUMAN=17               ;LINE UNIT MAINT.
```

```
                        .SBTTL PROTOCOL DEPENDANT MACROS
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;

177777                  MICPC=177777    ;INIT MICRO PC
```

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

353
354             000000
355             000000
```

```
                        .TITLE DMC11 DDCMP PROTOCOL IMPLEMENTATION
                        .IDENT  /V0001/
                        .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          OCTOBER 13, 1976
                        .SBTTL  THIS VERSION WAS USED TO BLAST THE FIRST
                        .SBTTL  RELEASE ON OCTOBER 13, 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                  LRC=ERC+1               ;LAST POINTER RECD
000025                  RCL1=LRC+1              ;RECEIVE LINK #1
000032                  RCL2=RCL1+5             ;   "       "   #2
000037                  RCL3=RCL2+5             ;   "       "   #3
000044                  RCL4=RCL3+5
000051                  RCL5=RCL4+5
000056                  RCL6=RCL5+5
000063                  RCL7=RCL6+5
000070                  STC=RCL7+5              ;START OF OUTPUT CHAIN---NEXT TMT DONE
000071                  LTC=STC+1               ;LAST TRANSMITTED
000072                  ETC=LTC+1               ;END OF TRANSMIT CHAIN
000073                  TML1=ETC+1              ;TRANSMIT LINK #1
000101                  TML2=TML1+6             ;    "        "   #2
000107                  TML3=TML2+6             ;    "        "    #3
000115                  TML4=TML3+6
000123                  TML5=TML4+6
000131                  TML6=TML5+6
000137                  TML7=TML6+6
000145                  TML8=TML7+6
000153                  T=TML8+6                ;TYPE FIELD
000154                  ST=T+1          ;SUBTYPE FIELD
000155                  ISP17=ST+1      ;MSG ACKED IMAGE
000156                  IMG10=ISP17+1           ;IMAGE OF BIT 1 OF SP10
000157                  IMG11=IMG10+1           ;IMAGE OF SP11
000160                  IMG12=IMG11+1           ;IMAGE OF SP12
000161                  IMG17=IMG12+1           ;IMAGE OF SP17
000162                  PRTST=IMG17+1           ;PORT STATE
000163                  TYPTAB=PRTST+1  ;TYPE TABLE---
                                        ;72 TYPE TABLE REP
                                        ;73  "      "  NAK
000165                  TYPSTT=TYPTAB+2 ;74  "      "  START
                                        ;75  "      "  STACK
                                        ;
                                        ;
000170                  OSTATE=TYPSTT+3 ;OLD STATE POINTER
000171                  ISP11=OSTATE+1  ;SP11 IMAGE
000172                  ISP12=ISP11+1   ;SP12 IMAGE
000173                  INCONS=ISP12+1  ;IN CONTROL CSR IMAGE
```

```
000174                  RTHRS=INCONS+1  ;RECV THRESHOLD LINK
000240                  NXTINT=240      ;NEXT INTERRUPT POSITION
000241                  NXTSP=NXTINT+1  ;END OF INTERUPT CHAIN
000242                  INTSTK=NXTSP+1  ;STACK OF INTERUPTS
000400                  MMEND=40C       ;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
                                .=12322
012322                  INIT:   SP      BR,SELB,SP0             ;CLEAR   SP0
(1)     000000                  MICPC=MICPC+1
(1)     012322  063220          <MOVE!SPX!BR!SELB!SP0>
012324                          SP      BR,SELB,SP3             ;PAGE ONE TRANSFER ADDRESS
(1)     000001                  MICPC=MICPC+1
(1)     012324  063223          <MOVE!SPX!BR!SELB!SP3>
012326                          SP      BR,SELB,SP17            ;CLEAR SP17
(1)     000002                  MICPC=MICPC+1
(1)     012326  063237          <MOVE!SPX!BR!SELB!SP17>
012330                          OUT     BR,<SELA!0INCON>        ;ZERO THE IN CONTROL CSR
(1)     000003                  MICPC=MICPC+1
(1)     012330  061200          <MOVE!WROUTX!BR!<SELA!0INCON>>
012332                          OUT     BR,<SELA!00CON>         ;ZERO THE OUT CONTROL CSR
(1)     000004                  MICPC=MICPC+1
(1)     012332  061202          <MOVE!WROUTX!BR!<SELA!00CON>>
012334                          SP      IMM,370,SP10            ;WRITE 5 ONE BITS TO THE HIGH ORDER
(1)     000005                  MICPC=MICPC+1
(1)     012334  003370          <MOVE!SPX!IMM!370!SP10>
                                                                ;BITS OF SP10
012336                  5$:     SP      BR,AA,SP10              ;SHIFT SP10 LEFT SETTING CARRY THE
(1)     000006                  MICPC=MICPC+1
(1)     012336  063130          <MOVE!SPX!BR!AA!SP10>
                                                                ;FIRST 5 TIMES THRU THE LOOP
012340                          MEMINC  BR,ADDC!SP3             ;WRITE A ONE TO THE FIRST 5 MEMORY
(1)     000007                  MICPC=MICPC+1
(1)     012340  076423          <MOVE!WRMEM!INCMAR!BR!<ADDC!SP3>>
                                                                ;LOCATIONS AND ZERO THE REST
012342                          SP      BR,INCA,SP0             ;INCREMENT COUNTER
(1)     000010                  MICPC=MICPC+1
(1)     012342  063060          <MOVE!SPX!BR!INCA!SP0>
012344                          Z       10$                     ;ALL DONE
(1)     000011                  MICPC=MICPC+1
(1)     012344  101413          <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
012346                          ALWAYS  5$                      ;KEEP GOING
(1)     000012                  MICPC=MICPC+1
(1)     012346  100406          <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
012350                  10$:    SPBR    IMM,1,SP1               ;WRITE A 1 TO THE BR AND SP1
(1)     000013                  MICPC=MICPC+1
(1)     012350  003401          <MOVE!SPBRX!IMM!1!SP1>
012352                          SP      BR,SELB,SP11            ;WRITE A 1 TO SP11
(1)     000014                  MICPC=MICPC+1
(1)     012352  063231          <MOVE!SPX!BR!SELB!SP11>
012354                          SP      BR,SELB,SP12            ;WRITE A 1 TO SP12
(1)     000015                  MICPC=MICPC+1
(1)     012354  063232          <MOVE!SPX!BR!SELB!SP12>
012356                          LDMA    IMM,PRTST               ;POINT MAR TO UNNUM MSG SKELETON
(1)     000016                  MICPC=MICPC+1
(1)     012356  010162          <MOVE!LDMAR!IMM!<PRTST&377>>
```

```
  504  012360                  PSTATI  NIDLE2                  ;WRITE PORT IDLE STATE
  (1)  012360                  MEMINC  IMM,<<NIDLE2-INIT&777/2>>
  (2)          000017          MICPC=MICPC+1
  (2)  012360  016520          <MOVE!WRMEM!INCMAR!IMM!<<NIDLE2-INIT&777/2>>>
  505  012362                  BRWRTE  IMM,226                 ;WRITE SYNC TO MEMORY
  (1)          000020          MICPC=MICPC+1
  (1)  012362  000626          <MOVE!WRTEBR!IMM!<226>>
  506  012364                  OUTPUT  BR,SELB!SYNC    ;LOAD THE SYNC REGISTER
  (1)          000021          MICPC=MICPC+1
  (1)  012364  062234          <MOVE!WROUT!BR!<SELB!SYNC>>
  507  012366                  MEMINC  IMM,3                   ;REP
  (1)          000022          MICPC=MICPC+1
  (1)  012366  016403          <MOVE!WRMEM!INCMAR!IMM!<3>>
  508  012370                  MEM     IMM,2                   ;NAK
  (1)          000023          MICPC=MICPC+1
  (1)  012370  002402          <MOVE!WRMEM!IMM!<2>>
  509  012372                  SP      MEMX!INCMAR,SELB,SP15           ;SET STARTING COUNT
  (1)          000024          MICPC=MICPC+1
  (1)  012372  057235          <MOVE!SPX!MEMX!INCMAR!SELB!SP15>
  510  012374                  MEMINC  IMM,6                   ;START
  (1)          000025          MICPC=MICPC+1
  (1)  012374  016406          <MOVE!WRMEM!INCMAR!IMM!<6>>
  511  012376                  MEMINC  IMM,7                   ;STACK
  (1)          000026          MICPC=MICPC+1
  (1)  012376  016407          <MOVE!WRMEM!INCMAR!IMM!<7>>
  512  012400                  MEMINC  IMM,1                   ;ACK
  (1)          000027          MICPC=MICPC+1
  (1)  012400  016401          <MOVE!WRMEM!INCMAR!IMM!<1>>
  513  012402                  LDMA    IMM,STC                 ;LOAD ADDRESS OF LAST TMT CHAIN
  (1)          000030          MICPC=MICPC+1
  (1)  012402  010070          <MOVE!LDMAR!IMM!<STC&377>>
  514  012404                  MEMINC  IMM,TML1                ;STORE ADDRESS OF FIRST TMT LINK
  (1)          000031          MICPC=MICPC+1
  (1)  012404  016473          <MOVE!WRMEM!INCMAR!IMM!<TML1>>
  515  012406                  MEMINC  IMM,TML1
  (1)          000032          MICPC=MICPC+1
  (1)  012406  016473          <MOVE!WRMEM!INCMAR!IMM!<TML1>>
  516  012410                  MEMINC  IMM,TML1
  (1)          000033          MICPC=MICPC+1
  (1)  012410  016473          <MOVE!WRMEM!INCMAR!IMM!<TML1>>
  517  012412                  LDMA    IMM,SRC                 ;LOAD ADDRESS OF LAST RECV CHAIN
  (1)          000034          MICPC=MICPC+1
  (1)  012412  010022          <MOVE!LDMAR!IMM!<SRC&377>>
  518  012414                  MEMINC  IMM,RCL1                ;SET UP ADDRESS OF FIRST RECV LINK
  (1)          000035          MICPC=MICPC+1
  (1)  012414  016425          <MOVE!WRMEM!INCMAR!IMM!<RCL1>>
  519  012416                  MEMINC  IMM,RCL1
  (1)          000036          MICPC=MICPC+1
  (1)  012416  016425          <MOVE!WRMEM!INCMAR!IMM!<RCL1>>
  520  012420                  MEMINC  IMM,RCL1
  (1)          000037          MICPC=MICPC+1
  (1)  012420  016425          <MOVE!WRMEM!INCMAR!IMM!<RCL1>>
  521  012422                  LDMA    IMM,NXTINT              ;ADDRESS OF NEXT INTERRUPT POINTER TO MAR
  (1)          000040          MICPC=MICPC+1
  (1)  012422  010240          <MOVE!LDMAR!IMM!<NXTINT&377>>
  522  012424                  MEMINC  IMM,INTSTK                      ;INITIALIZE NEXT INTERRUPT POINTER
```

```
(1)             000041
(1)  012424     016642
523  012426
(1)             000042
(1)  012426     002642
524  012430
(1)             000043
(1)  012430     000600
525  012432
(1)             000044
(1)  012432     061221
526
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<INTSTK>>
        MEM     IMM,INTSTK                      ;INITIALIZE INSERTION POINTER
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<INTSTK>>
        BRWRTE  IMM,200                 ;WRITE THE RUN BIT TO THE BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        OUT     BR,<SELB!OMAIN>         ;WRITE  THE RUN BIT TO MAINT CSR
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<SELB!OMAIN>>
                                        ;FALL INTO IDLE LOOP
```

```
                                        .SBTTL  IDLE--PROGRAM IDLE LOOP
                                        ;PROGRAM IDLE LOOP
                                        ;DISPATCHES TO APPROPRIATE SERVICE ROUTINES
                                        ;USES STATE POINTERS FOR TMT,RCV,CSR ACTIVITY
                                        ;
533                                     ;
534                                     ;
535                                     ;
536                                     ;
537                                     ;
538  012434                     IDLE:   BRWRTE  BR,<SELA!SP10>          ;READ TRANSMIT STATUS WORD FROM SP10 TO BR
(1)            000045                   MICPC=MICPC+1
(1)  012434    060610                   <MOVE!WRTEBR!BR!<SELA!SP10>>
539  012436                             BR1     TMTDA                   ;IF DATA TO SEND-- BRANCH
(1)            000046                   MICPC=MICPC+1
(1)  012436    112400                   <JUMP!BR1CON!<TMTDA-INIT&3000*4>!<TMTDA-INIT&777/2>>
540  012440                             BR0     TMTDA                   ;IF DATA TO SEND-- BRANCH
(1)            000047                   MICPC=MICPC+1
(1)  012440    112000                   <JUMP!BR0CON!<TMTDA-INIT&3000*4>!<TMTDA-INIT&777/2>>
541  012442                             ALWAYS  I1
(1)            000050                   MICPC=MICPC+1
(1)  012442    100452                   <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
542                                     ;
543  012444                     XEXIT:  SP      BR,SELB,SP2
(1)            000051                   MICPC=MICPC+1
(1)  012444    063222                   <MOVE!SPX!BR!SELB!SP2>
544  012446                     I1:     BRWRTE  IBUS,RCVCON             ;READ LINE UNIT RECEIVE CONTROL WORD
(1)            000052                   MICPC=MICPC+1
(1)  012446    020640                   <MOVE!WRTEBR!IBUS!<RCVCON>>
545  012450                             .BR4    BR,SELA,SP3!PAGE1       ;BRANCH BASED UPON RECV STATE
(1)            000053                   MICPC=MICPC+1
(1)  012450    167203                   <JUMP! BR4CON!BR!SELA!SP3!PAGE1>
546  012452                     I2:     LDMA    IMM,PRTST               ;ADDRESS PORT STATE
(1)            000054                   MICPC=MICPC+1
(1)  012452    010162                   <MOVE!LDMAR!IMM!<PRTST&377>>
547  012454                             .ALWAY  MEMX,SELB,0             ;INDEX
(1)            000055                   MICPC=MICPC+1
(1)  012454    140620                   <JUMP!ALCOND!MEMX!SELB!0>
```

```
                                                .SBTTL  BASSRV---- BASE SERVICE ROUTINE
549
550  012456                                     BASSRV: PSTATE  NIDLE2
(1)  012456                                             MEM     IMM,<<NIDLE2-INIT&777/2>>
(2)                 000056                              MICPC=MICPC+1
(2)  012456         002520                              <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
551  012460                                             LDMA    IMM,BASE                        ;CLEAR TO MAR SO IT POINTS TO BASE POINT
(1)                 000057                              MICPC=MICPC+1
(1)  012460         010017                              <MOVE!LDMAR!IMM!<BASE&377>>
552  012462                                             MEMINC  IBUS,PORT1                      ;READ CSR4
(1)                 000060                              MICPC=MICPC+1
(1)  012462         136500                              <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
553  012464                                             MEMINC  IBUS,PORT2                      ;READ CSR5
(1)                 000061                              MICPC=MICPC+1
(1)  012464         136520                              <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
554  012466                                             MEM     IBUS,PORT4
(1)                 000062                              MICPC=MICPC+1
(1)  012466         122560                              <MOVE!WRMEM!IBUS!<PORT4>>
555  012470                                             SP      IBUS,INCON,SP0                  ;READ INPUT CONTROL CSR
(1)                 000063                              MICPC=MICPC+1
(1)  012470         123000                              <MOVE!SPX!IBUS!INCON!SP0>
556  012472                                             BRWRTE  IMM,100                         ;CLEAR THE BR
(1)                 000064                              MICPC=MICPC+1
(1)  012472         000500                              <MOVE!WRTEBR!IMM!<100>>
557  012474                                             OUT     BR,<AANDB!0INCON>               ;CLEAR THE INCONTROL CSR
(1)                 000065                              MICPC=MICPC+1
(1)  012474         061260                              <MOVE!WROUTX!BR!<AANDB!0INCON>>
558  012476                                             BRWRTE  IMM,120                         ; MASK FOR HDX AND DTR
(1)                 000066                              MICPC=MICPC+1
(1)  012476         000520                              <MOVE!WRTEBR!IMM!<120>>
559  012500                                             OUTPUT  BR,<SELB!0MODEM>
(1)                 000067                              MICPC=MICPC+1
(1)  012500         062233                              <MOVE!WROUT!BR!<SELB!0MODEM>>
560  012502                                             BRWRTE  MEMX,SELB                       ;READ SEL6
(1)                 000070                              MICPC=MICPC+1
(1)  012502         040620                              <MOVE!WRTEBR!MEMX!<SELB>>
561  012504                                             BR4     RESUME                          ;IF SET RESUME
(1)                 000071                              MICPC=MICPC+1
(1)  012504         103100                              <JUMP!BR4CON!<RESUME-INIT&3000*4>!<RESUME-INIT&777/2>>
562  012506                                             LDMA    IMM,T                           ;LOAD ADDRESS OF TYPE FIELD
(1)                 000072                              MICPC=MICPC+1
(1)  012506         010153                              <MOVE!LDMAR!IMM!<T&377>>
563  012510                                             MEMINC  IMM,6                           ;WRITE START TYPE TO MEMORY
(1)                 000073                              MICPC=MICPC+1
(1)  012510         016406                              <MOVE!WRMEM!INCMAR!IMM!<6>>
564  012512                                             MEM     IMM,300                         ;WRITE SELECT AND FINAL TO MEMORY
(1)                 000074                              MICPC=MICPC+1
(1)  012512         002700                              <MOVE!WRMEM!IMM!<300>>
565  012514                                             SP      BR,DECA,SP1     ;TURN OFF INIT MODE
(1)                 000075                              MICPC=MICPC+1
(1)  012514         063161                              <MOVE!SPX!BR!DECA!SP1>
566  012516                                     BS1:    BRWRTE  IMM,241         ;SET OK TO SEND,STARTMODE AND UNNUM PENDING
(1)                 000076                              MICPC=MICPC+1
(1)  012516         000641                              <MOVE!WRTEBR!IMM!<241>>
567  012520                                             ALWAYS  SA3
(1)                 000077                              MICPC=MICPC+1
(1)  012520         110733                              <JUMP!ALCOND!<SA3-INIT&3000*4>!<SA3-INIT&777/2>>
```

```
   568  012522
   (1)              000100
   (1)  012522      003004
   569  012524
   (1)              000101
   (1)  012524      063070
   570
   571  012526
   (1)              000102
   (1)  012526      010017
   572  012530
   (1)              000103
   (1)  012530      000736
   573  012532
   (1)              000104
   (1)  012532      110462
   574  012534
   (1)              000105
   (1)  012534      010156
   575  012536
   (1)              000106
   (1)  012536      043310
   576  012540
   (1)              000107
   (1)  012540      057231
   577  012542
   (1)              000110
   (1)  012542      057232
   578  012544
   (1)              000111
   (1)  012544      043237
   579  012546
   (1)              000112
   (1)  012546      063170
   580
   581  012550
   (1)              000113
   (1)  012550      000520
   582  012552
   (1)              000114
   (1)  012552      063233
   583
   584  012554
   (1)              000115
   (1)  012554      063161
   585  012556
   (1)              000116
   (1)  012556      000600
   586  012560
   (1)              000117
   (1)  012560      110733
```

```
RESUME: SP      IMM,SP4,4               ;SET UP SP4 FOR COUNTING NPRS
        MICPC=MICPC+1
        <MOVE!SPX!IMM!SP4!4>
        SP      BR,INCA,SP10            ;SET UNNUMB MESSAGE PENDING TO
        MICPC=MICPC+1
        <MOVE!SPX!BR!INCA!SP10>
                                        ;TRICK TRANSMITTER CODE
        LDMA    IMM,BASE                ;ADDRESS BASE TABLE ADDRESS
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<BASE&377>>
        STATE   FUDGE                   ;SET TMTR STATE TO ENTER TABLE UPDATE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<FUDGE-INIT&777/2>>
        ALWAYS  TBO                     ;GO SET UP MXT BITS AND ADRESS OF BASE FOR NPRS
        MICPC=MICPC+1
        <JUMP!ALCOND!<TBO-INIT&3000*4>!<TBO-INIT&777/2>>
BS2:    LDMA    IMM,IMG10
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<IMG10&377>>
        SP      MEMX,AORB,SP10          ;RESTORE BIT 1 OF SP10
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!AORB!SP10>
        SP      MEMX!INCMAR,SELB,SP11           ;RESTORE SP11
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!INCMAR!SELB!SP11>
        SP      MEMX!INCMAR,SELB,SP12           ;RESTORE SP12
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!INCMAR!SELB!SP12>
        SP      MEMX,SELB,SP17          ;RESTORE        SP17
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP17>
        SP      BR,DECA,SP10            ;TURN OFF UNNUM MESSAGE PENDING AND
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP10>
                                        ;ZERO THE BRG
        STATE   NIDLE2                  ;PORT STATUS
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<NIDLE2-INIT&777/2>>
        SP      BR,SELB,SP13            ;SAVE IN SP13 - NOTE THAT STATUS READ INTO
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP13>
                                        ;RAM WAS TABLE UPDATE WHICH SAVED STATUS IN SP13
        SP      BR,DECA,SP1             ;CLEAR INIT MODE
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP1>
        BRWRTE  IMM,200                 ;SET OK TO SEND
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        ALWAYS  SA3
        MICPC=MICPC+1
        <JUMP!ALCOND!<SA3-INIT&3000*4>!<SA3-INIT&777/2>>
```

```
                                .SBTTL  NIDLE2---NO CSR ACTIVITY STATE
012562 000120           NIDLE2: BRWRTE  BR,SELA!SP1             ;READ PORT STATUS WORD
                                MICPC=MICPC+1
012562 060601                   <MOVE!WRTEBR!BR!<SELA!SP1>>
012564 000121                   BR4     OUTINT
                                MICPC=MICPC+1
012564 103220                   <JUMP!BR4CON!<OUTINT-INIT&3000*4>!<OUTINT-INIT&777/2>>
012566 000122                   SPBR    IBUS,INCON,SPO          ;READ INPUT CONTROL CSR
                                MICPC=MICPC+1
012566 123400                   <MOVE!SPBRX!IBUS!INCON!SPO>
012570 000123                   BRSHFT                          ;SHIFT IT RIGHT
                                MICPC=MICPC+1
012570 001620                   <MOVE!SHFTBR!WRTEBR!SELB>
012572 000124                   BR4     INWAT1                  ;IF RQI SET -- BRANCH
                                MICPC=MICPC+1
012572 103155                   <JUMP!BR4CON!<INWAT1-INIT&3000*4>!<INWAT1-INIT&777/2>>
                                                                ;TO RE-READ THE IN CNTRL REGISTER TO AVOID
                                                                ;A RACE IN MICRO-P READ/UNIBUS WRITE
012574                  NIDLE6: BRWRTE  BR,SELA!SP1             ;READ PORT STATUS
012574 000125                   MICPC=MICPC+1
012574 060601                   <MOVE!WRTEBR!BR!<SELA!SP1>>
012576 000126                   BR0     10$
                                MICPC=MICPC+1
012576 102131                   <JUMP!BR0CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
012600 000127                   SPBR    IBUS,UBBR,SPO           ;TIMER EXPIRES?
                                MICPC=MICPC+1
012600 123620                   <MOVE!SPBRX!IBUS!UBBR!SPO>
012602 000130                   BR4     TIMSRV
                                MICPC=MICPC+1
012602 113245                   <JUMP!BR4CON!<TIMSRV-INIT&3000*4>!<TIMSRV-INIT&777/2>>
012604 000131           10$:    SPBR    IBUS,MODEM,SPO          ;READ MODEM CONTROL CSR
                                MICPC=MICPC+1
012604 023660                   <MOVE!SPBRX!IBUS!MODEM!SPO>
012606 000132                   BRWRTE  BR,AA!SPO               ;SHIFT IT LEFT
                                MICPC=MICPC+1
012606 060520                   <MOVE!WRTEBR!BR!<AA!SPO>>
012610 000133                   BR4     SETDSR                  ;IF DSR SET, CLEAR FLAG
                                MICPC=MICPC+1
012610 103150                   <JUMP!BR4CON!<SETDSR-INIT&3000*4>!<SETDSR-INIT&777/2>>
012612 000134                   BRWRTE  BR,SELA!SP10            ;READ LINE STATUS WORD
                                MICPC=MICPC+1
012612 060610                   <MOVE!WRTEBR!BR!<SELA!SP10>>
012614 000135                   BRSHFT
                                MICPC=MICPC+1
012614 001620                   <MOVE!SHFTBR!WRTEBR!SELB>
012616 000136                   BR4     IDLE                    ;START MODE
                                MICPC=MICPC+1
012616 103045                   <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
012620 000137                   BRWRTE  BR,AA!SP1               ;READ PORT STATUS WORD
                                MICPC=MICPC+1
012620 060521                   <MOVE!WRTEBR!BR!<AA!SP1>>
012622 000140                   BR1     IDLE                    ;INIT MODE
                                MICPC=MICPC+1
012622 102445                   <JUMP!BR1CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
012624 000141                   BR7     IDLE                    ;DISCONNECT ERROR ALREADY SENT
                                MICPC=MICPC+1
```

```
(1)  012624  103445          <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
623  012626                  SPBR    IBUS,MAIN,SPO   ;READ THE MAINT REGISTER
(1)            000142        MICPC=MICPC+1
(1)  012626  123420          <MOVE!SPBRX!IBUS!MAIN!SPO>
624  012630                  BRWRTE  BR,ADD!SPO              ;SHIFT LEFT
(1)            000143        MICPC=MICPC+1
(1)  012630  060400          <MOVE!WRTEBR!BR!<ADD!SPO>>
625  012632                  BR4     IDLE                    ;LU LOOP -- EXIT
(1)            000144        MICPC=MICPC+1
(1)  012632  103045          <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
626  012634                  BRWRTE  IMM,100                 ;WRITE DISCONNECT ERROR
(1)            000145        MICPC=MICPC+1
(1)  012634  000500          <MOVE!WRTEBR!IMM!<100>>
627  012636                  SP      BR,AORB,SP1             ;FLAG ERROR RECORDED
(1)            000146        MICPC=MICPC+1
(1)  012636  063301          <MOVE!SPX!BR!AORB!SP1>
628  012640                  ALWAYS  ERRXX                   ;MAKE A CONTROL OUT
(1)            000147        MICPC=MICPC+1
(1)  012640  114671          <JUMP!ALCOND!<ERRXX-INIT&3000*4>!<ERRXX-INIT&777/2>>
629  012642  SETDSR:  BRWRTE  IMM,277                 ;CLEAR DISCONNECT ERROR FLAG
(1)            000150        MICPC=MICPC+1
(1)  012642  000677          <MOVE!WRTEBR!IMM!<277>>
630  012644                  ALWAYS  CLRIDL
(1)            000151        MICPC=MICPC+1
(1)  012644  100652          <JUMP!ALCOND!<CLRIDL-INIT&3000*4>!<CLRIDL-INIT&777/2>>
```

```
                        .SBTTL INWAIT---WAIT FOR RQI TO CLEAR
INWAIT: SPBR    IBUS,INCON,SPO          ;READ INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!INCON!SPO>
        BRWRTE  BR,<AA!SPO>             ;SHIFT IT LEFT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SPO>>
        BR7     NIDLE3                  ;INTERRUPT ENABLE HAS BEEN SET
        MICPC=MICPC+1
        <JUMP!BR7CON!<NIDLE3-INIT&3000*4>!<NIDLE3-INIT&777/2>>

INWAT1: SPBR    IBUS,INCON,SPO          ;READ THE INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!INCON!SPO>
        BR7     INWAT2                  ;READY IN STILL SET
        MICPC=MICPC+1
        <JUMP!BR7CON!<INWAT2-INIT&3000*4>!<INWAT2-INIT&777/2>>
NIDLE3: PSTATE  INWAT1                  ;UPDATE STATE TO INPUT
        MEM     IMM,<<INWAT1-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<INWAT1-INIT&777/2>>>
        BRWRTE  BR,AA!SPO               ;SHIFT CSR LEFT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SPO>>
        BR7     ININT
        MICPC=MICPC+1
        <JUMP!BR7CON!<ININT-INIT&3000*4>!<ININT-INIT&777/2>>
        PSTATE  INWAIT                  ;UPDATE STATE POINTER TO NO INTERRUPT GENERATED
        MEM     IMM,<<INWAIT-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<INWAIT-INIT&777/2>>>
NIDLE4: BRWRTE  IMM,200
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        OUT     BR,AORB!OINCON          ;SET THE RDYI
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AORB!OINCON>>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>

INWAT2: BRSHFT                          ;SHIFT THE BR RIGHT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     NIDLE6                  ;RQI SET--- GO AWAY
        MICPC=MICPC+1
        <JUMP!BR4CON!<NIDLE6-INIT&3000*4>!<NIDLE6-INIT&777/2>>
INSRV:  SPBR    IBUS,INCON,SPO          ;READ THE INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!INCON!SPO>
        BR1     30$                     ;--SENSE OR BASE
        MICPC=MICPC+1
        <JUMP!BR1CON!<30$-INIT&3000*4>!<30$-INIT&777/2>>
        BR0     10$                     ;CNTL I
        MICPC=MICPC+1
        <JUMP!BR0CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
```

```
   661  012710                       BRSHFT                          ;MUST BE BA/CC-SHIFT FOR IN OR OUT
   (1)            000173             MICPC=MICPC+1
   (1)  012710  001620               <MOVE!SHFTBR!WRTEBR!SELB>
   662  012712                       BR1     15$
   (1)            000174             MICPC=MICPC+1
   (1)  012712  102601               <JUMP!BRICON!<15$-INIT&3000*4>!<15$-INIT&777/2>>
   663  012714                       PSTATE  TBASRV                  ;TRANSMITTER
   (1)  012714                       MEM     IMM,<<TBASRV-INIT&777/2>>
   (2)            000175             MICPC=MICPC+1
   (2)  012714  002703               <MOVE!WRMEM!IMM!<<TBASRV-INIT&777/2>>>
   664  012716                       ALWAYS  20$
   (1)            000176             MICPC=MICPC+1
   (1)  012716  100602               <JUMP!ALCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
   665  012720              10$:     PSTATE  CTLSRV
   (1)  012720                       MEM     IMM,<<CTLSRV-INIT&777/2>>
   (2)            000177             MICPC=MICPC+1
   (2)  012720  002661               <MOVE!WRMEM!IMM!<<CTLSRV-INIT&777/2>>>
   666  012722                       ALWAYS  20$
   (1)            000200             MICPC=MICPC+1
   (1)  012722  100602               <JUMP!ALCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
   667  012724              15$:     PSTATE  RBASRV
   (1)  012724                       MEM     IMM,<<RBASRV-INIT&777/2>>
   (2)            000201             MICPC=MICPC+1
   (2)  012724  002726               <MOVE!WRMEM!IMM!<<RBASRV-INIT&777/2>>>
   668  012726              20$:     BRWRTE  BR,SELA!SP1             ;INIT MODE
   (1)            000202             MICPC=MICPC+1
   (1)  012726  060601               <MOVE!WRTEBR!BR!<SELA!SP1>>
   669  012730                       BRO     PROCER                  ;IF INIT MODE--ERROR
   (1)            000203             MICPC=MICPC+1
   (1)  012730  102211               <JUMP!BROCON!<PROCER-INIT&3000*4>!<PROCER-INIT&777/2>>
   670  012732                       ALWAYS  IDLE
   (1)            000204             MICPC=MICPC+1
   (1)  012732  100445               <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
   679  012734              30$:     BRO     35$                             ;IF BASE---PROCESS
   (1)            000205             MICPC=MICPC+1
   (1)  012734  102207               <JUMP!BROCON!<35$-INIT&3000*4>!<35$-INIT&777/2>>
   680  012736                       ALWAYS  PROCER
   (1)            000206             MICPC=MICPC+1
   (1)  012736  100611               <JUMP!ALCOND!<PROCER-INIT&3000*4>!<PROCER-INIT&777/2>>
   681  012740              35$:     BRWRTE  BR,SELA!SP1                     ;INIT MODE?
   (1)            000207             MICPC=MICPC+1
   (1)  012740  060601               <MOVE!WRTEBR!BR!<SELA!SP1>>
   682  012742                       BRO     BASSRV
   (1)            000210             MICPC=MICPC+1
   (1)  012742  102056               <JUMP!BROCON!<BASSRV-INIT&3000*4>!<BASSRV-INIT&777/2>>
   684  012744              PROCER:  BRWRTE  IMM,100                 ;CLEAR INPUT CONTROL CSR
   (1)            000211             MICPC=MICPC+1
   (1)  012744  000500               <MOVE!WRTEBR!IMM!<100>>
   685  012746                       OUT     BR,AANDB!0INCON         ;;
   (1)            000212             MICPC=MICPC+1
   (1)  012746  061260               <MOVE!WROUTX!BR!<AANDB!0INCON>>
   686  012750                       LDMA    IMM,<<RTHRS+3>>         ;ADDRESS ERROR LINK
   (1)            000213             MICPC=MICPC+1
   (1)  012750  010177               <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
   687  012752                       MEMINC  IMM,2
   (1)            000214             MICPC=MICPC+1
```

```
    (1)  012752  016402          <MOVE!WRMEM!INCMAR!IMM!<2>>
    688  012754                  MEM      IMM,0
    (1)               000215     MICPC=MICPC+1
    (1)  012754  002400          <MOVE!WRMEM!IMM!<0>>
    689  012756                  OUTPUT   MEMX,SELB!OMODEM          ;CLEAR DATA TERMINAL READY
    (1)               000216     MICPC=MICPC+1
    (1)  012756  042233          <MOVE!WROUT!MEMX!<SELB!OMODEM>>
    690  012760                  ALWAYS   RCEXX                     ;POST THE ERROR - FATAL
    (1)               000217     MICPC=MICPC+1
    (1)  012760  114527          <JUMP!ALCOND!<RCEXX-INIT&3000*4>!<RCEXX-INIT&777/2>>
```

```
                        .SBTTL  OUTINT---SET UP OUTPUT INTERRUPT [RDYO]
                    OUTINT:
                        PSTATE  OUTWAIT             ;PORT STATUS TO WAITING FOR OUT
                        MEM     IMM,<<OUTWAIT-INIT&777/2>>
                        MICPC=MICPC+1
                        <MOVE!WRMEM!IMM!<<OUTWAIT-INIT&777/2>>>
                                                    ;COMPLETION
                        LDMA    IMM,NXTINT          ;ADDRESS OF NEXT INTERRUPT POINTER
                        MICPC=MICPC+1
                        <MOVE!LDMAR!IMM!<NXTINT&377>>
                        LDMA    MEMX,SELB           ;NEXT INTERRUPT
                        MICPC=MICPC+1
                        <MOVE!LDMAR!MEMX!<SELB>>
                        SP      IBUS,OCON,SPO       ;READ THE OUTPUT CONTROL CSR
                        MICPC=MICPC+1
                        <MOVE!SPX!IBUS!OCON!SPO>
                        OUT     <MEMX!INCMAR>,<AORB!OOCON>  ;WRITE THE OUT CONTROL CSR
                        MICPC=MICPC+1
                        <MOVE!WROUTX!MEMX!INCMAR!<AORB!OOCON>>
                        LDMA    MEMX,SELB           ;ADDRESS LINK
                        MICPC=MICPC+1
                        <MOVE!LDMAR!MEMX!<SELB>>
                        BRWRTE  <BR!INCMAR>,<AA!SPO>  ;KICK PAST LINK STATUS BYTE
                        MICPC=MICPC+1
                        <MOVE!WRTEBR!BR!INCMAR!<AA!SPO>>
                                                    ;SHIFT CSRO IMAGE LEFT
                                                    ;***DO NOT CHANGE BR UNTIL BR7***
                        OUT     <MEMX!INCMAR>,<SELB!OPORT1>  ;WRITE LOW BYTE OF BA TO CSR
                        MICPC=MICPC+1
                        <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT1>>
                        OUT     <MEMX!INCMAR>,<SELB!OPORT2>  ;WRITE HIGH BYTE OF BA TO CSR
                        MICPC=MICPC+1
                        <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT2>>
                        OUT     <MEMX!INCMAR>,<SELB!OPORT4>  ;WRITE HIGH BYTE OF COUNT TO CSR
                        MICPC=MICPC+1
                        <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT4>>
                        OUT     <MEMX!INCMAR>,<SELB!OPORT3>  ;WRITE THE LOW BYTE OF COUNT
                        MICPC=MICPC+1
                        <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT3>>
                                                    ;***HERE IS BR7***
                        BR7     PE1                 ;INTERRUPT ENABLE IS SET
                        MICPC=MICPC+1
                        <JUMP!BR7CON!<PE1-INIT&3000*4>!<PE1-INIT&777/2>>
                                                    ;GENERATE AN INTERRUPT
                    PINT2:
                        LDMA    IMM,NXTINT          ;ADDRESS NEXT INTERRUPT QUEUE
                        MICPC=MICPC+1
                        <MOVE!LDMAR!IMM!<NXTINT&377>>
                        SP      MEMX,SELB,SPO       ;COPY ADDRESS FOR NEXT INT TO SPO
                        MICPC=MICPC+1
                        <MOVE!SPX!MEMX!SELB!SPO>
                        MEM     IMM,INTSTK          ;ASSUME WRAP AROUND CASE
                        MICPC=MICPC+1
                        <MOVE!WRMEM!IMM!<INTSTK>>
                        BRWRTE  IMM,<<MMEND-2>>     ;ADDRESS OF LAST INT IN STACK
                        MICPC=MICPC+1
```

```
(1)  013020  000776                    <MOVE!WRTEBR!IMM!<<MMEND-2>>>
727  013022                            CMP     BR,SPO                  ;SHOULD WE WRAP
(1)                                    MICPC=MICPC+1
(1)  013022  060360                    <SUBTC!BR!SPO>
728  013024                            Z       5$                      ;YES--BRANCH
(1)                                    MICPC=MICPC+1
(1)  013024  101644                    <JUMP!ZCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
729  013026                            BRWRTE  IMM,2                   ;OFFSET FOR NEXT POINTER
(1)                                    MICPC=MICPC+1
(1)  013026  000402                    <MOVE!WRTEBR!IMM!<2>>
730  013030                            MEM     BR,ADD!SPO              ;UPDATE POINTER
(1)                                    MICPC=MICPC+1
(1)  013030  062400                    <MOVE!WRMEM!BR!<ADD!SPO>>
731  013032                     5$:    SP      MEMX,SELB,SPO           ;COPY POINTER TO SPO
(1)                                    MICPC=MICPC+1
(1)  013032  043220                    <MOVE!SPX!MEMX!SELB!SPO>
732  013034                            LDMA    IMM,NXTSP               ;PICK UP START OF IN QUEUE
(1)                                    MICPC=MICPC+1
(1)  013034  010241                    <MOVE!LDMAR!IMM!<NXTSP&377>>
733  013036                            CMP     MEMX,SPO                ;COMPARE TO END
(1)                                    MICPC=MICPC+1
(1)  013036  040360                    <SUBTC!MEMX!SPO>
734  013040                            Z       10$                     ;IF EQUAL--CLEAR INT PENDING
(1)                                    MICPC=MICPC+1
(1)  013040  101651                    <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
735  013042                            ALWAYS  IDLE
(1)                                    MICPC=MICPC+1
(1)  013042  100445                    <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
736  013044                     10$:   BRWRTE  IMM,357                 ;MASK TO CLEAR INT PENDING
(1)                                    MICPC=MICPC+1
(1)  013044  000757                    <MOVE!WRTEBR!IMM!<357>>
737  013046                     CLRIDL: SP      BR,AANDB,SP1
(1)                                    MICPC=MICPC+1
(1)  013046  063261                    <MOVE!SPX!BR!AANDB!SP1>
738  013050                            ALWAYS  IDLE
(1)                                    MICPC=MICPC+1
(1)  013050  100445                    <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
  740
  741  013052                        .SBTTL  OUTWAI--WAIT FOR RDYO TO GO AWAY
  (1)            000254
  (1)  013052    123440  OUTWAI: SPBR    IBUS,OCON,SPO           ;READ OUTPUT CONTROL CSR
                                 MICPC=MICPC+1
                                 <MOVE!SPBRX!IBUS!OCON!SPO>
  742  013054                    BR7     NIDLE6                  ;RDYO SET --GET OUT
  (1)            000255          MICPC=MICPC+1
  (1)  013054    103525          <JUMP!BR7CON!<NIDLE6-INIT&3000*4>!<NIDLE6-INIT&777/2>>
  743  013056                    BRWRTE  IMM,100         ;CLEAR CONTROL BITS
  (1)            000256          MICPC=MICPC+1
  (1)  013056    000500          <MOVE!WRTEBR!IMM!<100>>
  744  013060                    OUT     BR,OOCON!AANDB
  (1)            000257          MICPC=MICPC+1
  (1)  013060    061262          <MOVE!WROUTX!BR!<OOCON!AANDB>>
  745  013062                    ALWAYS  INS13
  (1)            000260          MICPC=MICPC+1
  (1)  013062    100674          <JUMP!ALCOND!<INS13-INIT&3000*4>!<INS13-INIT&777/2>>
```

```
  747
  748  013064                    .SBTTL  CTLSRV--CNTL I SERVICE
  (1)            000261  CTLSRV: SPBR    IBUS,PORT4,SPO                  ;TO SPO
  (1)  013064    123560          MICPC=MICPC+1
                                 <MOVE!SPBRX!IBUS!PORT4!SPO>
  749  013066                    BRSHFT
  (1)            000262          MICPC=MICPC+1
  (1)  013066    001620          <MOVE!SHFTBR!WRTEBR!SELB>
  750  013070                    BR1     HDSEL                           ;IF SET IS HALF DUPLEX
  (1)            000263          MICPC=MICPC+1
  (1)  013070    102755          <JUMP!BR1CON!<HDSEL-INIT&3000*4>!<HDSEL-INIT&777/2>>
  751  013072                    BRWRTE  IMM,100                         ;MASK FOR DTR
  (1)            000264          MICPC=MICPC+1
  (1)  013072    000500          <MOVE!WRTEBR!IMM!<100>>
  752  013074                    OUTPUT  BR,SELB!OMODEM                  ;TURN OFF HALF-DUPLEX
  (1)            000265          MICPC=MICPC+1
  (1)  013074    062233          <MOVE!WROUT!BR!<SELB!OMODEM>>
  753  013076          INS11:    BRWRTE  DP,<SELA!SPO>                   ;RESTORE THE CNTL WORD
  (1)            000266          MICPC=MICPC+1
  (1)  013076    060600          <MOVE!WRTEBR!DP!<SELA!SPO>>
  754  013100                    BRO     CBOOT                           ;IF SET IS BOOT
  (1)            000267          MICPC=MICPC+1
  (1)  013100    102276          <JUMP!BROCON!<CBOOT-INIT&3000*4>!<CBOOT-INIT&777/2>>
  755  013102          INS12:    SP      IBUS,INCON,SPO                  ;READ THE INPUT CONTROL CSR
  (1)            000270          MICPC=MICPC+1
  (1)  013102    123000          <MOVE!SPX!IBUS!INCON!SPO>
  756  013104                    BRWRTE  IMM,100                         ;ZERO THE BR REGISTER EXCEPT INT ENABLE
  (1)            000271          MICPC=MICPC+1
  (1)  013104    000500          <MOVE!WRTEBR!IMM!<100>>
  757  013106                    OUT     BR,<AANDB!OINCON>               ;CLEAR IN CONTROL CSR
  (1)            000272          MICPC=MICPC+1
  (1)  013106    061260          <MOVE!WROUTX!BR!<AANDB!OINCON>>
  758  013110                    LDMA    IMM,PRTST                       ;ADDRESS PORT STATE
  (1)            000273          MICPC=MICPC+1
  (1)  013110    010162          <MOVE!LDMAR!IMM!<PRTST&377>>
  759  013112          INS13:    PSTATE  NIDLE2
  (1)  013112                    MEM     IMM,<<NIDLE2-INIT&777/2>>
  (2)            000274          MICPC=MICPC+1
  (2)  013112    002520          <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
  760  013114                    ALWAYS  IDLE
  (1)            000275          MICPC=MICPC+1
  (1)  013114    100445          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
  761                            ;
  762  013116          CBOOT:    BRWRTE  IMM,200                         ;MASK FOR BOOT MODE
  (1)            000276          MICPC=MICPC+1
  (1)  013116    000600          <MOVE!WRTEBR!IMM!<200>>
  763  013120                    SP      BR,AORB,SP1                     ;IN PORT STATUS WORD
  (1)            000277          MICPC=MICPC+1
  (1)  013120    063301          <MOVE!SPX!BR!AORB!SP1>
  764  013122                    BRWRTE  IMM,204                         ;MASK FOR OK TO SEND AND LINE IDLE
  (1)            000300          MICPC=MICPC+1
  (1)  013122    000604          <MOVE!WRTEBR!IMM!<204>>
  765  013124                    SP      BR,SELB,SP10                    ;IN LINE STATUS
  (1)            000301          MICPC=MICPC+1
  (1)  013124    063230          <MOVE!SPX!BR!SELB!SP10>
  766  013126                    ALWAYS  INS12
  (1)            000302          MICPC=MICPC+1
```

(1) 013126 100670 <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>

```
768
769  013130                          .SBTTL  TBASRV--TRANSMITTER BUFFER ADDRESS SERVICE
(1)               000303
(1)  013130  010072  TBASRV: LDMA    IMM,ETC                    ;GET POINTER TO END OF TMT CHAIN
                              MICPC=MICPC+1
                              <MOVE!LDMAR!IMM!<ETC&377>>
770  013132                   LDMA    MEMX,<SELB!SPX!SPO>        ;FIND THE LINK
(1)               000304     MICPC=MICPC+1
(1)  013132  053220           <MOVE!LDMAR!MEMX!<SELB!SPX!SPO>>
771  013134                   MEMINC  IMM,1                      ;BUFFER ASSIGNED IN IN LINK FLAGS
(1)               000305     MICPC=MICPC+1
(1)  013134  016401           <MOVE!WRMEM!INCMAR!IMM!<1>>
772  013136                   BRWRTE  <IMM!INCMAR>,6             ;POINT PAST NUMBER FIELD
(1)               000306     MICPC=MICPC+1
(1)  013136  014406           <MOVE!WRTEBR!IMM!INCMAR!<6>>
773                                                             ;SET BR FOR ADDITION TO SPO
774  013140                   MEMINC  IBUS,PORT1
(1)               000307     MICPC=MICPC+1
(1)  013140  136500           <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
775  013142                   MEMINC  IBUS,PORT2
(1)               000310     MICPC=MICPC+1
(1)  013142  136520           <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
776  013144                   MEMINC  IBUS,PORT4
(1)               000311     MICPC=MICPC+1
(1)  013144  136560           <MOVE!WRMEM!INCMAR!IBUS!<PORT4>>
777  013146                   MEMINC  IBUS,PORT3
(1)               000312     MICPC=MICPC+1
(1)  013146  136540           <MOVE!WRMEM!INCMAR!IBUS!<PORT3>>
778  013150                   SP      BR,ADD,SPO                 ;OFFSET TO NEXT TRANSMIT LINK
(1)               000313     MICPC=MICPC+1
(1)  013150  063000           <MOVE!SPX!BR!ADD!SPO>
779  013152                   BRWRTE  IMM,T                      ;LOAD BR WITH ADDRESS OF CHAIN END
(1)               000314     MICPC=MICPC+1
(1)  013152  000553           <MOVE!WRTEBR!IMM!<T>>
780  013154                   LDMA    IMM,ETC
(1)               000315     MICPC=MICPC+1
(1)  013154  010072           <MOVE!LDMAR!IMM!<ETC&377>>
781  013156                   CMP     BR,SPO                     ;END OF CHAIN?
(1)               000316     MICPC=MICPC+1
(1)  013156  060360           <SUBTC!BR!SPO>
782  013160                   Z       20$                        ;IF YES--BRANCH
(1)               000317     MICPC=MICPC+1
(1)  013160  101724           <JUMP!ZCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
783  013162                   MEM     BR,SELA!SPO                ;UPDATE THE END POINTER IN MEMORY
(1)               000320     MICPC=MICPC+1
(1)  013162  062600           <MOVE!WRMEM!BR!<SELA!SPO>>
784  013164          10$:     BRWRTE  IMM,2                   ;NUMBERED MSG PENDING MASK
(1)               000321     MICPC=MICPC+1
(1)  013164  000402           <MOVE!WRTEBR!IMM!<2>>
785  013166                   SP      BR,AORB,SP10            ;UPDATE LINE STATUS
(1)               000322     MICPC=MICPC+1
(1)  013166  063310           <MOVE!SPX!BR!AORB!SP10>
786  013170                   ALWAYS  INS12
(1)               000323     MICPC=MICPC+1
(1)  013170  100670           <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
787  013172          20$:     MEM     IMM,TML1                   ;WRAP IT AROUND
(1)               000324     MICPC=MICPC+1
(1)  013172  002473           <MOVE!WRMEM!IMM!<TML1>>
```

```
   788  013174                          ALWAYS  10$                            ;CONTINUE PROCESSING
   (1)            000325                MICPC=MICPC+1
   (1)  013174    100721                <JUMP!ALCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
```

```
                                .SBTTL  RBASRV--RECEIVE BUFFER ADDRESS SERVICE
013176                  RBASRV: LDMA    IMM,ERC                         ;ADDRES END OF RECEIVE CHAIN
                                MICPC=MICPC+1
                                (MOVE!LDMAR!IMM!(ERC&377))
013200                          LDMA    MEMX,(SELB!SPX!SPO)             ;GET THE POINTER TO LINK
                                MICPC=MICPC+1
                                (MOVE!LDMAR!MEMX!(SELB!SPX!SPO))
013202                          MEMINC  IMM,1
                                MICPC=MICPC+1
                                (MOVE!WRMEM!INCMAR!IMM!(1))
013204                          MEMINC  IBUS,PORT1
                                MICPC=MICPC+1
                                (MOVE!WRMEM!INCMAR!IBUS!(PORT1))
013206                          MEMINC  IBUS,PORT2
                                MICPC=MICPC+1
                                (MOVE!WRMEM!INCMAR!IBUS!(PORT2))
013210                          MEMINC  IBUS,PORT4
                                MICPC=MICPC+1
                                (MOVE!WRMEM!INCMAR!IBUS!(PORT4))
013212                          MEMINC  IBUS,PORT3
                                MICPC=MICPC+1
                                (MOVE!WRMEM!INCMAR!IBUS!(PORT3))
                                ;;;;NOTE INVERTED ORDER OF PORT 3 AND PORT4
013214                          LDMA    IMM,ERC
                                MICPC=MICPC+1
                                (MOVE!LDMAR!IMM!(ERC&377))
013216                          MEM     IMM,RCL1                        ;ASSUME WRAP AROUND CASE
                                MICPC=MICPC+1
                                (MOVE!WRMEM!IMM!(RCL1))
013220                          BRWRTE  IMM,RCL7                        ;GET ADDRESS OF END OF CAHIN AREA
                                MICPC=MICPC+1
                                (MOVE!WRTEBR!IMM!(RCL7))
013222                          CMP     BR,SPO                                  ;CHECK FOR END
                                MICPC=MICPC+1
                                (SUBTC!BR!SPO)
013224                          Z       INS12                                   ;IF EQUAL BRANCH
                                MICPC=MICPC+1
                                (JUMP!ZCOND!(INS12-INIT&3000*4)!(INS12-INIT&777/2))
013226                          BRWRTE  IMM,5                           ;CALCULATE ADDRESS OF NEXT LINK
                                MICPC=MICPC+1
                                (MOVE!WRTEBR!IMM!(5))
013230                          MEM     BR,ADD!SPO                      ;..
                                MICPC=MICPC+1
                                (MOVE!WRMEM!BR!(ADD!SPO))
013232                          ALWAYS  INS12                                   ;EXIT
                                MICPC=MICPC+1
                                (JUMP!ALCOND!(INS12-INIT&3000*4)!(INS12-INIT&777/2))
013234                  RA1:    BRWRTE  IMM,317                         ;MASK TO CLEAR START MODE AND CLR ACTIVE
                                MICPC=MICPC+1
                                (MOVE!WRTEBR!IMM!(317))
013236                          SPBR    BR,AANDB,SP10                   ;CLEAR BIT IN LINE STATUS WORD
                                MICPC=MICPC+1
                                (MOVE!SPBRX!BR!AANDB!SP10)
013240                  RA3:    BRWRTE  IMM,0                           ;CLEAR BR
                                MICPC=MICPC+1
                                (MOVE!WRTEBR!IMM!(0))
```

```
  810  013242
  (1)              000350
  (1)  013242  063233
  811  013244
  (1)              000351
  (1)  013244  000424
  812  013246
  (1)              000352
  (1)  013246  104422
  813
  814  013250
  (1)              000353
  (1)  013250  000402
  815  013252
  (1)              000354
  (1)  013252  114671
  816
```

```
        SP      BR,SELB,SP13            ;SET NUMB MESSAGE TYPE IN SP13
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP13>
        STATE   RCVB                    ;CHANGE RECEIVE STATE POINTER TO STATE B
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVB-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
        ;
RTHRES: BRWRTE  IMM,2
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        ALWAYS  ERRXX
        MICPC=MICPC+1
        <JUMP!ALCOND!<ERRXX-INIT&3000*4>!<ERRXX-INIT&777/2>>
        ;
```

```
830  013254
(1)            000355
(1)  013254    000500
831  013256
(1)            000356
(1)  013256    063310
832  013260
(1)            000357
(1)  013260    100666
833
834  013262
(1)            000360
(1)  013262    000700
835  013264
(1)            000361
(1)  013264    123220
836  013266
(1)            000362
(1)  013266    061311
841  013270
(1)            000363
(1)  013270    100634
843
844  013272
(1)            000364
(1)  013272    002730
845
846  013274
(1)            000365
(1)  013274    000402
847  013276
(1)            000366
(1)  013276    061231
848  013300
(1)            000367
(1)  013300    120620
849  013302
(1)            000370
(1)  013302    102767
850  013304
(1)            000371
(1)  013304    154620
851  013306
(1)            000372
(1)  013306    120620
852  013310
(1)            000373
(1)  013310    103364
853  013312
(1)            000374
(1)  013312    114733
854
855  013314
(1)            000375
(1)  013314    120600
856  013316
```

```
HDSEL:  BRWRTE  IMM,100                         ;HD MASK TO BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<100>>
        SP      BR,AORB,SP10                    ;UPDATE PORT STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        ALWAYS  INS11
        MICPC=MICPC+1
        <JUMP!ALCOND!<INS11-INIT&3000*4>!<INS11-INIT&777/2>>
;
PE1:    BRWRTE  IMM,300                 ;MASK FOR INTERUPT AND VECTOR THROUGH X04
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<300>>
        SP      IBUS,UBBR,SP0           ;READ BR CONTROL REG
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!UBBR!SP0>
        OUT     BR,<AORB!OBR>           ;INTERRUPT
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AORB!OBR>>
        ALWAYS  PINT2
        MICPC=MICPC+1
        <JUMP!ALCOND!<PINT2-INIT&3000*4>!<PINT2-INIT&777/2>>
;
HALTED: MEMADR  EM6
        MICPC=MICPC+1
        <MOVE!WRMEM!<EM6-INIT&777/2>>
                                        ;FALL INTO ACLOW
ACLOW:  BRWRTE  IMM,2                   ;CAUSE AN AC LOW
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        OUT     BR,<SELB!OBR>
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<SELB!OBR>>
5$:     BRWRTE  IBUS,UBBR               ;WAIT FOR IT TO COMPLETE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<UBBR>>
        BR1     5$
        MICPC=MICPC+1
        <JUMP!BR1CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        .ALWAY  MEMX,SELB,PAGE3
        MICPC=MICPC+1
        <JUMP!ALCOND!MEMX!SELB!PAGE3>
CKTIME: BRWRTE  IBUS,UBBR               ;READ BR CONTROL REG
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<UBBR>>
        BR4     HALTED
        MICPC=MICPC+1
        <JUMP!BR4CON!<HALTED-INIT&3000*4>!<HALTED-INIT&777/2>>
        ALWAYS  EM1
        MICPC=MICPC+1
        <JUMP!ALCOND!<EM1-INIT&3000*4>!<EM1-INIT&777/2>>
;
TBU1:   BRWRTE  IBUS,NPR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<NPR>>
        BR0     IDLE
```

```
    (1)             000376
    (1)  013316     102045          MICPC=MICPC+1
                                    <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
    857  013320                     ALWAYS  EC2
    (1)             000377          MICPC=MICPC+1
    (1)  013320     114760          <JUMP!ALCOND!<EC2-INIT&3000*4>!<EC2-INIT&777/2>>
    858
```

```
                                .=INIT+1000
                                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.
 860          013322    RCVA:  SP      IBUS,RCVDAT,SPO          ;READ RECEIVE CHARACTE  R TO SPO
 861          000377              MICPC=MICPC+1
 862                              <MOVE!SPX!IBUS!RCVDAT!SPO>
 863                              BRWRTE  BR,SELA!SP1              ;READ PORT STATUS WORD
 864                              MICPC=MICPC+1
 865                              <MOVE!WRTEBR!BR!<SELA!SP1>>
 866  013322                      BR0     5$                       ;IF INIT MODE---ONLY BOOT OK
(1)           000400              MICPC=MICPC+1
(1)   013322  023200              <JUMP!BROCON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
867   013324                      BR7     5$                       ;IF BOOT MODE---ONLY BOOT OK
(1)           000401              MICPC=MICPC+1
(1)   013324  060601              <JUMP!BR7CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
868   013326                      BRWRTE  IMM,201                  ;SOH TO BR
(1)           000402              MICPC=MICPC+1
(1)   013326  106012              <MOVE!WRTEBR!IMM!<201>>
869   013330                      CMP     BR,SPO                   ;COMPARE BR TO SPO
(1)           000403              MICPC=MICPC+1
(1)   013330  107412              <SUBTC!BR!SPO>
870   013332                      Z       RA1                      ;IF EQUAL-IS NUMBERED MESSAGE
(1)           000404              MICPC=MICPC+1
(1)   013332  000601              <JUMP!ZCOND!<RA1-INIT&3000*4>!<RA1-INIT&777/2>>
871   013334                      BRWRTE  IMM,5                    ;ENQ TO BR
(1)           000405              MICPC=MICPC+1
(1)   013334  060360              <MOVE!WRTEBR!IMM!<5>>
872   013336                      CMP     BR,SPO                   ;COMPARE ENQ TO SPO
(1)           000406              MICPC=MICPC+1
(1)   013336  101745              <SUBTC!BR!SPO>
873   013340                      Z       RA2                      ;IF EQUAL-IS UNNUMBERED MESSAGE
(1)           000407              MICPC=MICPC+1
(1)   013340  000405              <JUMP!ZCOND!<RA2-INIT&3000*4>!<RA2-INIT&777/2>>
874   013342              5$:     BRWRTE  IMM,220                  ;DLE TO BR
(1)           000410              MICPC=MICPC+1
(1)   013342  060360              <MOVE!WRTEBR!IMM!<220>>
875   013344                      CMP     BR,SPO                   ;COMPARE DLE TO SPO
(1)           000411              MICPC=MICPC+1
(1)   013344  105421              <SUBTC!BR!SPO>
876   013346                      Z       BOOT                     ;IF EQUAL IS BOOT
(1)           000412              MICPC=MICPC+1
(1)   013346  000620              <JUMP!ZCOND!<BOOT-INIT&3000*4>!<BOOT-INIT&777/2>>
877   013350              FLUSH:  OUTPUT  IMM,<200!ORCVCO>         ;FLUSH THE INPUT SILO
(1)           000413              MICPC=MICPC+1
(1)   013350  060360              <MOVE!WROUT!IMM!<200!ORCVCO>>
878   013352                      BRWRTE  IMM,357                  ;MASK TO CLEAR--CLEAR ACTIVE
(1)           000414              MICPC=MICPC+1
(1)   013352  105762              <MOVE!WRTEBR!IMM!<357>>
879   013354                      SP      BR,AANDB,SP10            ;IN LINE STATUS WORD
(1)           000415              MICPC=MICPC+1
(1)   013354  002212              <MOVE!SPX!BR!AANDB!SP10>
880   013356                      ALWAYS  RM1                      ;SET STATE TO RCVA AND RETURN TO IDLE
(1)           000416              MICPC=MICPC+1
(1)   013356  000757
881   013360
(1)           000417
(1)   013360  063270
882   013362
(1)           000420
```

```
    (1)  013362  114662
883  013364
    (1)          000421
    (1)  013364  000700
888  013366
    (1)          000422
    (1)  013366  063223
889  013370
    (1)          000423
    (1)  013370  100445

            <JUMP!ALCOND!<RM1-INIT&3000*4>!<RM1-INIT&777/2>>
RA2:        STATE   RCVI                    ;CHANGE RECEIVE STATE TO I
            MICPC=MICPC+1
            <MOVE!WRTEBR!IMM!<RCVI-INIT&777/2>>
REXIT:      SP      BR,SELB,SP3
            MICPC=MICPC+1
            <MOVE!SPX!BR!SELB!SP3>
            ALWAYS  IDLE
            MICPC=MICPC+1
            <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
                                        ;SBTTL  RCVB--ROUTINE TO HANDLE FIRST CHARACTER OF COUNT FIELD
                                        ;ENTERED FROM IDLE LOOP
                                        ;STORES COUNT FIELD AND SETS UP RCVC AS NEXT STATE
                                        ;
013372                          RCVB:
013372                                  SP      IBUS,RCVDAT,SP4         ;READ CHARACTER TO SP4
(1)     000424                          MICPC=MICPC+1
(1)     013372  023204                  <MOVE!SPX!IBUS!RCVDAT!SP4>
013374                                  LDMA    IMM,LRC                         ;LOAD ADDRESS OF START OF RECV  CHAIN
(1)     000425                          MICPC=MICPC+1
(1)     013374  010024                  <MOVE!LDMAR!IMM!<LRC&377>>
013376                                  SP      MEMX!LDMAR,SELB,SP14            ;COPY POINTER TO SP14
(1)     000426                          MICPC=MICPC+1
(1)     013376  053234                  <MOVE!SPX!MEMX!LDMAR!SELB!SP14>
                                                                        ;AND LOAD MAR WITH ADDRESS OF CURRENT BA
013400                                  BRWRTE  MEMX,INCMAR!SELB                ;READ FLAGS BYTE
(1)     000427                          MICPC=MICPC+1
(1)     013400  054620                  <MOVE!WRTEBR!MEMX!<INCMAR!SELB>>
013402                                  BR0     RB1                             ;RECV BUFFER ASSIGNED---CONTINUE
(1)     000430                          MICPC=MICPC+1
(1)     013402  106042                  <JUMP!BR0CON!<RB1-INIT&3000*4>!<RB1-INIT&777/2>>
013404                                  BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
(1)     000431                          MICPC=MICPC+1
(1)     013404  060601                  <MOVE!WRTEBR!BR!<SELA!SP1>>
013406                                  BR7     RB3                             ;MAINT MODE
(1)     000432                          MICPC=MICPC+1
(1)     013406  107440                  <JUMP!BR7CON!<RB3-INIT&3000*4>!<RB3-INIT&777/2>>
013410                                  LDMA    IMM,T                           ;ERROR--LOAD TYPE FIELD ADDRESS IN MAR
(1)     000433                          MICPC=MICPC+1
(1)     013410  010153                  <MOVE!LDMAR!IMM!<T&377>>
013412                                  MEMINC  IMM,2                           ;LOAD NAK TYPE
(1)     000434                          MICPC=MICPC+1
(1)     013412  016402                  <MOVE!WRMEM!INCMAR!IMM!<2>>
013414                                  MEM     IMM,310                         ;LOAD SUB-TYPE NO BUFFERS
(1)     000435                          MICPC=MICPC+1
(1)     013414  002710                  <MOVE!WRMEM!IMM!<310>>
013416                                  LDMA    IMM,NTLS
(1)     000436                          MICPC=MICPC+1
(1)     013416  010012                  <MOVE!LDMAR!IMM!<NTLS&377>>
013420                                  ALWAYS  RH5                             ;BRANCH TO SEND NAK ROUTINE
(1)     000437                          MICPC=MICPC+1
(1)     013420  104557                  <JUMP!ALCOND!<RH5-INIT&3000*4>!<RH5-INIT&777/2>>
013422                          RB3:    BRWRTE  IMM,4                           ;MASK FOR NO BUFFER AVAILABLE
(1)     000440                          MICPC=MICPC+1
(1)     013422  000404                  <MOVE!WRTEBR!IMM!<4>>
013424                                  SP      BR,AORB,SP1             ;SET THE FLAG
(1)     000441                          MICPC=MICPC+1
(1)     013424  063301                  <MOVE!SPX!BR!AORB!SP1>
013426                          RB1:    STATE   RCVC
(1)     000442                          MICPC=MICPC+1
(1)     013426  000462                  <MOVE!WRTEBR!IMM!<RCVC-INIT&777/2>>
013430                          RB0:    SP      BR,SELB,SP3
(1)     000443                          MICPC=MICPC+1
(1)     013430  063223                  <MOVE!SPX!BR!SELB!SP3>
013432                                  OUTPUT  <MEMX!INCMAR>,<SELB!OBA1>       ;OUTPUT LOW ORDER BYTE OF ADDRESS
(1)     000444                          MICPC=MICPC+1
```

```
(1)  013432  056226          <MOVE!WROUT!MEMX!INCMAR!<SELB!OBA1>>
915  013434                  OUTPUT  MEMX!INCMAR,<SELB!OBA2> ;OUTPUT HIGH BYTE OF ADDRESS
(1)            000445        MICPC=MICPC+1
(1)  013434  056227          <MOVE!WROUT!MEMX!INCMAR!<SELB!OBA2>>
916  013436                  SP      IBUS,UBBR,SP0           ;READ THE BUS REQ REGISTER
(1)            000446        MICPC=MICPC+1
(1)  013436  123220          <MOVE!SPX!IBUS!UBBR!SP0>
917  013440                  BRWRTE  IMM,101                 ;MASK OFF ALL BUT NXM AND VEC4 BITS
(1)            000447        MICPC=MICPC+1
(1)  013440  000501          <MOVE!WRTEBR!IMM!<101>>
918  013442                  SP      BR,AANDB,SP0            ;AND SAVE IN SP0
(1)            000450        MICPC=MICPC+1
(1)  013442  063260          <MOVE!SPX!BR!AANDB!SP0>
919  013444                  SP      IMM,300,SP5             ;MASK TO ISOLATE EX. MEM BITS
(1)            000451        MICPC=MICPC+1
(1)  013444  003305          <MOVE!SPX!IMM!300!SP5>
920                                                         ;NOTE THIS REALLY WRITES A 305 BUT THE
921                                                         ;5 GETS SHIFTED OUT
922  013446                  BRWRTE  MEMX,AANDB!SP5          ;MASK ALL BUT EX. MEM BITS
(1)            000452        MICPC=MICPC+1
(1)  013446  040665          <MOVE!WRTEBR!MEMX!<AANDB!SP5>>
923  013450                  BRSHFT                          ;SHIFT THEM INTO THE CORRECT POSITION
(1)            000453        MICPC=MICPC+1
(1)  013450  001620          <MOVE!SHFTBR!WRTEBR!SELB>
924  013452                  BRSHFT
(1)            000454        MICPC=MICPC+1
(1)  013452  001620          <MOVE!SHFTBR!WRTEBR!SELB>
925  013454                  BRSHFT
(1)            000455        MICPC=MICPC+1
(1)  013454  001620          <MOVE!SHFTBR!WRTEBR!SELB>
926  013456                  BRSHFT
(1)            000456        MICPC=MICPC+1
(1)  013456  001620          <MOVE!SHFTBR!WRTEBR!SELB>
927  013460                  OUT     BR,AORB!OBR                     ;WRITE EX MEM BITS OUT
(1)            000457        MICPC=MICPC+1
(1)  013460  061311          <MOVE!WROUTX!BR!<AORB!OBR>>
928  013462                  ALWAYS  IDLE
(1)            000460        MICPC=MICPC+1
(1)  013462  100445          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
929  013464          RB2:    ALWAYS  I2
(1)            000461        MICPC=MICPC+1
(1)  013464  100454          <JUMP!ALCOND!<I2-INIT&3000*4>!<I2-INIT&777/2>>
```

```
                                        .SBTTL  RCVC--ROUTINE TO HANDLE SECOND CHARACTER OF COUNT FIELD, SELECT AND FINA
                                        ;ENTERED FROM IDLE LOOP
                                        ;INTERPRETS SELECT AND FINAL
                                        ;CHECKS FOR COUNT TOO LARGE
                                        ;

013466                          RCVC:   ALWAYS  SELQSY                  ;"CALL" SELECT/QSYNC SUBROUTINE
(1)     000462                          MICPC=MICPC+1
(1)     013466  114470                  <JUMP!ALCOND!<SELQSY-INIT&3000*4>!<SELQSY-INIT&777/2>>
013470                                  LDMA    BR,<SELA!SP14>          ;LOAD ADDRESS OF CURRENT COUNT
(1)     000463                          MICPC=MICPC+1
(1)     013470  070214                  <MOVE!LDMAR!BR!<SELA!SP14>>
013472                                  BRWRTE  BR!INCMAR,SELA!SP1      ;READ STATUS BYTE
(1)     000464                          MICPC=MICPC+1
(1)     013472  074601                  <MOVE!WRTEBR!BR!INCMAR!<SELA!SP1>>
013474                                  BRWRTE  SHFTBR!INCMAR,SELB      ;SHIFT IT RIGHT
(1)     000465                          MICPC=MICPC+1
(1)     013474  015620                  <MOVE!WRTEBR!SHFTBR!INCMAR!<SELB>>
013476                                  BR1     RC5                     ;NO BUFFER ASSIGNED IN MAINT MODE
(1)     000466                          MICPC=MICPC+1
(1)     013476  106475                  <JUMP!BR1CON!<RC5-INIT&3000*4>!<RC5-INIT&777/2>>
013500                                  BRWRTE  IMM!INCMAR,77           ;MASK FOR COUNT BITS
(1)     000467                          MICPC=MICPC+1
(1)     013500  014477                  <MOVE!WRTEBR!IMM!INCMAR!<77>>
013502                                  SP      MEMX!INCMAR,SELB,SPO    ;COPY HIGH BYTE OF COUNT TO SPO
(1)     000470                          MICPC=MICPC+1
(1)     013502  057220                  <MOVE!SPX!MEMX!INCMAR!SELB!SPO>
013504                                  BRWRTE  BR,AANDB!SPO            ;MASK TO BR
(1)     000471                          MICPC=MICPC+1
(1)     013504  060660                  <MOVE!WRTEBR!BR!<AANDB!SPO>>
013506                                  CMP     BR,SP5          ;COMPARE HIGH ORDER BITS OF COUNT
(1)     000472                          MICPC=MICPC+1
(1)     013506  060365                  <SUBTC!BR!SP5>
013510                                  C       RCFATL                  ;IF CARRY--TOO BIG ERROR
(1)     000473                          MICPC=MICPC+1
(1)     013510  115116                  <JUMP!CCOND!<RCFATL-INIT&3000*4>!<RCFATL-INIT&777/2>>
013512                                  Z       RCLOW                   ;IF EQUAL COMPARE LOW ORDER BITS OF COUNT
(1)     000474                          MICPC=MICPC+1
(1)     013512  115513                  <JUMP!ZCOND!<RCLOW-INIT&3000*4>!<RCLOW-INIT&777/2>>
013514                          RC5:    STATE   RCVD                    ;SET NEXT STATE TO D
(1)     000475                          MICPC=MICPC+1
(1)     013514  000477                  <MOVE!WRTEBR!IMM!<RCVD-INIT&777/2>>
013516                                  ALWAYS  REXIT
(1)     000476                          MICPC=MICPC+1
(1)     013516  104422                  <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
```

```
                        .SBTTL  RCVD--ROUTINE TO HANDLE RESPONSE FIELD FOR NUMBERED MESSAGES
                        ;
013520                  SP      BR,DECA,SP4             ;DECREMENTLOW BYTE OF COUNT
(1)     000477                  MICPC=MICPC+1
(1)     013520  063164          <MOVE!SPX!BR!DECA!SP4>
013522                  C       RD3                     ;NO OVERFLOW
(1)     000500                  MICPC=MICPC+1
(1)     013522  105102          <JUMP!CCOND!<RD3-INIT&3000*4>!<RD3-INIT&777/2>>
013524                  SP      BR,DECA,SP5             ;OVERFLOW - DECREMENT HIGH BYTE
(1)     000501                  MICPC=MICPC+1
(1)     013524  063165          <MOVE!SPX!BR!DECA!SP5>
013526  RD3:    STATE   RCVE
(1)     000502                  MICPC=MICPC+1
(1)     013526  000523          <MOVE!WRTEBR!IMM!<RCVE-INIT&777/2>>
013530  RD2:    SP      BR,SELB,SP3             ;SAVE THE STATE
(1)     000503                  MICPC=MICPC+1
(1)     013530  063223          <MOVE!SPX!BR!SELB!SP3>
013532                  SPBR    IBUS,RCVDAT,SP0         ;INPUT THE CHARACTER
(1)     000504                  MICPC=MICPC+1
(1)     013532  023600          <MOVE!SPBRX!IBUS!RCVDAT!SP0>
013534                  BRWRTE  BR,SUB!SP17             ;COMPARE NEW R TO LAST R
(1)     000505                  MICPC=MICPC+1
(1)     013534  060757          <MOVE!WRTEBR!BR!<SUB!SP17>>
013536                  BR7     10$                     ;IF NEW IS GREATER---PROCESS
(1)     000506                  MICPC=MICPC+1
(1)     013536  107510          <JUMP!BR7CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
013540                  ALWAYS  IDLE
(1)     000507                  MICPC=MICPC+1
(1)     013540  100445          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
013542  10$:    BRWRTE  BR,SELA!SP1             ;READ STATUS BYTE
(1)     000510                  MICPC=MICPC+1
(1)     013542  060601          <MOVE!WRTEBR!BR!<SELA!SP1>>
013544                  BR7     IDLE                    ;MAINT. MODE - GET OUT
(1)     000511                  MICPC=MICPC+1
(1)     013544  103445          <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
013546                  BRWRTE  BR,SELA!SP10
(1)     000512                  MICPC=MICPC+1
(1)     013546  060610          <MOVE!WRTEBR!BR!<SELA!SP10>>
013550                  BRSHFT
(1)     000513                  MICPC=MICPC+1
(1)     013550  001620          <MOVE!SHFTBR!WRTEBR!SELB>
013552                  BR4     IDLE
(1)     000514                  MICPC=MICPC+1
(1)     013552  103045          <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
013554                  LDMA    IMM,ISP17               ;ADDRESS LAST ACKED IMAGE
(1)     000515                  MICPC=MICPC+1
(1)     013554  010155          <MOVE!LDMAR!IMM!<ISP17&377>>
013556                  MEM     BR,SELA!SP0             ;COPY THE CHAR
(1)     000516                  MICPC=MICPC+1
(1)     013556  062600          <MOVE!WRMEM!BR!<SELA!SP0>>
013560  RD5:    BRWRTE  IMM,REPST!LDMAR         ;SET UP COUNT FOR TIMER
(1)     000517                  MICPC=MICPC+1
(1)     013560  010403          <MOVE!WRTEBR!IMM!<REPST!LDMAR>>
                                                ;****DEPENDENT ON REPST = 2
013562                  MEM     IMM,1                   ;RESET REP THRESHOLD
(1)     000520                  MICPC=MICPC+1
```

```
    (1)  013562  002401          <MOVE!WRMEM!IMM!<1>>
    971  013564                  SP      BR,SELB,SP15            ;RESET THE COUNT
    (1)          000521          MICPC=MICPC+1
    (1)  013564  063235          <MOVE!SPX!BR!SELB!SP15>
    972  013566                  ALWAYS  IDLE
    (1)          000522          MICPC=MICPC+1
    (1)  013566  100445          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
                                        .SBTTL  RCVE--ROUTINE TO HANDLE N FIELD OF NUMBERED MESSAGE
                                        ;
013570                          RCVE:   BRWRTE  BR,SELA!SP1             ;READ THE STATUS BYTE
(1)     000523                          MICPC=MICPC+1
(1)  013570  060601                     <MOVE!WRTEBR!BR!<SELA!SP1>>
013572                                  BR7     RCVQ
(1)     000524                          MICPC=MICPC+1
(1)  013572  107713                     <JUMP!BR7CON!<RCVQ-INIT&3000*4>!<RCVQ-INIT&777/2>>
013574                                  BRWRTE  IBUS,RCVDAT             ;INPUT THE CHARACTER
(1)     000525                          MICPC=MICPC+1
(1)  013574  020600                     <MOVE!WRTEBR!IBUS!<RCVDAT>>
013576                                  CMP     BR,SP11
(1)     000526                          MICPC=MICPC+1
(1)  013576  060371                     <SUBTC!BR!SP11>
013600                                  Z       5$
(1)     000527                          MICPC=MICPC+1
(1)  013600  105532                     <JUMP!ZCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
013602                                  SP      BR,DECA,SP13            ;FORCE MSG TYPE TO -1
(1)     000530                          MICPC=MICPC+1
(1)  013602  063173                     <MOVE!SPX!BR!DECA!SP13>
013604                                  ALWAYS  RE2
(1)     000531                          MICPC=MICPC+1
(1)  013604  104533                     <JUMP!ALCOND!<RE2-INIT&3000*4>!<RE2-INIT&777/2>>
013606                          5$:     SP      BR,INCA,SP11            ;UPDATE R FIELD
(1)     000532                          MICPC=MICPC+1
(1)  013606  063071                     <MOVE!SPX!BR!INCA!SP11>
013610                          RE2:    STATE   RCVF                    ;NEXT RECEIVE STATE IS F
(1)     000533                          MICPC=MICPC+1
(1)  013610  000535                     <MOVE!WRTEBR!IMM!<RCVF-INIT&777/2>>
013612                                  ALWAYS  REXIT
(1)     000534                          MICPC=MICPC+1
(1)  013612  104422                     <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
```

```
  987
  988  013614                  .SBTTL  RCVF--ROUTINE TO IGNORE ADDRESS
  (1)          000535  RCVF:   STATE   RCVG
  (1)  013614  000540          MICPC=MICPC+1
                               <MOVE!WRTEBR!IMM!<RCVG-INIT&777/2>>
  989  013616  RCVF1:  NOP     IBUS,RCVDAT,0           ;INPUT CHARACTER - AND DISCARD
  (1)          000536          MICPC=MICPC+1
  (1)  013616  020200          <IBUS!RCVDAT!0>
  990  013620                  ALWAYS  REXIT
  (1)          000537          MICPC=MICPC+1
  (1)  013620  104422          <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
  991                          ;
  992                          .SBTTL  RCVG--ROUTINE TO IGNORE CRC1
  993                          ;
  994  013622  RCVG:   STATE   RCVH                   ;NEXT STATE IS RCVH
  (1)          000540          MICPC=MICPC+1
  (1)  013622  000542          <MOVE!WRTEBR!IMM!<RCVH-INIT&777/2>>
  995  013624                  ALWAYS  RCVF1
  (1)          000541          MICPC=MICPC+1
  (1)  013624  104536          <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
```

```
 997
 998
 999  013626
1000  013626
 (1)            000542
 (1)  013626    023200
1001  013630
 (1)            000543
 (1)  013630    020640
1002  013632
 (1)            000544
 (1)  013632    116167
1003  013634
 (1)            000545
 (1)  013634    060601
1004  013636
 (1)            000546
 (1)  013636    107751
1005  013640
 (1)            000547
 (1)  013640    060610
1006  013642
 (1)            000550
 (1)  013642    001620
1007  013644
 (1)            000551
 (1)  013644    117315
1008  013646
 (1)            000552
 (1)  013646    010153
1009  013650
 (1)            000553
 (1)  013650    016402
1010  013652
 (1)            000554
 (1)  013652    016701
1011  013654
 (1)            000555
 (1)  013654    062617
1012  013656
 (1)            000556
 (1)  013656    010013
1013  013660
 (1)            000557
 (1)  013660    043220
1014  013662
 (1)            000560
 (1)  013662    062460
1015  013664
 (1)            000561
 (1)  013664    010001
1016  013666
 (1)            000562
 (1)  013666    040620
1017  013670
 (1)            000563
```

```
                .SBTTL  RCVH--ROUTINE TO HANDLE CRC2 AND TO DISPATCH NUMBERED AND UNNUMBERED TYP
                ;
RCVH:
                SP      IBUS,RCVDAT,SPO                 ;GET CHAR IN SPO
                MICPC=MICPC+1
                <MOVE!SPX!IBUS!RCVDAT!SPO>
                BRWRTE  IBUS,RCVCON                     ;READ RECVR CONTROL REGISTER
                MICPC=MICPC+1
                <MOVE!WRTEBR!IBUS!<RCVCON>>
                BRO     TDON1                           ;IF BCC MATCH SET CRC IS GOOD
                MICPC=MICPC+1
                <JUMP!BROCON!<TDON1-INIT&3000*4>!<TDON1-INIT&777/2>>
                BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
                MICPC=MICPC+1
                <MOVE!WRTEBR!BR!<SELA!SP1>>
                BR7     RHX                             ;MAINT MODE
                MICPC=MICPC+1
                <JUMP!BR7CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
                BRWRTE  DP,<SELA!SP10>          ;READ PORT STATUS WORD TO BR
                MICPC=MICPC+1
                <MOVE!WRTEBR!DP!<SELA!SP10>>
                BRSHFT
                MICPC=MICPC+1
                <MOVE!SHFTBR!WRTEBR!SELB>
                BR4     SNAK1                   ;IF START MODE--PROCEED TO RESEND START
                MICPC=MICPC+1
                <JUMP!BR4CON!<SNAK1-INIT&3000*4>!<SNAK1-INIT&777/2>>
                LDMA    IMM,T                           ;ELSE BCC ERROR--LOAD ADDRESS OF TYPE FI
                MICPC=MICPC+1
                <MOVE!LDMAR!IMM!<T&377>>
                MEMINC  IMM,2                           ;WRITE NAK TYPE
                MICPC=MICPC+1
                <MOVE!WRMEM!INCMAR!IMM!<2>>
                MEMINC  IMM,301                         ;WRITE HEADER BCC ERROR SUBTYPE
                MICPC=MICPC+1
                <MOVE!WRMEM!INCMAR!IMM!<301>>
                MEM     BR,SELA!SP17            ;RESTORE LAST ACKED  IMAGE
                MICPC=MICPC+1
                <MOVE!WRMEM!BR!<SELA!SP17>>
                LDMA    IMM,NHDS                ;ADDRESS CUM ERROR COUNTER
                MICPC=MICPC+1
                <MOVE!LDMAR!IMM!<NHDS&377>>
RH5:            SP      MEMX,SELB,SPO           ;WRITE IT TO SPO
                MICPC=MICPC+1
                <MOVE!SPX!MEMX!SELB!SPO>
                MEM     BR,INCA!SPO             ;INCREMENT IT
                MICPC=MICPC+1
                <MOVE!WRMEM!BR!<INCA!SPO>>
                LDMA    IMM,NAKST               ;ADDRESS NAKS TMTED DYNAMIC
                MICPC=MICPC+1
                <MOVE!LDMAR!IMM!<NAKST&377>>
                BRWRTE  MEMX,SELB               ;WRITE IT TO BR
                MICPC=MICPC+1
                <MOVE!WRTEBR!MEMX!<SELB>>
                BSHFTB                          ;SHIFT IT RIGHT
                MICPC=MICPC+1
```

```
  (1)  013670  061620          <MOVE!SHFTBR!SELB!BR>
1018   013672                 MEM     BR,SELB                 ;UPDATE IT
  (1)           000564        MICPC=MICPC+1
  (1)  013672  062620          <MOVE!WRMEM!BR!<SELB>>
1019   013674                 BRO     NTHRES                  ;BRANCH IF THRESHOLD EXCEEDED
  (1)           000565        MICPC=MICPC+1
  (1)  013674  116264          <JUMP!BROCON!<NTHRES-INIT&3000*4>!<NTHRES-INIT&777/2>>
1020   013676                 ALWAYS  SNAK
  (1)           000566        MICPC=MICPC+1
  (1)  013676  114712          <JUMP!ALCOND!<SNAK-INIT&3000*4>!<SNAK-INIT&777/2>>
1021   013700          RH3:   BRWRTE  DP,<DECA!SP13>          ;LOAD TYPE RECEIVED--DECREMENTING
  (1)           000567        MICPC=MICPC+1
  (1)  013700  060573          <MOVE!WRTEBR!DP!<DECA!SP13>>
1022   013702                 Z       RH1                     ;IF ALUOUT IS ALL ONES IS NUMBERED MSG
  (1)           000570        MICPC=MICPC+1
  (1)  013702  115505          <JUMP!ZCOND!<RH1-INIT&3000*4>!<RH1-INIT&777/2>>
1023   013704                 RSTATE  RCVA
  (1)           000571        MICPC=MICPC+1
  (1)  013704  000400          <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
  (1)           000572        MICPC=MICPC+1
  (1)  013706  063223          <MOVE!SPX!BR!SELB!SP3>
1024   013710                 BRWRTE  DP,<SELA!SP10>          ;LOAD LINE STATUS WORD IN BR
  (1)           000573        MICPC=MICPC+1
  (1)  013710  060610          <MOVE!WRTEBR!DP!<SELA!SP10>>
1025   013712                 BR4     FLUSH1
  (1)           000574        MICPC=MICPC+1
  (1)  013712  117040          <JUMP!BR4CON!<FLUSH1-INIT&3000*4>!<FLUSH1-INIT&777/2>>
1026   013714          CG1:   BRSHFT                          ;SHIFT RIGHT
  (1)           000575        MICPC=MICPC+1
  (1)  013714  001620          <MOVE!SHFTBR!WRTEBR!SELB>
1027   013716                 BR4     10$
  (1)           000576        MICPC=MICPC+1
  (1)  013716  107204          <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1028   013720                 LDMA    IMM,TYPTAB              ;ADDRESS TYPE TABLE
  (1)           000577        MICPC=MICPC+1
  (1)  013720  010163          <MOVE!LDMAR!IMM!<TYPTAB&377>>
1029   013722                 CMP     <MEMX!INCMAR>,SP13
  (1)           000600        MICPC=MICPC+1
  (1)  013722  054373          <SUBTC!MEMX!INCMAR!SP13>
1030   013724                 Z       REP
  (1)           000601        MICPC=MICPC+1
  (1)  013724  115404          <JUMP!ZCOND!<REP-INIT&3000*4>!<REP-INIT&777/2>>
1031   013726                 CMP     <MEMX!INCMAR>,SP13
  (1)           000602        MICPC=MICPC+1
  (1)  013726  054373          <SUBTC!MEMX!INCMAR!SP13>
1032   013730                 Z       NAK
  (1)           000603        MICPC=MICPC+1
  (1)  013730  115443          <JUMP!ZCOND!<NAK-INIT&3000*4>!<NAK-INIT&777/2>>
1033   013732          10$:   LDMA    IMM,TYPSTT              ;SET POINTER TO START TYPE
  (1)           000604        MICPC=MICPC+1
  (1)  013732  010165          <MOVE!LDMAR!IMM!<TYPSTT&377>>
1034   013734                 CMP     <MEMX!INCMAR>,SP13
  (1)           000605        MICPC=MICPC+1
  (1)  013734  054373          <SUBTC!MEMX!INCMAR!SP13>
1035   013736                 Z       START
  (1)           000606        MICPC=MICPC+1
```

```
  (1)  013736  115413
1036
1037  013740
  (1)          000607
  (1)  013740  054373
1038  013742
  (1)          000610
  (1)  013742  115425
1039  013744
  (1)          000611
  (1)  013744  054373
1040  013746
  (1)          000612
  (1)  013746  105620
1041  013750
  (1)          000613
  (1)  013750  100445
1043  013752
  (1)          000614
  (1)  013752  023640
1044  013754
  (1)          000615
  (1)  013754  060400
1045  013756
  (1)          000616
  (1)  013756  103452
1046  013760
  (1)          000617
  (1)  013760  110405
1047  013762
  (1)          000620
  (1)  013762  060530
1048  013764
  (1)          000621
  (1)  013764  107223
1049  013766
  (1)          000622
  (1)  013766  100445
1050  013770
  (1)          000623
  (1)  013770  000727
1051  013772
  (1)          000624
  (1)  013772  063270
1052  013774
  (1)          000625
  (1)  013774  104517
```

```
        <JUMP!ZCOND!<START-INIT&3000*4>!<START-INIT&777/2>>
                                                ;STACK TYPE
        CMP     <MEMX!INCMAR>,SP13
        MICPC=MICPC+1
        <SUBTC!MEMX!INCMAR!SP13>
        Z       STACK
        MICPC=MICPC+1
        <JUMP!ZCOND!<STACK-INIT&3000*4>!<STACK-INIT&777/2>>
        CMP     <MEMX!INCMAR>,SP13      ;ACK TYPE
        MICPC=MICPC+1
        <SUBTC!MEMX!INCMAR!SP13>
        Z       ACK
        MICPC=MICPC+1
        <JUMP!ZCOND!<ACK-INIT&3000*4>!<ACK-INIT&777/2>>
        ALWAYS  IDLE                            ;OTHERWISE IGNORE--MUST BE OBS MSG
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
RCVCK:  SPBR    IBUS,RCVCON,SPO                 ;READ RCVR CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!RCVCON!SPO>
        BRWRTE  BR,ADD!SPO                      ;SHIFT LEFT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<ADD!SPO>>
        BR7     I1
        MICPC=MICPC+1
        <JUMP!BR7CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
        ALWAYS  TA1
        MICPC=MICPC+1
        <JUMP!ALCOND!<TA1-INIT&3000*4>!<TA1-INIT&777/2>>
ACK:    BRWRTE  BR,AA!SP10                      ;READ LINE STATUS-SHIFTING LEFT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SP10>>
        BR4     5$                              ;IF START RECD -- CLEAR START MODE
        MICPC=MICPC+1
        <JUMP!BR4CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
5$:     BRWRTE  IMM,327                         ;CLEAR START MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<327>>
        SP      BR,AANDB,SP10                   ;IN LINE STATUS
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP10>
        ALWAYS  RD5
        MICPC=MICPC+1
        <JUMP!ALCOND!<RD5-INIT&3000*4>!<RD5-INIT&777/2>>
```

```
                                ;*****************TIME CRITICAL CODE-- CHANGE WITH GREAT CARE*********************
                                        .SBTTL  RCVK01--ROUTINE TO HANDLE FIRST BYTE ODD RECEIVE
1055
1056
1057  013776                    RCVK01: SPBR    IBUS,NPR,SPO                    ;READ NPR REGISTER
 (1)          000626                    MICPC=MICPC+1
 (1)  013776  123600                    <MOVE!SPBRX!IBUS!NPR!SPO>
1058  014000                            BRO     IDLE
 (1)          000627                    MICPC=MICPC+1
 (1)  014000  102045                    <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1059  014002                            BRWRTE  IMM,200                         ;MASK FOR CO(BYTE TRANSFER)
 (1)          000630                    MICPC=MICPC+1
 (1)  014002  000600                    <MOVE!WRTEBR!IMM!<200>>
1060  014004                            OUT     BR,<AORB!ONPR>                  ;TURN ON CO
 (1)          000631                    MICPC=MICPC+1
 (1)  014004  061310                    <MOVE!WROUTX!BR!<AORB!ONPR>>
1061  014006                            STATE   RKE1
 (1)          000632                    MICPC=MICPC+1
 (1)  014006  000670                    <MOVE!WRTEBR!IMM!<RKE1-INIT&777/2>>
1062  014010                            ALWAYS  RCVK02
 (1)          000633                    MICPC=MICPC+1
 (1)  014010  104637                    <JUMP!ALCOND!<RCVK02-INIT&3000*4>!<RCVK02-INIT&777/2>>
1063                                    .SBTTL  RCVK0--PROCESS ODD CHARACTER
1072  014012                    RCVK0:  SPBR    IBUS,NPR,SPO                    ;IS AN NPR GOING
 (1)          000634                    MICPC=MICPC+1
 (1)  014012  123600                    <MOVE!SPBRX!IBUS!NPR!SPO>
1073  014014                            BRO     IDLE                            ;IF SO --- GET OUT
 (1)          000635                    MICPC=MICPC+1
 (1)  014014  102045                    <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1074  014016                            STATE   RCVKE
 (1)          000636                    MICPC=MICPC+1
 (1)  014016  000645                    <MOVE!WRTEBR!IMM!<RCVKE-INIT&777/2>>
1075  014020                    RCVK02: SP      BR,SELB,SP3                     ;SET STATE
 (1)          000637                    MICPC=MICPC+1
 (1)  014020  063223                    <MOVE!SPX!BR!SELB!SP3>
1076  014022                            OUTPUT  IBUS,RCVDAT!OUTDA2              ;OUTPUT A CHAR
 (1)          000640                    MICPC=MICPC+1
 (1)  014022  022203                    <MOVE!WROUT!IBUS!<RCVDAT!OUTDA2>>
1077  014024                    RK2:
1079  014024                    RK8:    BRWRTE  IMM,21                          ;SET OUT NPR (C1) AND NPR REQ
 (1)          000641                    MICPC=MICPC+1
 (1)  014024  000421                    <MOVE!WRTEBR!IMM!<21>>
1080  014026                            SP      IBUS,NPR,SPO                    ;READ NPR REGISTER
 (1)          000642                    MICPC=MICPC+1
 (1)  014026  123200                    <MOVE!SPX!IBUS!NPR!SPO>
1081  014030                    RK7:    OUT     BR,<AORB!ONPR>                  ;WRITE NPR REGISTER
 (1)          000643                    MICPC=MICPC+1
 (1)  014030  061310                    <MOVE!WROUTX!BR!<AORB!ONPR>>
1082  014032                            ALWAYS  IDLE
 (1)          000644                    MICPC=MICPC+1
 (1)  014032  100445                    <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
                        .SBTTL  RCVKE--HANDLE EVEN BYTES
1093
1094  014034           RCVKE:  BRWRTE  IBUS,NPR                        ;READ NPR CONTROL REGISTER
  (1)                           MICPC=MICPC+1
  (1)  014034  120600           <MOVE!WRTEBR!IBUS!<NPR>>
1099  014036                   BRO     IDLE
  (1)                           MICPC=MICPC+1
  (1)  014036  102045           <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1101  014040           RK5:    SP      IBUS,IOBA1,SPO                  ;READ LOW BYTE OF BA TO SP
  (1)                           MICPC=MICPC+1
  (1)  014040  023140           <MOVE!SPX!IBUS!IOBA1!SPO>
1102  014042                   OUTPUT  DP,<INCA!OBA1>                  ;WRITE INCREMENTED BA
  (1)                           MICPC=MICPC+1
  (1)  014042  062066           <MOVE!WROUT!DP!<INCA!OBA1>>
1103  014044           RK50:   SP      BR,DECA,SP4                     ;DECREMENT CHARACTER COUNT
  (1)                           MICPC=MICPC+1
  (1)  014044  063164           <MOVE!SPX!BR!DECA!SP4>
1104  014046                   C       10$                             ;NO OVERFLOW
  (1)                           MICPC=MICPC+1
  (1)  014046  105255           <JUMP!CCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1105  014050                   SP      BR,DECA,SP5                     ;OVERFLOW - DECREMENT HIGH BYTE
  (1)                           MICPC=MICPC+1
  (1)  014050  063165           <MOVE!SPX!BR!DECA!SP5>
1106  014052                   Z       RL3                             ;BYTE COUNT ZERO
  (1)                           MICPC=MICPC+1
  (1)  014052  105721           <JUMP!ZCOND!<RL3-INIT&3000*4>!<RL3-INIT&777/2>>
1107  014054           10$:    SP      IBUS,IOBA1,SPO                  ;READ INCREMENTED BA
  (1)                           MICPC=MICPC+1
  (1)  014054  023140           <MOVE!SPX!IBUS!IOBA1!SPO>
1108  014056                   OUTPUT  DP,<INCA!OBA1>                  ;WRITE INCREMENTED BA
  (1)                           MICPC=MICPC+1
  (1)  014056  062066           <MOVE!WROUT!DP!<INCA!OBA1>>
1109  014060                   C       ICBA22                          ;IF CARRY INC BA HIGH
  (1)                           MICPC=MICPC+1
  (1)  014060  115030           <JUMP!CCOND!<ICBA22-INIT&3000*4>!<ICBA22-INIT&777/2>>
1110  014062           RK9:    SP      IBUS,RCVDAT,SPO                 ;READ CHAR AND SAVE IN SPO
  (1)                           MICPC=MICPC+1
  (1)  014062  023200           <MOVE!SPX!IBUS!RCVDAT!SPO>
1111  014064           RK3:    OUTPUT  BR,SELA!OUTDA1                  ;WRITE THE CHARACTER
  (1)                           MICPC=MICPC+1
  (1)  014064  062202           <MOVE!WROUT!BR!<SELA!OUTDA1>>
1112  014066                   SP      BR,DECA,SP4                     ;DECREMENT THE COUNTOF BYTES
  (1)                           MICPC=MICPC+1
  (1)  014066  063164           <MOVE!SPX!BR!DECA!SP4>
1113  014070                   C       RK6                             ;NO OVERFLOW
  (1)                           MICPC=MICPC+1
  (1)  014070  105266           <JUMP!CCOND!<RK6-INIT&3000*4>!<RK6-INIT&777/2>>
1114  014072                   SP      BR,DECA,SP5                     ;DECREMENT HIGH BYTE OF COUNT
  (1)                           MICPC=MICPC+1
  (1)  014072  063165           <MOVE!SPX!BR!DECA!SP5>
1115  014074                   Z       RL4                             ;BYTE COUNT ZERO
  (1)                           MICPC=MICPC+1
  (1)  014074  111765           <JUMP!ZCOND!<RL4-INIT&3000*4>!<RL4-INIT&777/2>>
1125  014076           RK6:    STATE   RCVKO
  (1)                           MICPC=MICPC+1
  (1)  014076  000634           <MOVE!WRTEBR!IMM!<RCVKO-INIT&777/2>>
1126  014100                   ALWAYS  REXIT
```

```
  (1)           000667
  (1)  014100   104422
1128
1129  014102
  (1)           000670
  (1)  014102   123200
1130  014104
  (1)           000671
  (1)  014104   000577
1131  014106
  (1)           000672
  (1)  014106   061270
1132  014110
  (1)           000673
  (1)  014110   104651
1133
1134
1135  014112
  (1)           000674
  (1)  014112   023200
1136  014114
  (1)           000675
  (1)  014114   060601
1137  014116
  (1)           000676
  (1)  014116   117600
1138  014120
  (1)           000677
  (1)  014120   104661
```

```
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
        .
RKE1:   SP      IBUS,NPR,SPO                    ;READ NPR REGISTER
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!NPR!SPO>
        BRWRTE  IMM,177                         ;MASK FOR ALL BUT CO
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<177>>
        OUT     BR,<AANDB!ONPR>                 ;TURN OFF ALL BUT CO
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AANDB!ONPR>>
        ALWAYS  RK50
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK50-INIT&3000*4>!<RK50-INIT&777/2>>
;*****************END OF TIME CRITICAL PATH*************************************

RCVKE0: SP      IBUS,RCVDAT,SPO         ;READ CHARACTER AND SAVE IN SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!RCVDAT!SPO>
        BRWRTE  BR,SELA!SP1             ;READ STATUS BYTE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR7     PASWRD                  ;MAINT MODE - SEE IF RLD MESSAGE
        MICPC=MICPC+1
        <JUMP!BR7CON!<PASWRD-INIT&3000*4>!<PASWRD-INIT&777/2>>
        ALWAYS  RK3                     ;OTHERWISE PROCESS NORMALLY
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
```

```
1140
1141  014122                    .SBTTL  RCVI--STORE UNNUMBERED MESSAGE TYPE
RCVI:   SP      IBUS,RCVDAT,SP13        ;STORE UNNUMBERED TYPE
        MICPC=MICPC+1
 (1)            000700          <MOVE!SPX!IBUS!RCVDAT!SP13>
 (1)    014122  023213
1142  014124    STATE   RCVJ                    ;NEXT STATE IS J
        MICPC=MICPC+1
 (1)            000701          <MOVE!WRTEBR!IMM!<RCVJ-INIT&777/2>>
 (1)    014124  000703
1143  014126    ALWAYS  REXIT
        MICPC=MICPC+1
 (1)            000702          <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
 (1)    014126  104422          ;
1144
```

```
 1146
 1147  014130                          .SBTTL  RCVJ--ROUTINE TO HANDLE SUBTYPE FIELD,SELECT AND FINAL
 1147  014130                  RCVJ:   ALWAYS  SELQSY          ;"CALL" SELECT AND QSYNC SUBROUTINE
  (1)                                  MICPC=MICPC+1
  (1)  014130  114470                  <JUMP!ALCOND!<SELQSY-INIT&3000*4>!<SELQSY-INIT&777/2>>
 1148  014132                          STATE   RCVR                    ;NEXT STATE IS N
  (1)                                  MICPC=MICPC+1
  (1)  014132  000706                  <MOVE!WRTEBR!IMM!<RCVR-INIT&777/2>>
 1149  014134                          ALWAYS  REXIT
  (1)                                  MICPC=MICPC+1
  (1)  014134  104422                  <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
```

```
                        .SBTTL  RCVR--UNNUMBERED MESSAGE RESPONSE FIELD
                        ;ENTERED FROM IDLE LOOP
                        ;
014136                  BRWRTE  IMM,3                   ;REP MESSAGE TYPE TO BR
(1)             000706  MICPC=MICPC+1
(1)     014136  000403  <MOVE!WRTEBR!IMM!<3>>
014140                  NOP     BR,SUB,SP13             ;IS TYPE ACK OR NAK
(1)             000707  MICPC=MICPC+1
(1)     014140  060353  <BR!SUB!SP13>
014142                  STATE   RCVQ                    ;NEXT STATE IS RCVQ
(1)             000710  MICPC=MICPC+1
(1)     014142  000713  <MOVE!WRTEBR!IMM!<RCVQ-INIT&777/2>>
                                                        ;***NOTE THIS INSTR DOES NOT CLOCK "C"
014144                  C       RCVF1                   ;IF NOT IGNORE
(1)             000711  MICPC=MICPC+1
(1)     014144  105136  <JUMP!CCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
014146                  ALWAYS  RD2                     ;DO RANGE CHECKS
(1)             000712  MICPC=MICPC+1
(1)     014146  104503  <JUMP!ALCOND!<RD2-INIT&3000*4>!<RD2-INIT&777/2>>
```

```
                                        .SBTTL  RCVQ--UNNUMBERED MESSAGE--NUMBER FIELD
                                        ;ENTER  FROM IDLE
                                        ;
1161                                    ;
1162                                    ;
1163                                    ;
1164  014150                    RCVQ:   STATE   RCVF                    ;NEXT STATE IS ADDRESS
 (1)            000713                 MICPC=MICPC+1
 (1)  014150    000535                 <MOVE!WRTEBR!IMM!<RCVF-INIT&777/2>>
1165  014152                            ALWAYS  RCVF1
 (1)            000714                 MICPC=MICPC+1
 (1)  014152    104536                 <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
```

```
1167
1168
1169  014154                        .SBTTL  RCVL--PROCESS CRC3
                                    ;ENTERED FROM  IDLE LOOP
1169  014154                RCVL:   SPBR    IBUS,NPR,SPD                    ;READ NPR CONTROL
 (1)          000715                MICPC=MICPC+1
 (1)  014154  123600                <MOVE!SPBRX!IBUS!NPR!SPD>
1174  014156                        BRO     IDLE
 (1)          000716                MICPC=MICPC+1
 (1)  014156  102045                <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1176  014160                RL2:    BRWRTE  IMM,177                         ;MASK TO TURN OFF CO
 (1)          000717                MICPC=MICPC+1
 (1)  014160  000577                <MOVE!WRTEBR!IMM!<177>>
1177  014162                        OUT     BR,AANDB!ONPR
 (1)          000720                MICPC=MICPC+1
 (1)  014162  061270                <MOVE!WROUTX!BR!<AANDB!ONPR>>
1178  014164                RL3:    SPBR    IBUS,IOBA1,SPD                  ;READ LOW ORDER BYTE OF ADDRESS
 (1)          000721                MICPC=MICPC+1
 (1)  014164  023540                <MOVE!SPBRX!IBUS!IOBA1!SPD>
1179  014166                        OUTPUT  BR,INCA!OBA1                    ;INCREMENT THE LOW ORDER BYTE
 (1)          000722                MICPC=MICPC+1
 (1)  014166  062066                <MOVE!WROUT!BR!<INCA!OBA1>>
1180  014170                        SP      IBUS,IOBA2,SPD                  ;READ HIGH BYTE
 (1)          000723                MICPC=MICPC+1
 (1)  014170  023160                <MOVE!SPX!IBUS!IOBA2!SPD>
1181  014172                        OUTPUT  BR,AC!OBA2                      ;ADD CARRY TO HIGH BYTE
 (1)          000724                MICPC=MICPC+1
 (1)  014172  062107                <MOVE!WROUT!BR!<AC!OBA2>>
1182                                ;
1183  014174                        STATE   RCVM
 (1)          000725                MICPC=MICPC+1
 (1)  014174  000727                <MOVE!WRTEBR!IMM!<RCVM-INIT&777/2>>
1184  014176                        ALWAYS  RCVF1
 (1)          000726                MICPC=MICPC+1
 (1)  014176  104536                <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
1185                                ;
1190                                ;
```

```
                                        .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
1197  014200                    RCVM:   BRWRTE  IBUS,UBBR                       ;READ UNIBUS BR REGISTER
 (1)              000727                MICPC=MICPC+1
 (1)  014200  120620                    <MOVE!WRTEBR!IBUS!<UBBR>>
1198  014202                            BRO     NXMERR                          ;NON-EXISTANT MEMORY
 (1)              000730                MICPC=MICPC+1
 (1)  014202  116063                    <JUMP!BROCON!<NXMERR-INIT&3000*4>!<NXMERR-INIT&777/2>>
1199  014204                            SP      IBUS,RCVDAT,SPO                 ;READ CRC CHARACTER
 (1)              000731                MICPC=MICPC+1
 (1)  014204  023200                    <MOVE!SPX!IBUS!RCVDAT!SPO>
1200  014206                            BRWRTE  IBUS,RCVCON                     ;READ RECEIVER CONTROL REGISTER
 (1)              000732                MICPC=MICPC+1
 (1)  014206  020640                    <MOVE!WRTEBR!IBUS!<RCVCON>>
1201  014210                            BRO     RCVM1                           ;IF CRC GOOD -- PROCESS
 (1)              000733                MICPC=MICPC+1
 (1)  014210  116216                    <JUMP!BROCON!<RCVM1-INIT&3000*4>!<RCVM1-INIT&777/2>>
1202  014212                            BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
 (1)              000734                MICPC=MICPC+1
 (1)  014212  060601                    <MOVE!WRTEBR!BR!<SELA!SP1>>
1203  014214                            BR7     RHX                             ;CRC ERROR IN BOOT MODE - FLUSH
 (1)              000735                MICPC=MICPC+1
 (1)  014214  107751                    <JUMP!BR7CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
1204  014216                            LDMA    IMM,T                           ;ELSE SEND NAK --DATA ERROR
 (1)              000736                MICPC=MICPC+1
 (1)  014216  010153                    <MOVE!LDMAR!IMM!<T&377>>
1205  014220                            MEMINC  IMM,2                           ;NAK TYPE
 (1)              000737                MICPC=MICPC+1
 (1)  014220  016402                    <MOVE!WRMEM!INCMAR!IMM!<2>>
1206  014222                            MEMINC  IMM,302                         ;DATA ERROR SUBTYPE
 (1)              000740                MICPC=MICPC+1
 (1)  014222  016702                    <MOVE!WRMEM!INCMAR!IMM!<302>>
1207  014224                            LDMA    IMM,NDATS
 (1)              000741                MICPC=MICPC+1
 (1)  014224  010014                    <MOVE!LDMAR!IMM!<NDATS&377>>
1208  014226                            ALWAYS  RH5                             ;SEND NAK
 (1)              000742                MICPC=MICPC+1
 (1)  014226  104557                    <JUMP!ALCOND!<RH5-INIT&3000*4>!<RH5-INIT&777/2>>
1209
1210  014230                    RCVM0:  LDMA    IMM,<<RTHRS+3>>                 ;POINT TO ERROR WORD
 (1)              000743                MICPC=MICPC+1
 (1)  014230  010177                    <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
1211  014232                            BRWRTE  IMM,10                          ;MAINT MESSAGE ERROR
 (1)              000744                MICPC=MICPC+1
 (1)  014232  000410                    <MOVE!WRTEBR!IMM!<10>>
1212  014234                            ALWAYS RCEXY                            ;GIVE FATAL ERROR
 (1)              000745                MICPC=MICPC+1
 (1)  014234  114525                    <JUMP!ALCOND!<RCEXY-INIT&3000*4>!<RCEXY-INIT&777/2>>
```

```
                                        .SBTTL  EM2--PROCESS RLD MESSAGE
                                        ;ENTERED FROM IDLE LOOP
                                        ;IF RLD PASSWORD CHECKS TRIGGER THE BOOT ROM

1214
1215
1216
1217
1218  014236                    EM2:    BRWRTE  IBUS,RCVDAT             ;READ THE CHAR
 (1)          000746                    MICPC=MICPC+1
 (1)  014236  020600                    <MOVE!WRTEBR!IBUS!<RCVDAT>>
1219  014240                            CMP     BR,SP16                 ;IS IT A MATCH
 (1)          000747                    MICPC=MICPC+1
 (1)  014240  060376                    <SUBTC!BR!SP16>
1220  014242                            Z       EM3
 (1)          000750                    MICPC=MICPC+1
 (1)  014242  105757                    <JUMP!ZCOND!<EM3-INIT&3000*4>!<EM3-INIT&777/2>>
1221                                                                    ;FALL INTO RHX
1222  014244                    RHX:    BRWRTE  BR,AA!SP1               ;READ STATUS BYTE SHIFTED LEFT
 (1)          000751                    MICPC=MICPC+1
 (1)  014244  060521                    <MOVE!WRTEBR!BR!<AA!SP1>>
1223  014246                            BR4     10$                     ;DLE RECEIVED IN NORMAL MODE
 (1)          000752                    MICPC=MICPC+1
 (1)  014246  107354                    <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1224  014250                            ALWAYS  FLUSH                   ;ALREADY IN MAINT MODE
 (1)          000753                    MICPC=MICPC+1
 (1)  014250  104415                    <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
1225  014252                    10$:    BRWRTE  IMM,163                 ;MASK TO CLEAR ALL MAINT RELATED BITS
 (1)          000754                    MICPC=MICPC+1
 (1)  014252  000563                    <MOVE!WRTEBR!IMM!<163>>
1226  014254                            SP      BR,AANDB,SP1            ;CLEAR THEM
 (1)          000755                    MICPC=MICPC+1
 (1)  014254  063261                    <MOVE!SPX!BR!AANDB!SP1>
1227  014256                            ALWAYS  FLUSH
 (1)          000756                    MICPC=MICPC+1
 (1)  014256  104415                    <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
1228
1229  014260                    EM3:    SP      BR,DECA,SP4             ;DECREMENT CHARACTER COUNT BY ONE
 (1)          000757                    MICPC=MICPC+1
 (1)  014260  063164                    <MOVE!SPX!BR!DECA!SP4>
1230  014262                            Z       EMTRIG                  ;TRIGGER AC LOW
 (1)          000760                    MICPC=MICPC+1
 (1)  014262  115720                    <JUMP!ZCOND!<EMTRIG-INIT&3000*4>!<EMTRIG-INIT&777/2>>
1231  014264                            ALWAYS  IDLE
 (1)          000761                    MICPC=MICPC+1
 (1)  014264  100445                    <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
 1233
 1234  014266
  (1)                 000762
  (1)  014266  060601
 1235  014270
  (1)                 000763
  (1)  014270  103747
 1236  014272
  (1)                 000764
  (1)  014272  000610
 1237  014274
  (1)                 000765
  (1)  014274  063301
 1238  014276
  (1)                 000766
  (1)  014276  100747
 1239  014300
  (1)                 000767
  (1)  014300  000404
 1240  014302
  (1)                 000770
  (1)  014302  123200
 1241  014304
  (1)                 000771
  (1)  014304  061010
 1242  014306
  (1)                 000772
  (1)  014306  110753
 1243  014310
  (1)                 000773
  (1)  014310  000404
 1244  014312
  (1)                 000774
  (1)  014312  123220
 1245  014314
  (1)                 000775
  (1)  014314  061011
 1246  014316
  (1)                 000776
  (1)  014316  114767
 1247  014320
  (1)                 000777
  (1)  014320  000000
```

```
                .
BOOT:   BRWRTE  BR,SELA!SP1             ;SEE IF IN MAINT. MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR7     RA3                     ;BRANCH IF SO AND TREAT DLE LIKE NUM. MSG.
        MICPC=MICPC+1
        <JUMP!BR7CON!<RA3-INIT&3000*4>!<RA3-INIT&777/2>>
        BRWRTE  IMM,210                 ;MASK TO SET MAINT MODE AND DLE RECV'D
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<210>>
        SP      BR,AORB,SP1             ;SET THE BITS
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP1>
        ALWAYS  RA3                     ;TREAT LIKE NUMBERED MESSAGE
        MICPC=MICPC+1
        <JUMP!ALCOND!<RA3-INIT&3000*4>!<RA3-INIT&777/2>>
RESEXT: BRWRTE  IMM,4                   ;ADD TO MXT BITS
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<4>>
        SP      IBUS,NPR,SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!NPR!SPO>
        OUT     BR,ADD!ONPR             ;DO IT
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<ADD!ONPR>>
        ALWAYS  RES1
        MICPC=MICPC+1
        <JUMP!ALCOND!<RES1-INIT&3000*4>!<RES1-INIT&777/2>>
TABMXT: BRWRTE  IMM,4                   ;INCREMENT MXT
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<4>>
        SP      IBUS,UBBR,SPO           ;READ BR CONTROL
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!UBBR!SPO>
        OUT     BR,ADD!OBR
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<ADD!OBR>>
        ALWAYS  ECX
        MICPC=MICPC+1
        <JUMP!ALCOND!<ECX-INIT&3000*4>!<ECX-INIT&777/2>>
        $ZERO
        MICPC=MICPC+1
        000000
```

```
 1249                 014322
 1250                 000777
 1251
 1252
 1253  014322
  (1)                 001000
  (1)  014322  020620
 1254  014324
  (1)                 001001
  (1)  014324  173202
 1255  014326
  (1)                 001002
  (1)  014326  100452
```

```
                .=INIT+2000
                MICPC=777
                .SBTTL  TMTDA--TRANSMITTER DISPATCH ROUTINE
                ;
TMTDA:          BRWRTE  IBUS,TMTCON                     ;READ TRANMITTER CONTROL REGISTER
                MICPC=MICPC+1
                <MOVE!WRTEBR!IBUS!<TMTCON>>
                .BR4    DP,SELA,<2!PAGE2>               ;IF READY PROCEED
                MICPC=MICPC+1
                <JUMP! BR4CON!DP!SELA!2!PAGE2>
                ALWAYS  I1                              ;ELSE IDLE
                MICPC=MICPC+1
                <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
```

```
                                .SBTTL  TMTA--FIRST CHARACTER OF HEADER
                                ;
1257                            ;
1258                            ;
1259  014330                    TMTA:   BRWRTE  BR,AA!SP10                      ;SHIFT LEFT
 (1)                                    MICPC=MICPC+1
 (1)  014330  060530                    <MOVE!WRTEBR!BR!<AA!SP10>>
1260  014332                            BR7     RCVCK
 (1)                                    MICPC=MICPC+1
 (1)  014332  107614                    <JUMP!BR7CON!<RCVCK-INIT&3000*4>!<RCVCK-INIT&777/2>>
1261  014334                    TA1:    BRWRTE  BR,SELA!SP10                    ;REREAD STATUS
 (1)                                    MICPC=MICPC+1
 (1)  014334  060610                    <MOVE!WRTEBR!BR!<SELA!SP10>>
1262  014336                            BR0     NUMSYN                          ;IF UNNUMBPENDING  -- SEND IT
 (1)                                    MICPC=MICPC+1
 (1)  014336  112031                    <JUMP!BR0CON!<NUMSYN-INIT&3000*4>!<NUMSYN-INIT&777/2>>
1263  014340                            BRSHFT
 (1)                                    MICPC=MICPC+1
 (1)  014340  001620                    <MOVE!SHFTBR!WRTEBR!SELB>
1264  014342                            BR4     I1                              ;IF START MODE--EXIT
 (1)                                    MICPC=MICPC+1
 (1)  014342  103052                    <JUMP!BR4CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1265  014344                            BR1     NUMSYN                  ;IF LINE HAS GONE IDLE SEND SYN
 (1)                                    MICPC=MICPC+1
 (1)  014344  112431                    <JUMP!BR1CON!<NUMSYN-INIT&3000*4>!<NUMSYN-INIT&777/2>>
1266                                                                    ;ELSE--START TO SEND MESSAGE
1267  014346                    TMTEXT: TSTATE  TMTB
 (1)                                    MICPC=MICPC+1
 (1)  014346  000453                    <MOVE!WRTEBR!IMM!<TMTB-INIT&777/2>>
 (1)                                    MICPC=MICPC+1
 (1)  014350  063222                    <MOVE!SPX!BR!SELB!SP2>
1268  014352                            BRWRTE  BR,SELA!SP1                     ;ARE WE IN BOOT MODE
 (1)                                    MICPC=MICPC+1
 (1)  014352  060601                    <MOVE!WRTEBR!BR!<SELA!SP1>>
1269  014354                            BR7     TMTBT                   ;IF SO SEND DLE
 (1)                                    MICPC=MICPC+1
 (1)  014354  113427                    <JUMP!BR7CON!<TMTBT-INIT&3000*4>!<TMTBT-INIT&777/2>>
1270  014356                            BRWRTE  BR,<SELA!SP10>                  ;UNNUMB MESSGE?
 (1)                                    MICPC=MICPC+1
 (1)  014356  060610                    <MOVE!WRTEBR!BR!<SELA!SP10>>
1271  014360                            BR0     TMTUN                           ;IF SO --BRANCH
 (1)                                    MICPC=MICPC+1
 (1)  014360  112023                    <JUMP!BR0CON!<TMTUN-INIT&3000*4>!<TMTUN-INIT&777/2>>
1272  014362                            BRWRTE  IMM,201                         ;ELSE STORE SOH
 (1)                                    MICPC=MICPC+1
 (1)  014362  000601                    <MOVE!WRTEBR!IMM!<201>>
1273  014364                    TMTA5:  OUTPUT  BR,<SELB!TMTDAT>                ;IN TMT SILO
 (1)                                    MICPC=MICPC+1
 (1)  014364  062230                    <MOVE!WROUT!BR!<SELB!TMTDAT>>
1274  014366                            ALWAYS  I1
 (1)                                    MICPC=MICPC+1
 (1)  014366  100452                    <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1275  014370                    TMTUN:  TSTATE  TMTI
 (1)                                    MICPC=MICPC+1
 (1)  014370  000600                    <MOVE!WRTEBR!IMM!<TMTI-INIT&777/2>>
 (1)                                    MICPC=MICPC+1
 (1)  014372  063222                    <MOVE!SPX!BR!SELB!SP2>
1276  014374                            BRWRTE  IMM,5                           ;ENQ TO BR
```

```
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IMM!<5>>
                                        ALWAYS  TMTA5
                                        MICPC=MICPC+1
                                        <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
TMTBT:                  BRWRTE  IMM,220                 ;WRITE A DLE TO BR
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IMM!<220>>
                                        ALWAYS  TMTA5                   ;SEND IT
                                        MICPC=MICPC+1
                                        <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
;
NUMSYN:                 BRWRTE  BR,<SELA!SP10>                  ;READ LINE STATUS WORD
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!BR!<SELA!SP10>>
                                        BR7     5$                      ;IF OK TO SEND--PROCEED
                                        MICPC=MICPC+1
                                        <JUMP!BR7CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
                                        ALWAYS  I1                      ;ELSE--IDLE
                                        MICPC=MICPC+1
                                        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
5$:                     BRWRTE  IBUS,MODEM              ;ARE WE STILL SENDING?
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IBUS!<MODEM>>
                                        BRSHFT
                                        MICPC=MICPC+1
                                        <MOVE!SHFTBR!WRTEBR!SELB>
                                        BR4     I1                      ;RTS SET? IF SO WE ARE--STALL
                                        MICPC=MICPC+1
                                        <JUMP!BR4CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
                                        BRWRTE  IMM,373                 ;MASK TO TURN OFFLINE IDLE
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IMM!<373>>
                                        SP      BR,AANDB,SP10           ;IN LINE STATUS WORD
                                        MICPC=MICPC+1
                                        <MOVE!SPX!BR!AANDB!SP10>
                                        TSTATE  TMTA1
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IMM!<TMTA1-INIT&777/2>>
                                        MICPC=MICPC+1
                                        <MOVE!SPX!BR!SELB!SP2>
                                        BRWRTE  IMM,10
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IMM!<10>>
                                        SP      BR,SELB,SP6                     ;STORE IN SP6
                                        MICPC=MICPC+1
                                        <MOVE!SPX!BR!SELB!SP6>
TMTA1:                  SP      BR,DECA,SP6                     ;DECREMENT SYN COUNT
                                        MICPC=MICPC+1
                                        <MOVE!SPX!BR!DECA!SP6>
                                        Z       TMTEXT
                                        MICPC=MICPC+1
                                        <JUMP!ZCOND!<TMTEXT-INIT&3000*4>!<TMTEXT-INIT&777/2>>
                                        BRWRTE  IMM,1                           ;MASK FOR SOM
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IMM!<1>>
```

```
1300  014442
 (1)          001050
 (1)  014442  062231
1301  014444
 (1)          001051
 (1)  014444  000626
1302  014446
 (1)          001052
 (1)  014446  110421
        OUTPUT  BR,SELB!OTMTCO                  ;WRITE IT TO TMTP CONTROL
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!OTMTCO>>
        BRWRTE  IMM,226                         ;SYNC CHAR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<226>>
        ALWAYS  TMTA5
        MICPC=MICPC+1
        <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
```

```
                                .SBTTL  TMTB--OUTPUT FIRST CHAR OF COUNT
                                ;
014450                  TMTB:   SPBR    IBUS,NPR,SPO                    ;READ BR CONTROL REG
001053                          MICPC=MICPC+1
014450  123600                  <MOVE!SPBRX!IBUS!NPR!SPO>
014452                          BRQ     IDLE                    ;NPR GOING--GET OUT
001054                          MICPC=MICPC+1
014452  102045                  <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
014454                          LDMA    IMM,LTC                 ;GETPOINTER TO NEXT TMT LINK
001055                          MICPC=MICPC+1
014454  010071                  <MOVE!LDMAR!IMM!<LTC&377>>
014456                          LDMA    MEMX,SELB!SPX!SP16      ;POINT TO THE LINK
001056                          MICPC=MICPC+1
014456  053236                  <MOVE!LDMAR!MEMX!<SELB!SPX!SP16>>
014460                          MEMINC  IMM,3                   ;WRITE MSG TMTED TO FLAGS
001057                          MICPC=MICPC+1
014460  016403                  <MOVE!WRMEM!INCMAR!IMM!<3>>
014462                          MEMINC  DP,SELA!SP12            ;PICK UP MSGNO
001060                          MICPC=MICPC+1
014462  076612                  <MOVE!WRMEM!INCMAR!DP!<SELA!SP12>>
014464                          STATE   TMTC                    ;ADDRESS TMTR STATE
001061                          MICPC=MICPC+1
014464  000500                  <MOVE!WRTEBR!IMM!<TMTC-INIT&777/2>>
014466                  TBO:    SP      BR,SELB,SP2             ;UPDATE IT
001062                          MICPC=MICPC+1
014466  063222                  <MOVE!SPX!BR!SELB!SP2>
014470                          OUTPUT  <MEMX!INCMAR>,SELB!IBA1 ;WRITELOWBYTEOFADDRESS
001063                          MICPC=MICPC+1
014470  056224                  <MOVE!WROUT!MEMX!INCMAR!<SELB!IBA1>>
014472                          OUTPUT  <MEMX!INCMAR>,SELB!IBA2 ;WRITE HIGH BYTE OF ADDRESS
001064                          MICPC=MICPC+1
014472  056225                  <MOVE!WROUT!MEMX!INCMAR!<SELB!IBA2>>
014474                          SP      MEMX,SELB,SP7   ;HIGH BYTE OF COUNT TO SP7
001065                          MICPC=MICPC+1
014474  043227                  <MOVE!SPX!MEMX!SELB!SP7>
                                                        ;WAIT TO MASK OFF MEM EXT. BITS
014476                          SP      IMM,300,SPO             ;MASK FOR MXT
001066                          MICPC=MICPC+1
014476  003300                  <MOVE!SPX!IMM!300!SPO>
014500                          BRWRTE  MEMX!INCMAR,AANDB!SPO           ;TURN OFF CC2
001067                          MICPC=MICPC+1
014500  054660                  <MOVE!WRTEBR!MEMX!INCMAR!<AANDB!SPO>>
014502                          BRSHFT                          ;SHIFT BITS INTO CORRECT POSITION
001070                          MICPC=MICPC+1
014502  001620                  <MOVE!SHFTBR!WRTEBR!SELB>
014504                          BRSHFT
001071                          MICPC=MICPC+1
014504  001620                  <MOVE!SHFTBR!WRTEBR!SELB>
014506                          BRSHFT
001072                          MICPC=MICPC+1
014506  001620                  <MOVE!SHFTBR!WRTEBR!SELB>
014510                          BRSHFT
001073                          MICPC=MICPC+1
014510  001620                  <MOVE!SHFTBR!WRTEBR!SELB>
014512                          OUT     BR,SELB!ONPR
001074                          MICPC=MICPC+1
```

```
  (1)  014512  061230          <MOVE!WROUTX!BR!<SELB!ONPR>>
1325   014514                 SPBR    MEMX!INCMAR,SELB,SP6            ;LOWBYTE OF COUNT TO SP6
  (1)          001075         MICPC=MICPC+1
  (1)  014514  057626         <MOVE!SPBRX!MEMX!INCMAR!SELB!SP6>
1326   014516                 OUTPUT  BR,SELB!TMTDAT                  ;WRITE IT TO TMTR SILO
  (1)          001076         MICPC=MICPC+1
  (1)  014516  062230         <MOVE!WROUT!BR!<SELB!TMTDAT>>
1327   014520                 ALWAYS  IDLE
  (1)          001077         MICPC=MICPC+1
  (1)  014520  100445         <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1328                           ;
```

```
1330
1331
1332  014522                        .SBTTL  TMTC--OUTPUT SECOND CHAR OF COUNT
1333                                ;
1334  014522  001100  TMTC:   BRWRTE  IMM,77                    ;MASK TO CLEAR MXT BITS
(1)   014522  000477          MICPC=MICPC+1
(1)                           <MOVE!WRTEBR!IMM!<77>>
1335  014524                  SPBR    BR,AANDB,SP7              ;CLEAR THEM
(1)   014524  001101          MICPC=MICPC+1
(1)   014524  063667          <MOVE!SPBRX!BR!AANDB!SP7>
1336  014526                  OUTPUT  DP,<SELB!TMTDAT>          ;WRITE TO TMT SILO
(1)   014526  001102          MICPC=MICPC+1
(1)   014526  062230          <MOVE!WROUT!DP!<SELB!TMTDAT>>
1337  014530                  LDMA    IMM,LTC                   ;POINT TO TMT BUFFER
(1)   014530  001103          MICPC=MICPC+1
(1)   014530  010071          <MOVE!LDMAR!IMM!<LTC&377>>
1338  014532                  BRWRTE  IMM,6                     ;OFFSET TO NEXT LINK
(1)   014532  001104          MICPC=MICPC+1
(1)   014532  000406          <MOVE!WRTEBR!IMM!<6>>
1339  014534                  MEM     BR,ADD!SP16               ;UPDATE THE POINTER
(1)   014534  001105          MICPC=MICPC+1
(1)   014534  062416          <MOVE!WRMEM!BR!<ADD!SP16>>
1340  014536                  BRWRTE  IMM,TML8                          ;GET WRAPAROUND ADDRESS
(1)   014536  001106          MICPC=MICPC+1
(1)   014536  000545          <MOVE!WRTEBR!IMM!<TML8>>
1341  014540                  CMP     BR,SP16                   ;WRAPAORUND
(1)   014540  001107          MICPC=MICPC+1
(1)   014540  060376          <SUBTC!BR!SP16>
1342  014542                  Z       10$
(1)   014542  001110          MICPC=MICPC+1
(1)   014542  111513          <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1343  014544          5$:     STATE   TMTD
(1)   014544  001111          MICPC=MICPC+1
(1)   014544  000515          <MOVE!WRTEBR!IMM!<TMTD-INIT&777/2>>
1344  014546                  ALWAYS  XEXIT
(1)   014546  001112          MICPC=MICPC+1
(1)   014546  100451          <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
1345  014550          10$:    MEM     IMM,TML1                          ;GO BACK TO FIRST LINK
(1)   014550  001113          MICPC=MICPC+1
(1)   014550  002473          <MOVE!WRMEM!IMM!<TML1>>
1346  014552                  ALWAYS  5$
(1)   014552  001114          MICPC=MICPC+1
(1)   014552  110511          <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
1347                          ;
```

```
 1347
 1348  014554                          .SBTTL  TMTD--RESPONSE FIELD-NUMBERED MESSAGE
TMTD:  014554                  STATE   TMTE
(1)             001115                  MICPC=MICPC+1
(1)     014554  000523                  <MOVE!WRTEBR!IMM!<TMTE-INIT&777/2>>
 1349  014556                          SP      BR,DECA,SP6                     ;ADJUSRT COUNT FOR  TWO'S  COMPLEMENT
(1)             001116                  MICPC=MICPC+1
(1)     014556  063166                  <MOVE!SPX!BR!DECA!SP6>
 1350  014560                          C       TD2                             ;NO OVERFLOW
(1)             001117                  MICPC=MICPC+1
(1)     014560  111121                  <JUMP!CCOND!<TD2-INIT&3000*4>!<TD2-INIT&777/2>>
 1351  014562                          SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF COUNT
(1)             001120                  MICPC=MICPC+1
(1)     014562  063167                  <MOVE!SPX!BR!DECA!SP7>
 1352  014564  010171  TD2:     LDMA    IMM,ISP11                       ;RESP FIELD ADDR TO MAR
(1)             001121                  MICPC=MICPC+1
(1)     014564                          <MOVE!LDMAR!IMM!<ISP11&377>>
```

```
 1363  014566                          ALWAYS  TJ1
  (1)          001122                 MICPC=MICPC+1
  (1)  014566  110606                 <JUMP!ALCOND!<TJ1-INIT&3000*4>!<TJ1-INIT&777/2>>
```

```
 1367
 1368  014570
 1369  014570
  (1)              001123
  (1)  014570      123600
 1370  014572
  (1)              001124
  (1)  014572      102052
 1371  014574
  (1)              001125
  (1)  014574      060612
 1372  014576
  (1)              001126
  (1)  014576      062230
 1373  014600
  (1)              001127
  (1)  014600      000531
 1374  014602
  (1)              001130
  (1)  014602      110573
```

```
        .SBTTL  TMTE--NUMBER FIELD--NUMBERED MESSAGE
TMTE:
        SPBR    IBUS,NPR,SPO            ;READ NPR CONTROL REGISTER
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BRO     I1                      ;BUSY - GET OUT
        MICPC=MICPC+1
        <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
        BRWRTE  BR,SELA!SP12
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP12>>
        OUTPUT  BR,<SELB!TMTDAT>                ;WRITE IT TO THE SILO
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!TMTDAT>>
        STATE   TMTF
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTF-INIT&777/2>>
        ALWAYS  TH3
        MICPC=MICPC+1
        <JUMP!ALCOND!<TH3-INIT&3000*4>!<TH3-INIT&777/2>>
```

```
1376
1377
1378  014604                        .SBTTL  TMTF--NUMBERED MSG ADDRESS FIELD
                                    ;
                    TMTF:   STATE   TF1
                            MICPC=MICPC+1
 (1)        001131
 (1)  014604  000535               <MOVE!WRTEBR!IMM!<TF1-INIT&777/2>>
1379  014606                        TF2:    SP      BR,SELB,SP2
                            MICPC=MICPC+1
 (1)        001132
 (1)  014606  063222               <MOVE!SPX!BR!SELB!SP2>
1380  014610                        BRWRTE  IMM,1                   ;LOAD ADDRESS
                            MICPC=MICPC+1
 (1)        001133
 (1)  014610  000401               <MOVE!WRTEBR!IMM!<1>>
1386  014612                        ALWAYS  TMTA5
                            MICPC=MICPC+1
 (1)        001134
 (1)  014612  110421               <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
```

```
 1389
 1390  014614
  (1)                 001135
  (1)  014614         000402
 1391  014616
  (1)                 001136
  (1)  014616         062231
 1392  014620
  (1)                 001137
  (1)  014620         062230
 1393  014622
  (1)                 001140
  (1)  014622         020500
 1394  014624
  (1)                 001141
  (1)  014624         112155
 1395  014626
  (1)                 001142
  (1)  014626         000544
 1396  014630
  (1)                 001143
  (1)  014630         100451
```

```
                .SBTTL  TF1-NUMBERED MSG HEADER EOM
TF1:            BRWRTE  IMM,2                           ;EOM  MASK TO BR
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<2>>
                OUTPUT  BR,<SELB!OTMTCO>                ;UPDATE TMTR CONTROL REGISTER
                MICPC=MICPC+1
                <MOVE!WROUT!BR!<SELB!OTMTCO>>
                OUTPUT  BR,<SELB!TMTDAT>                ;OUTPUT A GARBAGE CHAR
                MICPC=MICPC+1
                <MOVE!WROUT!BR!<SELB!TMTDAT>>
                BRWRTE  IBUS,IIBA1                      ;READ LOW ORDER FROM INBA
                MICPC=MICPC+1
                <MOVE!WRTEBR!IBUS!<IIBA1>>
                BRO     TMTF1                           ;IF ODD BYTE--BRANCH
                MICPC=MICPC+1
                <JUMP!BROCON!<TMTF1-INIT&3000*4>!<TMTF1-INIT&777/2>>
                STATE   TMTH
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<TMTH-INIT&777/2>>
                ALWAYS  XEXIT
                MICPC=MICPC+1
                <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
```

```
1398
1399
1400
1401  014632
 (1)              001144
 (1)  014632      123600
1405  014634
 (1)              001145
 (1)  014634      102052
1406  014636
 (1)              001146
 (1)  014636      022010
1407  014640
 (1)              001147
 (1)  014640      023100
1408  014642
 (1)              001150
 (1)  014642      062064
1409  014644
 (1)              001151
 (1)  014644      063166
1410  014646
 (1)              001152
 (1)  014646      111155
1411  014650
 (1)              001153
 (1)  014650      063167
1412  014652
 (1)              001154
 (1)  014652      115402
1413  014654
1418  014654
 (1)              001155
 (1)  014654      000557
1419  014656
 (1)              001156
 (1)  014656      100451
1420  014660
1422  014660
 (1)              001157
 (1)  014660      123600
1423  014662
 (1)              001160
 (1)  014662      102052
1425  014664
 (1)              001161
 (1)  014664      022030
1426  014666
 (1)              001162
 (1)  014666      023100
1427  014670
 (1)              001163
 (1)  014670      062064
1428  014672
 (1)              001164
 (1)  014672      111372
```

```
;************TIME CRITICAL PATH--MODIFY WITH GREAT CARE
        .SBTTL  TMTH--ROUTINE TO OUTPUT DATA CHARACTERS
        ;
TMTH:   SPBR    IBUS,NPR,SPO                    ;READ NPR CONTROL
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BRO     I1                              ;IF NPR IN PROGRESS --BRANCH
        MICPC=MICPC+1
        <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
5$:     OUTPUT  IBUS,<INDAT1!TMTDAT>            ;WRITE THE EVEN CHAR TO TMT SILO
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<INDAT1!TMTDAT>>
        SP      IBUS,IIBA1,SPO                  ;READ LOW BYTE OF BA TO SP
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IIBA1!SPO>
        OUTPUT  BR,<INCA!IBA1>                  ;OUTPUT INCREMENTED BA
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<INCA!IBA1>>
        SP      BR,DECA,SP6                     ;DECREMENT CHARACTER  COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP6>
        C       TH6                             ;NO OVERFLOW
        MICPC=MICPC+1
        <JUMP!CCOND!<TH6-INIT&3000*4>!<TH6-INIT&777/2>>
        SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP7>
        Z       HEH1                            ;BYTE  COUNT ZERO
        MICPC=MICPC+1
        <JUMP!ZCOND!<HEH1-INIT&3000*4>!<HEH1-INIT&777/2>>
TH6:
TMTF1:  STATE   TMTH0
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTH0-INIT&777/2>>
        ALWAYS  XEXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
TMTH0:
        SPBR    IBUS,NPR,SPO                    ;NPR BUSY
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BRO     I1
        MICPC=MICPC+1
        <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
TH9:    OUTPUT  IBUS,<INDAT2!TMTDAT>            ;ODD CHAR TO SILO
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<INDAT2!TMTDAT>>
        SP      IBUS,IIBA1,SPO                  ;READ LOW BYTE TO BA
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IIBA1!SPO>
        OUTPUT  BR,<INCA!IBA1>                  ;OUTPUT THE INCREMENTED BA
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<INCA!IBA1>>
        C       HOINCH
        MICPC=MICPC+1
        <JUMP!CCOND!<HOINCH-INIT&3000*4>!<HOINCH-INIT&777/2>>
```

```
1429  014674
 (1)              001165
 (1)  014674  063166
1430  014676
 (1)              001166
 (1)  014676  111171
1431  014700
 (1)              001167
 (1)  014700  063167
1432  014702
 (1)              001170
 (1)  014702  115402
1433  014704
 (1)              001171
 (1)  014704  123600
1437  014706
 (1)              001172
 (1)  014706  000544
1438  014710
 (1)              001173
 (1)  014710  063222
1439  014712
 (1)              001174
 (1)  014712  000557
1440  014714
 (1)              001175
 (1)  014714  063260
1441  014716
 (1)              001176
 (1)  014716  000401
1449  014720
 (1)              001177
 (1)  014720  104643
1451
1452
```

```
TH8:    SP      BR,DECA,SP6                     ;DECREMENT CHARACTERCOUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP6>
        C       TH7                             ;NO OVERFLOW
        MICPC=MICPC+1
        <JUMP!CCOND!<TH7-INIT&3000*4>!<TH7-INIT&777/2>>
        SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF  COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP7>
        Z       HEH1                            ;BYTE COUNT ZERO
        MICPC=MICPC+1
        <JUMP!ZCOND!<HEH1-INIT&3000*4>!<HEH1-INIT&777/2>>
TH7:    SPBR    IBUS,NPR,SP0                    ;READ NPR REGISTER
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SP0>
        STATE   TMTH
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTH-INIT&777/2>>
TH3:    SP      BR,SELB,SP2                     ;SAVE TSTATE
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
TH3X:   BRWRTE  IMM,157                         ;CLEAR C0 AND C1
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<157>>
        SP      BR,AANDB,SP0                    ;CLEAR THE BITS
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP0>
        BRWRTE  IMM,1                           ;WRITE  NPR BITS TO BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<1>>
        ALWAYS  RK7
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK7-INIT&3000*4>!<RK7-INIT&777/2>>
;***************END TIME CRITICAL PATH*********************************
        ;
```

```
1454
1455  014722                    .SBTTL  TMTI--SEND UNNUMBERED TYPE FIELD
 (1)          001200
 (1)  014722  010153  TMTI:    LDMA    IMM,T                   ;ADDRESS OF TYPE FIELD TO MAR
                               MICPC=MICPC+1
                               <MOVE!LDMAR!IMM!<T&377>>
1456  014724                   SP      MEMX,SELB,SP6           ;COPY IT TO SP6
 (1)          001201
 (1)  014724  043226           MICPC=MICPC+1
                               <MOVE!SPX!MEMX!SELB!SP6>
1457  014726                   STATE   TMTJ
 (1)          001202
 (1)  014726  000604           MICPC=MICPC+1
                               <MOVE!WRTEBR!IMM!<TMTJ-INIT&777/2>>
1458  014730                   ALWAYS  TJ1
 (1)          001203
 (1)  014730  110606           MICPC=MICPC+1
                               <JUMP!ALCOND!<TJ1-INIT&3000*4>!<TJ1-INIT&777/2>>
1459                           ;
1460                           .SBTTL  TMTJ--SEND SUB-TYPE FIELD
1461  014732          TMTJ:    LDMA    IMM,ST                  ;ADDRESS OF SUB-TYPE FIELD TO MAR
 (1)          001204           MICPC=MICPC+1
 (1)  014732  010154           <MOVE!LDMAR!IMM!<ST&377>>
1462  014734                   STATE   TMTK
 (1)          001205           MICPC=MICPC+1
 (1)  014734  000610           <MOVE!WRTEBR!IMM!<TMTK-INIT&777/2>>
1463  014736          TJ1:     OUTPUT  MEMX,SELB!TMTDAT
 (1)          001206           MICPC=MICPC+1
 (1)  014736  042230           <MOVE!WROUT!MEMX!<SELB!TMTDAT>>
1464  014740                   ALWAYS  XEXIT
 (1)          001207           MICPC=MICPC+1
 (1)  014740  100451           <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
1465                           ;
```

```
                        .SBTTL  TMTK--OUTPUT RESPONSE FIELD (UNNUMB MSG)
                        ;
1467                    ;
1468                    ;
1469  014742            TMTK:   BRWRTE  IMM,3                           ;WRITE A 3 TO BR
 (1)            001210          MICPC=MICPC+1
 (1)  014742    000403          <MOVE!WRTEBR!IMM!<3>>
1470  014744                    NOP     BR,SUB,SP6                      ;IF TYPE LESS THAN 3
 (1)            001211          MICPC=MICPC+1
 (1)  014744    060346          <BR!SUB!SP6>
1471  014746                    TSTATE  TMTL
 (1)            001212          MICPC=MICPC+1
 (1)  014746    000616          <MOVE!WRTEBR!IMM!<TMTL-INIT&777/2>>
 (1)            001213          MICPC=MICPC+1
 (1)  014750    063222          <MOVE!SPX!BR!SELB!SP2>
1472  014752                    C       TMTLO
 (1)            001214          MICPC=MICPC+1
 (1)  014752    111223          <JUMP!CCOND!<TMTLO-INIT&3000*4>!<TMTLO-INIT&777/2>>
1473  014754                    ALWAYS  TD2
 (1)            001215          MICPC=MICPC+1
 (1)  014754    110521          <JUMP!ALCOND!<TD2-INIT&3000*4>!<TD2-INIT&777/2>>
1474                    ;
1475                    ;       .SBTTL  TMTL--UNNUMB MSG NUMBER FIELD
1476  014756            TMTL:   TSTATE  TMTM
 (1)            001216          MICPC=MICPC+1
 (1)  014756    000627          <MOVE!WRTEBR!IMM!<TMTM-INIT&777/2>>
 (1)            001217          MICPC=MICPC+1
 (1)  014760    063222          <MOVE!SPX!BR!SELB!SP2>
1477  014762                    BRWRTE  IMM,3
 (1)            001220          MICPC=MICPC+1
 (1)  014762    000403          <MOVE!WRTEBR!IMM!<3>>
1478  014764                    CMP     BR,SP6                          ;IS MESSAGE REP
 (1)            001221          MICPC=MICPC+1
 (1)  014764    060366          <SUBTC!BR!SP6>
1479  014766                    Z       TMTL1                           ;YES
 (1)            001222          MICPC=MICPC+1
 (1)  014766    111625          <JUMP!ZCOND!<TMTL1-INIT&3000*4>!<TMTL1-INIT&777/2>>
1480  014770            TMTLO:  BRWRTE  IMM,0                           ;ADDRESS CONTNAT OF ZERO
 (1)            001223          MICPC=MICPC+1
 (1)  014770    000400          <MOVE!WRTEBR!IMM!<0>>
1481  014772                    ALWAYS  TMTA5
 (1)            001224          MICPC=MICPC+1
 (1)  014772    110421          <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
1482                    ;
1483  014774            TMTL1:  BRWRTE  BR,DECA!SP12                    ;WRITE A RESPONSE
 (1)            001225          MICPC=MICPC+1
 (1)  014774    060572          <MOVE!WRTEBR!BR!<DECA!SP12>>
1489  014776                    ALWAYS  TMTA5
 (1)            001226          MICPC=MICPC+1
 (1)  014776    110421          <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
1491                    ;
```

```
                                .SBTTL  TMTM--UNNUMB MSG--STATION ADDRESS
1493
1494  015000                    TMTM:   STATE   TNEOM
 (1)          001227                    MICPC=MICPC+1
 (1)  015000  000631                    <MOVE!WRTEBR!IMM!<TNEOM-INIT&777/2>>
1495  015002                            ALWAYS  TF2
 (1)          001230                    MICPC=MICPC+1
 (1)  015002  110532                    <JUMP!ALCOND!<TF2-INIT&3000*4>!<TF2-INIT&777/2>>
1496  015004                    TNEOM:  BRWRTE  IMM,2                   ;END OF MESSAGE TO BR
 (1)          001231                    MICPC=MICPC+1
 (1)  015004  000402                    <MOVE!WRTEBR!IMM!<2>>
1497  015006                            OUTPUT  BR,<SELB!OTMTCO>
 (1)          001232                    MICPC=MICPC+1
 (1)  015006  062231                    <MOVE!WROUT!BR!<SELB!OTMTCO>>
1498  015010                            OUTPUT  BR,<SELB!TMTDAT>              ;OUTPUT A GARBAGE CHARACTER
 (1)          001233                    MICPC=MICPC+1
 (1)  015010  062230                    <MOVE!WROUT!BR!<SELB!TMTDAT>>
1499  015012                            BRWRTE  IMM,4                         ;SET UP LINE HAS GONE IDLE MASK
 (1)          001234                    MICPC=MICPC+1
 (1)  015012  000404                    <MOVE!WRTEBR!IMM!<4>>
1500  015014                            SPBR    BR,AORB,SP10                  ;UPDATE LINE STATUS WORD
 (1)          001235                    MICPC=MICPC+1
 (1)  015014  063710                    <MOVE!SPBRX!BR!AORB!SP10>
1501  015016                            BRWRTE  BR,AA!SP10                    ;SHIFT STATUS LEFT
 (1)          001236                    MICPC=MICPC+1
 (1)  015016  060530                    <MOVE!WRTEBR!BR!<AA!SP10>>
1502  015020                            BR7     10$                           ;IF HDX SET---BRANCH TO CLEAR OK TO SEND
 (1)          001237                    MICPC=MICPC+1
 (1)  015020  113643                    <JUMP!BR7CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1503  015022                            BRWRTE  IMM,376                       ;MASK TO TURN OFF UNNUMB PENDDING
 (1)          001240                    MICPC=MICPC+1
 (1)  015022  000776                    <MOVE!WRTEBR!IMM!<376>>
1504  015024                    5$:     SP      BR,AANDB,SP10                 ;MASK TO LINE STATUS WORD
 (1)          001241                    MICPC=MICPC+1
 (1)  015024  063270                    <MOVE!SPX!BR!AANDB!SP10>
1505  015026                            ALWAYS  TEOM2
 (1)          001242                    MICPC=MICPC+1
 (1)  015026  110734                    <JUMP!ALCOND!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
1506  015030                    10$:    BRWRTE  IMM,176                       ;CLEAR OK TO SEND AND UNNUMB PENPENDING
 (1)          001243                    MICPC=MICPC+1
 (1)  015030  000576                    <MOVE!WRTEBR!IMM!<176>>
1507  015032                            ALWAYS  5$
 (1)          001244                    MICPC=MICPC+1
 (1)  015032  110641                    <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
```

```
                        .SBTTL  TIMSRV--TIMEOUT ROUTINE--SENDS  REP
                        ;
                        .ENABLE LSB
1509                    
1510                    
1511                    
1512  015034            TIMSRV: SP      IBUS,UBBR,SP0                   ;READ UNIBUS BR REGISTER
 (1)          001245            MICPC=MICPC+1
 (1)  015034  123220            <MOVE!SPX!IBUS!UBBR!SP0>
1513  015036                    BRWRTE  IMM,177                         ;MASK OFF  BR REQ
 (1)          001246            MICPC=MICPC+1
 (1)  015036  000577            <MOVE!WRTEBR!IMM!<177>>
1514                                                                    ;RESET TIMER---SLICK MOVE
1515                                                                    ;SINCE TIMER IS RESET BY WRITING
1516                                                                    ;A 1 AND THE EXPIRATION  LOOKS
1517                                                                    ;LIKE 1--VOILA
1518  015040                    OUT     BR,<AANDB!OBR>                  ;AND THE BIT ON
 (1)          001247            MICPC=MICPC+1
 (1)  015040  061271            <MOVE!WROUTX!BR!<AANDB!OBR>>
1519  015042                    BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
 (1)          001250            MICPC=MICPC+1
 (1)  015042  060601            <MOVE!WRTEBR!BR!<SELA!SP1>>
1520  015044                    BR7     IDLE                            ;IF IN MAINT. MODE DISABLE TIMER
 (1)          001251            MICPC=MICPC+1
 (1)  015044  103445            <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1521  015046                    SP      BR,DECA,SP15                    ;DECREMENT THE COUNTER
 (1)          001252            MICPC=MICPC+1
 (1)  015046  063175            <MOVE!SPX!BR!DECA!SP15>
1522  015050                    Z       20$                             ;IF ALL ONES HAS EXPIRED
 (1)          001253            MICPC=MICPC+1
 (1)  015050  111662            <JUMP!ZCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
1523  015052            10$:    BRWRTE  BR,SELA!SP10                    ;READ LINE STATUS
 (1)          001254            MICPC=MICPC+1
 (1)  015052  060610            <MOVE!WRTEBR!BR!<SELA!SP10>>
1524  015054                    BR1     TABUPD                          ;NUMBERED MESSAGE IN PROGRESS
 (1)          001255            MICPC=MICPC+1
 (1)  015054  116737            <JUMP!BR1CON!<TABUPD-INIT&3000*4>!<TABUPD-INIT&777/2>>
1525  015056                    BR0     TABUPD                          ;UNNUMBMSGIN PROGRESS
 (1)          001256            MICPC=MICPC+1
 (1)  015056  116337            <JUMP!BR0CON!<TABUPD-INIT&3000*4>!<TABUPD-INIT&777/2>>
1526  015060                    BRSHFT
 (1)          001257            MICPC=MICPC+1
 (1)  015060  001620            <MOVE!SHFTBR!WRTEBR!SELB>
1527  015062                    BR4     IDLE                            ;START MODE
 (1)          001260            MICPC=MICPC+1
 (1)  015062  103045            <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1528  015064                    ALWAYS  SNDACK                          ;ELSE SEND ACK
 (1)          001261            MICPC=MICPC+1
 (1)  015064  110727            <JUMP!ALCOND!<SNDACK-INIT&3000*4>!<SNDACK-INIT&777/2>>
1529  015066            TIME1:
1530  015066            20$:    BRWRTE  IMM,2                           ;
 (1)          001262            MICPC=MICPC+1
 (1)  015066  000402            <MOVE!WRTEBR!IMM!<2>>
1531  015070                    SP      BR,SELB,SP15                    ;RESET THE TIMER TICK COUNT
 (1)          001263            MICPC=MICPC+1
 (1)  015070  063235            <MOVE!SPX!BR!SELB!SP15>
1532  015072                    BRWRTE  DP,<SELA!SP10>                  ;READ LINE STATUS WORD
 (1)          001264            MICPC=MICPC+1
 (1)  015072  060610            <MOVE!WRTEBR!DP!<SELA!SP10>>
```

```
1533  015074
 (1)              001265
 (1)  015074      001620
1534  015076
 (1)              001266
 (1)  015076      103076
1535  015100
 (1)              001267
 (1)  015100      060572
1536  015102
 (1)              001270
 (1)  015102      060377
1537  015104
 (1)              001271
 (1)  015104      111654
1538  015106
 (1)              001272
 (1)  015106      010153
1539  015110
 (1)              001273
 (1)  015110      016403
1540  015112
 (1)              001274
 (1)  015112      016700
1541  015114
 (1)              001275
 (1)  015114      010015
1542  015116
 (1)              001276
 (1)  015116      043220
1543  015120
 (1)              001277
 (1)  015120      062460
1544  015122
 (1)              001300
 (1)  015122      010003
1545  015124
 (1)              001301
 (1)  015124      040620
1546  015126
 (1)              001302
 (1)  015126      061620
1547  015130
 (1)              001303
 (1)  015130      062620
1548  015132
 (1)              001304
 (1)  015132      102353
1549  015134
 (1)              001305
 (1)  015134      000601
1550  015136
 (1)              001306
 (1)  015136      063310
1551  015140
 (1)              001307
```

```
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     BS1                             ;IF IN START MODE--BRANCH
        MICPC=MICPC+1
        <JUMP!BR4CON!<BS1-INIT&3000*4>!<BS1-INIT&777/2>>
BRWRTE  BR,DECA!SP12                            ;GET LAST NUMBER SENT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<DECA!SP12>>
        CMP     BR,SP17                         ;COMPARE TO LAST ACKED
        MICPC=MICPC+1
        <SUBTC!BR!SP17>
        Z       10$                     ;IF EQ --SEND ACK
        MICPC=MICPC+1
        <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
TIME2:  LDMA    IMM,T                           ;LOAD ADDRESS OF TYPE FIELD IN UNNUMB SK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,3                           ;LOAD REP TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<3>>
        MEMINC  IMM,300                         ;ZERO THE SUB-TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<300>>
        LDMA    IMM,REPCS                       ;CUMULATIVE REPS RECD
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<REPCS&377>>
        SP      MEMX,SELB,SP0                   ;COPY IT TO SP0
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP0>
        MEM     BR,INCA!SP0                     ;INCREMENT IT
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<INCA!SP0>>
        LDMA    IMM,REPST                       ;ADDRESS DYNAMIC REP COUNTER
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<REPST&377>>
        BRWRTE  MEMX,SELB                       ;COPY IT TO THE BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!<SELB>>
        BSHFTB
        MICPC=MICPC+1
        <MOVE!SHFTBR!SELB!BR>
        MEM     BR,SELB
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<SELB>>
        BRO     RTHRES
        MICPC=MICPC+1
        <JUMP!BROCON!<RTHRES-INIT&3000*4>!<RTHRES-INIT&777/2>>
        BRWRTE  IMM,201                 ;MASK FOR OK TO SEND
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<201>>
        SP      BR,AORB,SP10            ;OR IT IN
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        ALWAYS  IDLE
        MICPC=MICPC+1
```

```
   (1)  015140  100445
 1552
 1553
                <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
                .DSABLE LSB
                ;
```

```
1555  015142
 (1)            001310
 (1)  015142    120620
1556  015144
 (1)            001311
 (1)  015144    116063
1557  015146
 (1)            001312
 (1)  015146    000402
1558  015150
 (1)            001313
 (1)  015150    062231
1559  015152
 (1)            001314
 (1)  015152    062230
1560  015154
 (1)            001315
 (1)  015154    060601
1561  015156
 (1)            001316
 (1)  015156    113755
1562  015160
 (1)            001317
 (1)  015160    063072
1563  015162
 (1)            001320
 (1)  015162    010071
1564  015164
 (1)            001321
 (1)  015164    050220
1565  015166
 (1)            001322
 (1)  015166    040620
1566  015170
 (1)            001323
 (1)  015170    112334
1567  015172
 (1)            001324
 (1)  015172    000775
1568  015174
 (1)            001325
 (1)  015174    063670
1569  015176
 (1)            001326
 (1)  015176    112334
1570
1571  015200
 (1)            001327
 (1)  015200    010153
1572  015202
 (1)            001330
 (1)  015202    016401
1573  015204
 (1)            001331
 (1)  015204    000405
1574  015206

TEOM:   BRWRTE  IBUS,UBBR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<UBBR>>
        BRQ     NXMERR                          ;NON-EXISTANT MEMORY
        MICPC=MICPC+1
        <JUMP!BROCON!<NXMERR-INIT&3000*4>!<NXMERR-INIT&777/2>>
        BRWRTE  IMM,2                           ;EOM TO BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        OUTPUT  BR,<SELB!OTMTCO>                ;WRITE TMTR CONTROL
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!OTMTCO>>
        OUTPUT  BR,<SELB!TMTDAT>                ;WRITE GARBAGE DATA
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!TMTDAT>>
        BRWRTE  BR,SELA!SP1                     ;CKECK FOR BOOT MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR7     BTEOM                           ;---IF SET IS MAINT MSG
        MICPC=MICPC+1
        <JUMP!BR7CON!<BTEOM-INIT&3000*4>!<BTEOM-INIT&777/2>>
        SP      BR,INCA,SP12                    ;INCREMENT THE MESSAGE NUMBER
        MICPC=MICPC+1
        <MOVE!SPX!BR!INCA!SP12>
TEOM1:  LDMA    IMM,LTC                         ;ADDRESS LAST TMT LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<LTC&377>>
        LDMA    MEMX,SELB
        MICPC=MICPC+1
        <MOVE!LDMAR!MEMX!<SELB>>
        BRWRTE  MEMX,SELB
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!<SELB>>
        BRQ     TEOM2
        MICPC=MICPC+1
        <JUMP!BROCON!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
TEOM3:  BRWRTE  IMM,375                         ;TURN OFF MESSAGE PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<375>>
        SPBR    BR,AANDB,SP10                   ;
        MICPC=MICPC+1
        <MOVE!SPBRX!BR!AANDB!SP10>
        BRQ     TEOM2                           ;IF UNNUMB PENDING--GO AWAY
        MICPC=MICPC+1
        <JUMP!BROCON!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
        .SBTTL  SNDACK--ROUTINE TO SEND AN ACK
SNDACK: LDMA    IMM,T
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,1
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<1>>
        BRWRTE  IMM,5
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<5>>
SA2:    MEMINC  IMM,300
```

```
  (1)           001332
  (1)  015206   016700          MICPC=MICPC+1
                               <MOVE!WRMEM!INCMAR!IMM!<300>>
1575   015210                  SA3:    SP      BR,AORB,SP10
  (1)           001333          MICPC=MICPC+1
  (1)  015210   063310          <MOVE!SPX!BR!AORB!SP10>
1576                           .
1577   015212                  TEOM2:  STATE   TMTA
  (1)           001334          MICPC=MICPC+1
  (1)  015212   000403          <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
1578   015214                  ALWAYS  XEXIT
  (1)           001335          MICPC=MICPC+1
  (1)  015214   100451          <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
1579   015216                  FUDGE:  BRWRTE  IBUS,NPR                ;READ NPR CONTROL
  (1)           001336          MICPC=MICPC+1
  (1)  015216   120600          <MOVE!WRTEBR!IBUS!<NPR>>
1580   015220                  BRO     IDLE                    ;IF NPR GOING---LEAVE
  (1)           001337          MICPC=MICPC+1
  (1)  015220   102045          <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1581   015222                  BRWRTE  BR!LDMAR,SELA!SP4       ;LOAD THE MAR
  (1)           001340          MICPC=MICPC+1
  (1)  015222   070604          <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
1582   015224                  BR7     BS2                     ;IF SET - READ BACK ALL 200
  (1)           001341          MICPC=MICPC+1
  (1)  015224   103505          <JUMP!BR7CON!<BS2-INIT&3000*4>!<BS2-INIT&777/2>>
1583   015226                  MEMINC  IBUS,INDAT1             ;OTHERWISE RESTORE TWO BYTES
  (1)           001342          MICPC=MICPC+1
  (1)  015226   036400          <MOVE!WRMEM!INCMAR!IBUS!<INDAT1>>
1584   015230                  MEMINC  IBUS,INDAT2             ;..
  (1)           001343          MICPC=MICPC+1
  (1)  015230   036420          <MOVE!WRMEM!INCMAR!IBUS!<INDAT2>>
1585   015232                  BRWRTE  IMM,2                   ;UPDATE---UNIBUS ADDRESS
  (1)           001344          MICPC=MICPC+1
  (1)  015232   000402          <MOVE!WRTEBR!IMM!<2>>
1586   015234                  SP      BR,ADD,SP4              ;UPDATE NPR COUNTER
  (1)           001345          MICPC=MICPC+1
  (1)  015234   063004          <MOVE!SPX!BR!ADD!SP4>
1587   015236                  SP      IBUS,IIBA1,SPO          ;UPDATE ADDRESS LOW
  (1)           001346          MICPC=MICPC+1
  (1)  015236   023100          <MOVE!SPX!IBUS!IIBA1!SPO>
1588   015240                  OUTPUT  BR,ADD!IBA1
  (1)           001347          MICPC=MICPC+1
  (1)  015240   062004          <MOVE!WROUT!BR!<ADD!IBA1>>
1589   015242                  SP      IBUS,IIBA2,SPO          ;READ HIGH ADDRESS
  (1)           001350          MICPC=MICPC+1
  (1)  015242   023120          <MOVE!SPX!IBUS!IIBA2!SPO>
1590   015244                  OUTPUT  BR,AC!IBA2              ;UPDATE HIGH
  (1)           001351          MICPC=MICPC+1
  (1)  015244   062105          <MOVE!WROUT!BR!<AC!IBA2>>
1591   015246                  C       RESEXT                  ;IF CARRY---UPDATE MXT
  (1)           001352          MICPC=MICPC+1
  (1)  015246   105367          <JUMP!CCOND!<RESEXT-INIT&3000*4>!<RESEXT-INIT&777/2>>
1592   015250                  RES1:   SP      IBUS,NPR,SPO            ;READ NPR REGISTER
  (1)           001353          MICPC=MICPC+1
  (1)  015250   123200          <MOVE!SPX!IBUS!NPR!SPO>
1593   015252                  ALWAYS  TH3X                    ;GO DO ANOTHER NPR
  (1)           001354          MICPC=MICPC+1
```

```
SNDACK--ROUTINE TO SEND AN ACK

                <JUMP!ALCOND!<TH3X-INIT&3000*4>!<TH3X-INIT&777/2>>
  (1)  015252  110574
BTEOM:  BRWRTE  IMM,374                 ;MASK FOR CLEAR MSG PENDING
1594  015254
  (1)           001355
                MICPC=MICPC+1
  (1)  015254  000774
                <MOVE!WRTEBR!IMM!<374>>
1595  015256
                SP      BR,AANDB,SP10   ;TURN THEM OFF IN LINE STATUS WORD
  (1)           001356
                MICPC=MICPC+1
  (1)  015256  063270
                <MOVE!SPX!BR!AANDB!SP10>
1596  015260
                SP      BR,SELB,SP13    ;STORE UNRECOGNIZABLE VALUE INTO SP13
  (1)           001357
                MICPC=MICPC+1
  (1)  015260  063233
                <MOVE!SPX!BR!SELB!SP13>
1597
                                        ;SO "RH3" WILL EXIT BACK TO IDLE LOOP
1598  015262
                LDMA    IMM,STC         ;ADDRESS START OF TMT  CHAIN
  (1)           001360
                MICPC=MICPC+1
  (1)  015262  010070
                <MOVE!LDMAR!IMM!<STC&377>>
1599  015264
                SP      MEMX,SELB,SP0   ;COPY LINK ADDRESS
  (1)           001361
                MICPC=MICPC+1
  (1)  015264  043220
                <MOVE!SPX!MEMX!SELB!SP0>
1600  015266
                TSTATE  NUMSYN          ;CHANGE XMIT STATE TO LINE IS IDLE
  (1)           001362
                MICPC=MICPC+1
  (1)  015266  000431
                <MOVE!WRTEBR!IMM!<NUMSYN-INIT&777/2>>
  (1)           001363
                MICPC=MICPC+1
  (1)  015270  063222
                <MOVE!SPX!BR!SELB!SP2>
1601  015272
                ALWAYS  TDON2           ;POST A DONE
  (1)           001364
                MICPC=MICPC+1
  (1)  015272  114534
                <JUMP!ALCOND!<TDON2-INIT&3000*4>!<TDON2-INIT&777/2>>
1602  015274
RL4:    RSTATE  RCVL
  (1)           001365
                MICPC=MICPC+1
  (1)  015274  000715
                <MOVE!WRTEBR!IMM!<RCVL-INIT&777/2>>
  (1)           001366
                MICPC=MICPC+1
  (1)  015276  063223
                <MOVE!SPX!BR!SELB!SP3>
1603  015300
                SP      IBUS,NPR,SP0    ;READ NPR CONTROL REGISTER
  (1)           001367
                MICPC=MICPC+1
  (1)  015300  123200
                <MOVE!SPX!IBUS!NPR!SP0>
1610  015302
                BRWRTE  IMM,221
  (1)           001370
                MICPC=MICPC+1
  (1)  015302  000621
                <MOVE!WRTEBR!IMM!<221>>
1611  015304
                ALWAYS  RK7
  (1)           001371
                MICPC=MICPC+1
  (1)  015304  104643
                <JUMP!ALCOND!<RK7-INIT&3000*4>!<RK7-INIT&777/2>>
1613  015306
HOINCH: SP      IBUS,IIBA2,SP0
  (1)           001372
                MICPC=MICPC+1
  (1)  015306  023120
                <MOVE!SPX!IBUS!IIBA2!SP0>
1614  015310
                OUTPUT  BR,INCA!IBA2                    ;OUTPUT INCREMENTED BA
  (1)           001373
                MICPC=MICPC+1
  (1)  015310  062065
                <MOVE!WROUT!BR!<INCA!IBA2>>
1615  015312
                C       5$                              ;INCREMENT BYTEW COUNT
  (1)           001374
                MICPC=MICPC+1
  (1)  015312  111376
                <JUMP!CCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
1616  015314
                ALWAYS  TH8
  (1)           001375
                MICPC=MICPC+1
  (1)  015314  110565
                <JUMP!ALCOND!<TH8-INIT&3000*4>!<TH8-INIT&777/2>>
1617
                ;INCREMENT MXT BITS
1618  015316
5$:     SP      IBUS,NPR,SP0            ;READ NPR REG IWTH CURRENT MXT BITS
  (1)           001376
                MICPC=MICPC+1
  (1)  015316  123200
                <MOVE!SPX!IBUS!NPR!SP0>
1619  015320
                BRWRTE  IMM,4                   ;WRITE BIT TO ADD
```

```
 (1)            001377
 (1)  015320    000404
1620  015322
 (1)            001400
 (1)  015322    061010
1621  015324
 (1)            001401
 (1)  015324    110565
1622
1623  015326                  HEH1:   STATE   TEOM
 (1)            001402                 MICPC=MICPC+1
 (1)  015326    000710                 <MOVE!WRTEBR!IMM!<TEOM-INIT&777/2>>
1624  015330                           ALWAYS  XEXIT
 (1)            001403                 MICPC=MICPC+1
 (1)  015330    100451                 <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
1625                                    ;
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IMM!<4>>
                                        OUT     BR,<ADD!ONPR>           ;TURN ON PROPER MXT BITS
                                        MICPC=MICPC+1
                                        <MOVE!WROUTX!BR!<ADD!ONPR>>
                                        ALWAYS  TH8
                                        MICPC=MICPC+1
                                        <JUMP!ALCOND!<TH8-INIT&3000*4>!<TH8-INIT&777/2>>
                                        ;
```

```
1627
1628  015332
 (1)              001404
 (1)  015332      010016
1629  015334
 (1)              001405
 (1)  015334      043220
1630  015336
 (1)              001406
 (1)  015336      062460
1631  015340
 (1)              001407
 (1)  015340      010153
1632  015342
 (1)              001410
 (1)  015342      016402
1633  015344
 (1)              001411
 (1)  015344      016703
1634  015346
 (1)              001412
 (1)  015346      114712
1635
1636
1637  015350
 (1)              001413
 (1)  015350      060610
1638  015352
 (1)              001414
 (1)  015352      001620
1639  015354
 (1)              001415
 (1)  015354      117021
1640
1641  015356
 (1)              001416
 (1)  015356      010177
1642  015360
 (1)              001417
 (1)  015360      000600
1643  015362
 (1)              001420
 (1)  015362      114525
1644  015364
 (1)              001421
 (1)  015364      010153
1645  015366
 (1)              001422
 (1)  015366      016407
1646  015370
 (1)              001423
 (1)  015370      000411
1647  015372
 (1)              001424
 (1)  015372      110732
1648
```

```
                .SBTTL  REP HANDLER
REP:    LDMA    IMM,REPCR               ;LOAD MAR ADDRESS WITH POINTER TO REPS RECC
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<REPCR&377>>
        SP      MEMX,SELB,SPO           ;READ NUMBER OF REPS RECD
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SPO>
        MEM     DP,<INCA!SPO>           ;INCREMNT REPS RECD
        MICPC=MICPC+1
        <MOVE!WRMEM!DP!<INCA!SPO>>
        LDMA    IMM,T                   ;LOAD ADDRESS OF TYPE FIELD
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,2                   ; LOAD NAK TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<2>>
        MEMINC  IMM,303                 ;LOAD REP RESPONSE SUB-TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<303>>
        ALWAYS  SNAK                    ;SEND AN UNNUMB MSG
        MICPC=MICPC+1
        <JUMP!ALCOND!<SNAK-INIT&3000*4>!<SNAK-INIT&777/2>>
        ;
                .SBTTL  START HANDLER
START:  BRWRTE  DP,<SELA!SP10>          ;READ LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!WRTEBR!DP!<SELA!SP10>>
        BRSHFT                          ;GET START MODE BIT IN TESTABLE POSITION
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     10$                     ;IF IN START MODE SET STACK
        MICPC=MICPC+1
        <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
                                        ;ELSE SET UP START ERROR
        LDMA    IMM,<<RTHRS+3>>
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
        BRWRTE  IMM,200
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        ALWAYS  RCEXY
        MICPC=MICPC+1
        <JUMP!ALCOND!<RCEXY-INIT&3000*4>!<RCEXY-INIT&777/2>>
10$:    LDMA    IMM,T                   ;SET UP ADDRESS OF TYPE FIELD
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,7                   ;WRITE STACK TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<7>>
        BRWRTE  IMM,11                  ;SET START RECD AND UNNUMB PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<11>>
        ALWAYS  SA2                     ;SEND THE UNNUMBERED MESSAGE
        MICPC=MICPC+1
        <JUMP!ALCOND!<SA2-INIT&3000*4>!<SA2-INIT&777/2>>
        ;
```

```
 1649
 1650  015374
  (1)            001425
  (1)  015374    000727
 1651  015376
  (1)            001426
  (1)  015376    063270
 1652  015400
  (1)            001427
  (1)  015400    110662

                .SBTTL  STACK HANDLER
STACK:  BRWRTE  IMM,327                 ;MASK TO CLEAR START MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<327>>
        SP      BR,AANDB,SP10           ;CLEAR START MODE
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP10>
        ALWAYS  TIME1                   ;RESET TIMER AND IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<TIME1-INIT&3000*4>!<TIME1-INIT&777/2>>
```

```
1654  015402
 (1)              001430
 (1)  015402      023160
1655  015404
 (1)              001431
 (1)  015404      062067
1656  015406
 (1)              001432
 (1)  015406      115034
1657  015410
 (1)              001433
 (1)  015410      104660
1658
1659  015412
 (1)              001434
 (1)  015412      123220
1660  015414
 (1)              001435
 (1)  015414      000404
1661  015416
 (1)              001436
 (1)  015416      061011
1662  015420
 (1)              001437
 (1)  015420      104660
1663  015422
 (1)              001440
 (1)  015422      003200
1664  015424
 (1)              001441
 (1)  015424      062212
1665  015426
 (1)              001442
 (1)  015426      104575
1666  015430
 (1)              001443
 (1)  015430      010070
1667  015432
 (1)              001444
 (1)  015432      057220
1668  015434
 (1)              001445
 (1)  015434      062600
1669  015436
 (1)              001446
 (1)  015436      060477
1670  015440
 (1)              001447
 (1)  015440      063232
1671  015442
 (1)              001450
 (1)  015442      000406
1672
1673  015444
 (1)              001451
 (1)  015444      063310
```

```
ICBA22: SP      IBUS,IOBA2,SPO                  ;READTHEHIGH ORDERBITS OF BA TO SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA2!SPO>
        OUTPUT  DP,<INCA!OBA2>                  ;OUTPUT THE INCREMENTED COUNT
        MICPC=MICPC+1
        <MOVE!WROUT!DP!<INCA!OBA2>>
        C       5$                              ;IF CARRY SET INCREMENT THE MXTBITS
        MICPC=MICPC+1
        <JUMP!CCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        ALWAYS  RK9
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK9-INIT&3000*4>!<RK9-INIT&777/2>>
        .
5$:     SP      IBUS,UBBR,SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!UBBR!SPO>
        BRWRTE  IMM,4
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<4>>
        OUT     BR,<ADD!OBR>
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<ADD!OBR>>
        ALWAYS  RK9
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK9-INIT&3000*4>!<RK9-INIT&777/2>>
FLUSH1: SP      IMM,200,SPO                     ;FLUSH THE RECVR
        MICPC=MICPC+1
        <MOVE!SPX!IMM!200!SPO>
        OUTPUT  BR,<SELA!ORCVC0>
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELA!ORCVC0>>
        ALWAYS  CG1
        MICPC=MICPC+1
        <JUMP!ALCOND!<CG1-INIT&3000*4>!<CG1-INIT&777/2>>
NAK:    LDMA    IMM,STC                         ;ADDRESS START OF TMT CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        SP      MEMX!INCMAR,SELB,SPO            ;COPY IT TO SPO
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!INCMAR!SELB!SPO>
        MEM     BR,SELA!SPO                     ;COPY START  OF CHAIN
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<SELA!SPO>>
        BRWRTE  BR,INCA!SP17                    ;GETLASTMESSAGE ACKED
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<INCA!SP17>>
        SP      BR,SELB,SP12                    ;COPY TO CURRENT NUMBER
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP12>
        BRWRTE  IMM,6                           ;WRITE NUMBERED MSG PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
                                                ; AND LINE HAS GONE IDLE
        SP      BR,AORB,SP10                    ;SET IT IN LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
```

```
 1674  015446
  (1)                001452
  (1)  015446  063235
 1675  015450
  (1)                001453
  (1)  015450  110720
 1676  015452
  (1)                001454
  (1)  015452  000415
 1677  015454
  (1)                001455
  (1)  015454  123220
 1678  015456
  (1)                001456
  (1)  015456  063260
 1679  015460
  (1)                001457
  (1)  015460  000600
 1680  015462
  (1)                001460
  (1)  015462  061311
 1681  015464
  (1)                001461
  (1)  015464  123000
 1682  015466
  (1)                001462
  (1)  015466  100563
 1683
```

```
                SP      BR,SELB,SP15            ;RESET TIMER COUNT
                MICPC=MICPC+1
                <MOVE!SPX!BR!SELB!SP15>
                ALWAYS  TEOM1
                MICPC=MICPC+1
                <JUMP!ALCOND!<TEOM1-INIT&3000*4>!<TEOM1-INIT&777/2>>
ININT:  BRWRTE  IMM,15                          ;MASK FOR TURN OFF ALL BUT EXT MEM BITS + NXM
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<15>>
                SP      IBUS,UBBR,SPO           ;READ BR  CONTROL REGISTER
                MICPC=MICPC+1
                <MOVE!SPX!IBUS!UBBR!SPO>
                SP      BR,AANDB,SPO            ;MASK OFF VECTOR TO X04
                MICPC=MICPC+1
                <MOVE!SPX!BR!AANDB!SPO>
                BRWRTE  IMM,200                 ;MASK FOR INTERRUPT
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<200>>
                OUT     BR,AORB!OBR             ;INTERRUPT
                MICPC=MICPC+1
                <MOVE!WROUTX!BR!<AORB!OBR>>
                SP      IBUS,INCON,SPO          ;RESTORE INPUT CONTROLCSR
                MICPC=MICPC+1
                <MOVE!SPX!IBUS!INCON!SPO>
                ALWAYS  NIDLE4
                MICPC=MICPC+1
                <JUMP!ALCOND!<NIDLE4-INIT&3000*4>!<NIDLE4-INIT&777/2>>
                ;
```

```
1685
1686  015470
 (1)                001463
 (1)   015470  010177
1687  015472
 (1)                001464
 (1)   015472  016401
1688  015474
 (1)                001465
 (1)   015474  002400
1689  015476
 (1)                001466
 (1)   015476  043230
1690  015500
 (1)                001467
 (1)   015500  114527
```

```
        .SBTTL NXMERR ---NON EXISTANT MEMORY HANDLER
NXMERR: LDMA    IMM,<<RTHRS+3>>          ;ADDRESS ERROR LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
        MEMINC  IMM,1
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<1>>
        MEM     IMM,0                    ;NXM ERROR BIT
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<0>>
        SP      MEMX,SELB,SP10           ;CLEAR STATUS
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP10>
        ALWAYS  RCEXX
        MICPC=MICPC+1
        <JUMP!ALCOND!<RCEXX-INIT&3000*4>!<RCEXX-INIT&777/2>>
```

```
1692
1693
1694  015502                        .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
                              MICPC=MICPC+1
                              <MOVE!SPBRX!IBUS!RCVDAT!SP5>
                              BR7     15$                             ;SELECT SET?--BRANCH
                              MICPC=MICPC+1
                              <JUMP!BR7CON!<15$-INIT&3000*4>!<15$-INIT&777/2>>
                      5$:     BRWRTE  BR,AA!SP5                       ;SHIFTBR LEFT
                              MICPC=MICPC+1
                              <MOVE!WRTEBR!BR!<AA!SP5>>
                              BR7     20$                             ;FINAL SET?
                              MICPC=MICPC+1
                              <JUMP!BR7CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
                      10$:    BRWRTE  IMM,77                          ;MASK   TO BR
                              MICPC=MICPC+1
                              <MOVE!WRTEBR!IMM!<77>>
                              SPBR    BR,AANDB,SP5                    ;TURN OFF SELECTANDFINAL
                              MICPC=MICPC+1
                              <MOVE!SPBRX!BR!AANDB!SP5>
                              .ALWAY  BR,INCA,SP3!PAGE1
                              MICPC=MICPC+1
                              <JUMP!ALCOND!BR!INCA!SP3!PAGE1>
                      ;
                      15$:    BRWRTE  IMM,200                         ;SET OK TO SEND
                              MICPC=MICPC+1
                              <MOVE!WRTEBR!IMM!<200>>
1703  015522                  SP      BR,AORB,SP10                    ;IN LINE STATUS WORD
                              MICPC=MICPC+1
                              <MOVE!SPX!BR!AORB!SP10>
1704  015524                  ALWAYS  5$
                              MICPC=MICPC+1
                              <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
1705  015526           20$:   BRWRTE  IMM,20                          ;SETCLEARACTIVE
                              MICPC=MICPC+1
                              <MOVE!WRTEBR!IMM!<20>>
1706  015530                  SP      BR,AORB,SP10                    ;IN LINE STATUS WORD
                              MICPC=MICPC+1
                              <MOVE!SPX!BR!AORB!SP10>
1707  015532                  ALWAYS  10$
                              MICPC=MICPC+1
                              <JUMP!ALCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
```

```
 1709
 1710  015534
  (1)                001505
  (1)  015534  020540
 1711  015536
  (1)                001506
  (1)  015536  116111
 1716  015540
  (1)                001507
  (1)  015540  000674
 1717  015542
  (1)                001510
  (1)  015542  104422
 1718
 1719  015544
  (1)                001511
  (1)  015544  000626
 1720  015546
  (1)                001512
  (1)  015546  104422
 1721
 1722  015550
  (1)                001513
  (1)  015550  040364
 1723  015552
  (1)                001514
  (1)  015552  115116
 1724  015554
  (1)                001515
  (1)  015554  104475
 1725  015556
  (1)                001516
  (1)  015556  010153
 1726  015560
  (1)                001517
  (1)  015560  016402
 1727  015562
  (1)                001520
  (1)  015562  002711
 1728  015564
  (1)                001521
  (1)  015564  010175
 1729  015566
  (1)                001522
  (1)  015566  036540
 1730  015570
  (1)                001523
  (1)  015570  036560
 1731  015572
  (1)                001524
  (1)  015572  000420
 1732  015574
  (1)                001525
  (1)  015574  016400
 1733  015576
  (1)                001526
```

```
        ;FUGITIVE RECEIVE ROUTINES---DON'T FIT IN PAGE
RH1:    BRWRTE  IBUS,IOBA1                      ;READ LOW BYTE OF IN BA
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<IOBA1>>
        BRO     RCVODD                          ;IF SET IS ODD TRANSFER
        MICPC=MICPC+1
        <JUMP!BROCON!<RCVODD-INIT&3000*4>!<RCVODD-INIT&777/2>>
        STATE   RCVKE0
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVKE0-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>

RCVODD: STATE   RCVK01
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVK01-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>

RCLOW:  CMP     MEMX,SP4                        ;COMPARE LOW ORDER BITS OF COUNT
        MICPC=MICPC+1
        <SUBTC!MEMX!SP4>
        C       RCFATL                          ;CARRY--TOO BIG
        MICPC=MICPC+1
        <JUMP!CCOND!<RCFATL-INIT&3000*4>!<RCFATL-INIT&777/2>>
        ALWAYS  RC5                             ;ELSE CONTINUE
        MICPC=MICPC+1
        <JUMP!ALCOND!<RC5-INIT&3000*4>!<RC5-INIT&777/2>>
RCFATL: LDMA    IMM,T
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,2
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<2>>
        MEM     IMM,311
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<311>>
        LDMA    IMM,<<RTHRS+1>>                 ;ADDRESS ERROR LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<<RTHRS+1>&377>>
        MEMINC  IBUS,IOBA1
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IBUS!<IOBA1>>
        MEMINC  IBUS,IOBA2
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IBUS!<IOBA2>>
        BRWRTE  IMM,20
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<20>>
RCEXY:  MEMINC  IMM,0
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<0>>
        MEM     BR,SELB
        MICPC=MICPC+1
```

```
                                    (1)  015576  062620
                                  1734   015600
                                    (1)          001527
                                    (1)  015600  002212
                                  1735   015602
                                    (1)          001530
                                    (1)  015602  003001
                                  1736   015604
                                    (1)          001531
                                    (1)  015604  114674
                                  1737
                                  1738   015606
                                    (1)          001532
                                    (1)  015606  040757
                                  1739   015610
                                    (1)          001533
                                    (1)  015610  107567
                                  1740   015612
                                    (1)          001534
                                    (1)  015612  070200
                                  1741   015614
                                    (1)          001535
                                    (1)  015614  002400
                                  1742   015616
                                    (1)          001536
                                    (1)  015616  010070
                                  1743   015620
                                    (1)          001537
                                    (1)  015620  002473
                                  1744   015622
                                    (1)          001540
                                    (1)  015622  000545
                                  1745   015624
                                    (1)          001541
                                    (1)  015624  060360
                                  1746   015626
                                    (1)          001542
                                    (1)  015626  115545
                                  1747   015630
                                    (1)          001543
                                    (1)  015630  000406
                                  1748   015632
                                    (1)          001544
                                    (1)  015632  062400
                                  1749   015634
                                    (1)          001545
                                    (1)  015634  010241
                                  1750   015636
                                    (1)          001546
                                    (1)  015636  053223
                                  1751
                                  1752   015640
                                    (1)          001547
                                    (1)  015640  016600
                                  1753   015642
                                    (1)          001550

                                         <MOVE!WRMEM!BR!<SELB>>
RCEXX:  OUTPUT  IMM,<200!ORCVCO>        ;FLUSH THE INPUT SILO
        MICPC=MICPC+1
        <MOVE!WRQUT!IMM!<200!ORCVCO>>
        SP      IMM,1,SP1               ;SET INIT MODE IN PORT STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!IMM!1!SP1>
        ALWAYS  NTRS1
        MICPC=MICPC+1
        <JUMP!ALCOND!<NTRS1-INIT&3000*4>!<NTRS1-INIT&777/2>>
        .
TDON3:  BRWRTE  MEMX,SUB!SP17                   ;COMPARE RESPONSE TO MSG NO
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!<SUB!SP17>>
        BR7     RH3                             ;IF NEGATIVE EXIT
        MICPC=MICPC+1
        <JUMP!BR7CON!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
TDON2:  LDMA    BR,SELA!SPO                     ;ADDRESS THE TRANSMITLINK
        MICPC=MICPC+1
        <MOVE!LDMAR!BR!<SELA!SPO>>
        MEM     IMM,0                           ;TURN OF ASSIGNEDAND TMTED BITS IN FLAG
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<0>>
        LDMA    IMM,STC
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        MEM     IMM,TML1                        ;ASSUME WRAPAROUND
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<TML1>>
        BRWRTE  IMM,TML8                        ;WRAPAROUND?
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TML8>>
        CMP     BR,SPO
        MICPC=MICPC+1
        <SUBTC!BR!SPO>
        Z       TDON4                           ;YES
        MICPC=MICPC+1
        <JUMP!ZCOND!<TDON4-INIT&3000*4>!<TDON4-INIT&777/2>>
        BRWRTE  IMM,6                           ;OFFSET FOR NEXT TMT LINK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
        MEM     BR,ADD!SPO                      ;UPDATE THE POINTER
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<ADD!SPO>>
TDON4:  LDMA    IMM,NXTSP                       ;ADDRESS DONE LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NXTSP&377>>
        LDMA    MEMX,SELB!SPX!SP3               ;ADDRESS THE LINK,COPYING
        MICPC=MICPC+1
        <MOVE!LDMAR!MEMX!<SELB!SPX!SP3>>
                                                ;ITS ADDRESS TO SPO
        MEMINC  IMM,200                         ;WRITE THE INTERRUPT TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<200>>
        MEM     BR,INCA!SPO                     ;COPY ACTUAL LINK ADDRESS
        MICPC=MICPC+1
```

```
  (1)  015642  062460                  <MOVE!WRMEM!BR!<INCA!SP0>>
1754   015644                          LDMA    IMM,NXTSP                       ;ADDRESS PTR INT STACK
  (1)                                  MICPC=MICPC+1
  (1)                 001551           <MOVE!LDMAR!IMM!<NXTSP&377>>
  (1)  015644  010241
1755   015646                          MEM     IMM,INTSTK                      ;ASSUME WRAP AROUND
  (1)                 001552           MICPC=MICPC+1
  (1)  015646  002642                  <MOVE!WRMEM!IMM!<INTSTK>>
1756   015650                          BRWRTE  IMM,<<MMEND-2>>                 ;ADDRESS ENDOFINT STACK
  (1)                 001553           MICPC=MICPC+1
  (1)  015650  000776                  <MOVE!WRTEBR!IMM!<<MMEND-2>>>
1757   015652                          CMP     BR,SP3                          ;WRAPAROUND?
  (1)                 001554           MICPC=MICPC+1
  (1)  015652  060363                  <SUBTC!BR!SP3>
1758   015654                          Z       TDON40                          ;YES---BRANCH
  (1)                 001555           MICPC=MICPC+1
  (1)  015654  115560                  <JUMP!ZCOND!<TDON40-INIT&3000*4>!<TDON40-INIT&777/2>>
1759   015656                          BRWRTE  IMM,2                           ;OFFSET TO NEXT PAIR
  (1)                 001556           MICPC=MICPC+1
  (1)  015656  000402                  <MOVE!WRTEBR!IMM!<2>>
1760   015660                          MEM     BR,ADD!SP3                      ;UPDATE POINTER
  (1)                 001557           MICPC=MICPC+1
  (1)  015660  062403                  <MOVE!WRMEM!BR!<ADD!SP3>>
1761   015662                  TDON40: BRWRTE  IMM,20                          ;WRITE INTERUPT PENDING
  (1)                 001560           MICPC=MICPC+1
  (1)  015662  000420                  <MOVE!WRTEBR!IMM!<20>>
1762   015664                          SP      BR,AORB,SP1                     ;IN PORT STATUS WORD
  (1)                 001561           MICPC=MICPC+1
  (1)  015664  063301                  <MOVE!SPX!BR!AORB!SP1>
1763   015666                          LDMA    IMM,ETC                         ;ADDRESS NEXT EMPTY PTR
  (1)                 001562           MICPC=MICPC+1
  (1)  015666  010072                  <MOVE!LDMAR!IMM!<ETC&377>>
1764   015670                          SP      MEMX,SELB,SP0                   ;COPY IT TO SP0
  (1)                 001563           MICPC=MICPC+1
  (1)  015670  043220                  <MOVE!SPX!MEMX!SELB!SP0>
1765   015672                          LDMA    IMM,STC                         ;GET NEXT DONE PTR
  (1)                 001564           MICPC=MICPC+1
  (1)  015672  010070                  <MOVE!LDMAR!IMM!<STC&377>>
1766   015674                          CMP     MEMX,SP0                        ;IDENTICAL?
  (1)                 001565           MICPC=MICPC+1
  (1)  015674  040360                  <SUBTC!MEMX!SP0>
1767   015676                          Z       RH3                             ;FINISH PROCESSING HEADER
  (1)                 001566           MICPC=MICPC+1
  (1)  015676  105567                  <JUMP!ZCOND!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
1768
1769   015700                  TDON1:  LDMA    IMM,ISP17                       ;GET LAST ACKED
  (1)                 001567           MICPC=MICPC+1
  (1)  015700  010155                  <MOVE!LDMAR!IMM!<ISP17&377>>
1770   015702                          SP      MEMX,SELB,SP17                  ;STORE IT IN SP17
  (1)                 001570           MICPC=MICPC+1
  (1)  015702  043237                  <MOVE!SPX!MEMX!SELB!SP17>
1771   015704                          LDMA    IMM,STC                         ;GET START OF TMT CHAIN
  (1)                 001571           MICPC=MICPC+1
  (1)  015704  010070                  <MOVE!LDMAR!IMM!<STC&377>>
1772   015706                          LDMA    MEMX,SELB!SPBRX!SP0             ;ADDRESS THE LINK
  (1)                 001572           MICPC=MICPC+1
  (1)  015706  053620                  <MOVE!LDMAR!MEMX!<SELB!SPBRX!SP0>>
```

```
1773  015710
 (1)            001573
 (1)  015710    054620          BRWRTE  MEMX!INCMAR,SELB                    ;GET THE FLAGS
1774  015712                    MICPC=MICPC+1
 (1)            001574          <MOVE!WRTEBR!MEMX!INCMAR!<SELB>>
 (1)  015712    116532          BR1     TDON3                               ;IFBUFFER ASSIGNED PROCEED
1775  015714                    MICPC=MICPC+1
 (1)            001575          <JUMP!BR1CON!<TDON3-INIT&3000*4>!<TDON3-INIT&777/2>>
 (1)  015714    104567          ALWAYS  RH3                                 ;ELSE---EXIT
1776                            MICPC=MICPC+1
1777  015716                    <JUMP!ALCOND!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
 (1)            001576
 (1)  015716    000404  OVRRUN: BRWRTE  IMM,4
1778  015720                    MICPC=MICPC+1
 (1)            001577          <MOVE!WRTEBR!IMM!<4>>
 (1)  015720    114671          ALWAYS  NTRSO
1779                            MICPC=MICPC+1
1780                            <JUMP!ALCOND!<NTRSO-INIT&3000*4>!<NTRSO-INIT&777/2>>
1781                    ; INPUTS:
1782                    ;       SPO = RECEIVE CHARACTER
1783  015722
 (1)            001600  PASWRD: SP      IBUS,LNOSW,SP16                     ;READ PASSWD SWITCH
 (1)  015722    023336          MICPC=MICPC+1
1784  015724                    <MOVE!SPX!IBUS!LNOSW!SP16>
 (1)            001601          Z       10$                                 ;IF ALL ONES NO RLD ENABLED
 (1)  015724    115605          MICPC=MICPC+1
1785  015726                    <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
 (1)            001602          BRWRTE  IMM,6                               ;CHECK FOR ENTER MOP MODE
 (1)  015726    000406          MICPC=MICPC+1
1786  015730                    <MOVE!WRTEBR!IMM!<6>>
 (1)            001603          CMP     BR,SPO
 (1)  015730    060360          MICPC=MICPC+1
1787  015732                    <SUBTC!BR!SPO>
 (1)            001604          Z       20$                                 ;IF EQUAL   ENTER MOP
 (1)  015732    115613          MICPC=MICPC+1
1788  015734                    <JUMP!ZCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
 (1)            001605  10$:    BRWRTE  BR,SELA!SP1                         ;READ STATUS BYTE
 (1)  015734    060601          MICPC=MICPC+1
1789  015736                    <MOVE!WRTEBR!BR!<SELA!SP1>>
 (1)            001606          BRSHFT                                      ;SHIFT IT RIGHT
 (1)  015736    001620          MICPC=MICPC+1
1790  015740                    <MOVE!SHFTBR!WRTEBR!SELB>
 (1)            001607          BR1     RHX                                 ;MESSAGE WITH NO BUFFER ASSIGNED
 (1)  015740    106751          MICPC=MICPC+1
1791  015742                    <JUMP!BR1CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
 (1)            001610          BRSHFT                                      ;SHIFT RIGHT AGAIN
 (1)  015742    001620          MICPC=MICPC+1
1792  015744                    <MOVE!SHFTBR!WRTEBR!SELB>
 (1)            001611          BR1     RCVMO                               ;DLE RECEIVED IN NORMAL MODE
 (1)  015744    106743          MICPC=MICPC+1
1793  015746                    <JUMP!BR1CON!<RCVMO-INIT&3000*4>!<RCVMO-INIT&777/2>>
 (1)            001612          ALWAYS  RK3                                 ;HANDLE MAINT MODE MESSAGE
 (1)  015746    104661          MICPC=MICPC+1
1794  015750                    <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
 (1)            001613  20$:    SP      BR,DECA,SP4                         ;COUNT FOR NUMB OF COMPARES
 (1)  015750    063164          MICPC=MICPC+1
                                <MOVE!SPX!BR!DECA!SP4>
```

```
1795  015752
 (1)            001614
 (1)  015752    000746          STATE   EM2
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!IMM!<EM2-INIT&777/2>>
                                ALWAYS  REXIT
1796  015754
 (1)            001615          MICPC=MICPC+1
 (1)  015754    104422          <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
1797  015756           RCVM1:   LDMA    IMM,ISP11                      ;ADDRESS SP11 IMAGE
 (1)            001616          MICPC=MICPC+1
 (1)  015756    010171          <MOVE!LDMAR!IMM!<ISP11&377>>
1798  015760                   MEM     BR,DECA!SP11                   ;COPY SP11
 (1)            001617          MICPC=MICPC+1
 (1)  015760    062571          <MOVE!WRMEM!BR!<DECA!SP11>>
1799  015762                   LDMA    IMM,NXTSP
 (1)            001620          MICPC=MICPC+1
 (1)  015762    010241          <MOVE!LDMAR!IMM!<NXTSP&377>>
1800  015764                   SP      MEMX!LDMAR,SELB,SP3                    ;COPY TO SP3
 (1)            001621          MICPC=MICPC+1
 (1)  015764    053223          <MOVE!SPX!MEMX!LDMAR!SELB!SP3>
1801  015766                   MEMINC  IMM,204                        ;RECEIVE DONE IMAGE
 (1)            001622          MICPC=MICPC+1
 (1)  015766    016604          <MOVE!WRMEM!INCMAR!IMM!<204>>
1802  015770                   MEM     BR!LDMAR,SELA!SP14                     ;COPY LINK ADDRWESS TO NEXT INTE
 (1)            001623          MICPC=MICPC+1
 (1)  015770    072614          <MOVE!WRMEM!BR!LDMAR!<SELA!SP14>>
1803  015772                   MEMINC  IMM,0                          ;ZERO THE FLAGS
 (1)            001624          MICPC=MICPC+1
 (1)  015772    016400          <MOVE!WRMEM!INCMAR!IMM!<0>>
1804  015774                   SP      IBUS,IOBA1,SP4                 ;GET BEGIN ADDRESS LOW
 (1)            001625          MICPC=MICPC+1
 (1)  015774    023144          <MOVE!SPX!IBUS!IOBA1!SP4>
1805  015776                   SP      IBUS,IOBA2,SP5                 ;AND HIGH BYTE
 (1)            001626          MICPC=MICPC+1
 (1)  015776    023165          <MOVE!SPX!IBUS!IOBA2!SP5>
1806  016000                   SP      MEMX!INCMAR,SUB,SP4            ;SUBTRACT TO GET COUNT
 (1)            001627          MICPC=MICPC+1
 (1)  016000    057344          <MOVE!SPX!MEMX!INCMAR!SUB!SP4>
1807  016002                   C       10$                    ;IF CARRY SET THEN NO CARRY!
 (1)            001630          MICPC=MICPC+1
 (1)  016002    115232          <JUMP!CCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1808  016004                   SP      BR,DECA,SP5            ;DECREMENT HIGH BYTE OF ADDRESS
 (1)            001631          MICPC=MICPC+1
 (1)  016004    063165          <MOVE!SPX!BR!DECA!SP5>
1809  016006           10$:    SP      MEMX!INCMAR,SUB,SP5            ;SUBTRACT FOR COUNT
 (1)            001632          MICPC=MICPC+1
 (1)  016006    057345          <MOVE!SPX!MEMX!INCMAR!SUB!SP5>
1810  016010                   MEMINC  BR,SELA!SP5                    ;COPY TO MEMORY LINK
 (1)            001633          MICPC=MICPC+1
 (1)  016010    076605          <MOVE!WRMEM!INCMAR!BR!<SELA!SP5>>
1811  016012                   MEMINC  BR,SELA!SP4
 (1)            001634          MICPC=MICPC+1
 (1)  016012    076604          <MOVE!WRMEM!INCMAR!BR!<SELA!SP4>>
1812  016014                   BRWRTE  IMM,2
 (1)            001635          MICPC=MICPC+1
 (1)  016014    000402          <MOVE!WRTEBR!IMM!<2>>
1813  016016                   LDMA    IMM,NXTSP                      ;ADDRESS NEXT INT STACK
 (1)            001636          MICPC=MICPC+1
```

```
 (1)  016016  010241          <MOVE!LDMAR!IMM!<NXTSP&377>>
1814  016020                 MEM     BR,ADD!SP3
 (1)          001637         MICPC=MICPC+1
 (1)  016020  062403         <MOVE!WRMEM!BR!<ADD!SP3>>
1815  016022                 BRWRTE  IMM,<<MMEND-2>>                 ;ADDRESSEND OF INT STACK
 (1)          001640         MICPC=MICPC+1
 (1)  016022  000776         <MOVE!WRTEBR!IMM!<<MMEND-2>>>
1816  016024                 CMP     BR,SP3                          ;WRAP AROUND
 (1)          001641         MICPC=MICPC+1
 (1)  016024  060363         <SUBTC!BR!SP3>
1817  016026                 Z       RMIFLP                          ;IFYES-- BRANCH
 (1)          001642         MICPC=MICPC+1
 (1)  016026  115644         <JUMP!ZCOND!<RMIFLP-INIT&3000*4>!<RMIFLP-INIT&777/2>>
1818  016030                 ALWAYS  RMX1
 (1)          001643         MICPC=MICPC+1
 (1)  016030  114645         <JUMP!ALCOND!<RMX1-INIT&3000*4>!<RMX1-INIT&777/2>>
1819  016032  RMIFLP: MEM     IMM,INTSTK
 (1)          001644         MICPC=MICPC+1
 (1)  016032  002642         <MOVE!WRMEM!IMM!<INTSTK>>
1820  016034  RMX1:
1821  016034                 LDMA    IMM,LRC
 (1)          001645         MICPC=MICPC+1
 (1)  016034  010024         <MOVE!LDMAR!IMM!<LRC&377>>
1822  016036                 BRWRTE  IMM,5                           ;INDEX TO NEXT BUFFER
 (1)          001646         MICPC=MICPC+1
 (1)  016036  000405         <MOVE!WRTEBR!IMM!<5>>
1823  016040                 SP      BR,ADD,SP14                     ;UPDATE COPY OF POINTER
 (1)          001647         MICPC=MICPC+1
 (1)  016040  063014         <MOVE!SPX!BR!ADD!SP14>
1824  016042                 BRWRTE  IMM,STC                         ;ADDRESS OF WRAP AROUND POINT
 (1)          001650         MICPC=MICPC+1
 (1)  016042  000470         <MOVE!WRTEBR!IMM!<STC>>
1825  016044                 CMP     BR,SP14                         ;WRAPAROUND?
 (1)          001651         MICPC=MICPC+1
 (1)  016044  060374         <SUBTC!BR!SP14>
1826  016046                 Z       RMFLIP                          ;IF YES---BRANCH
 (1)          001652         MICPC=MICPC+1
 (1)  016046  115655         <JUMP!ZCOND!<RMFLIP-INIT&3000*4>!<RMFLIP-INIT&777/2>>
1827  016050                 MEM     BR,SELA!SP14                    ;ELSE UPDATE THE POINTER
 (1)          001653         MICPC=MICPC+1
 (1)  016050  062614         <MOVE!WRMEM!BR!<SELA!SP14>>
1828  016052                 ALWAYS  RMX
 (1)          001654         MICPC=MICPC+1
 (1)  016052  114656         <JUMP!ALCOND!<RMX-INIT&3000*4>!<RMX-INIT&777/2>>
1829  016054  RMFLIP: MEM     IMM,RCL1                        ;POINT TO FIRST LINK
 (1)          001655         MICPC=MICPC+1
 (1)  016054  002425         <MOVE!WRMEM!IMM!<RCL1>>
1830  016056  RMX:    BRWRTE  IMM,20                          ;MASK FOR INTERUPT PENDING
 (1)          001656         MICPC=MICPC+1
 (1)  016056  000420         <MOVE!WRTEBR!IMM!<20>>
1831  016060                 SP      DP,AORB,SP1                     ;UPDATE PORT STATUS WORD
 (1)          001657         MICPC=MICPC+1
 (1)  016060  063301         <MOVE!SPX!DP!AORB!SP1>
1832  016062                 BRWRTE  DP,<SELA!SP10>                  ;READ LINE STATUS WORD
 (1)          001660         MICPC=MICPC+1
 (1)  016062  060610         <MOVE!WRTEBR!DP!<SELA!SP10>>
```

```
1833  016064
 (1)              001661
 (1)  016064  107015          BR4     FLUSH                           ;IF CLEAR ACTIVE SET---FLUSH
                              MICPC=MICPC+1
                              <JUMP!BR4CON!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
1834  016066          RM1:    STATE   RCVA
 (1)              001662      MICPC=MICPC+1
 (1)  016066  000400          <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
1835  016070                  ALWAYS  REXIT
 (1)              001663      MICPC=MICPC+1
 (1)  016070  104422          <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
1836  016072          NTHRES: LDMA    IMM,ST
 (1)              001664      MICPC=MICPC+1
 (1)  016072  010154          <MOVE!LDMAR!IMM!<ST&377>>
1837  016074                  SPBR    MEMX,SELB,SPO
 (1)              001665      MICPC=MICPC+1
 (1)  016074  043620          <MOVE!SPBRX!MEMX!SELB!SPO>
1838  016076                  BRWRTE  BR,ADD!SPO                      ;SHIFT LEFT
 (1)              001666      MICPC=MICPC+1
 (1)  016076  060400          <MOVE!WRTEBR!BR!<ADD!SPO>>
1839  016100                  BR4     OVRRUN
 (1)              001667      MICPC=MICPC+1
 (1)  016100  117176          <JUMP!BR4CON!<OVRRUN-INIT&3000*4>!<OVRRUN-INIT&777/2>>
1840  016102                  BRWRTE  IMM,1
 (1)              001670      MICPC=MICPC+1
 (1)  016102  000401          <MOVE!WRTEBR!IMM!<1>>
1841  016104          ERRXX:
1842  016104          NTRS0:  LDMA    IMM,<<RTHRS+3>>
 (1)              001671      MICPC=MICPC+1
 (1)  016104  010177          <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
1843  016106                  MEMINC  IMM,0
 (1)              001672      MICPC=MICPC+1
 (1)  016106  016400          <MOVE!WRMEM!INCMAR!IMM!<0>>
1844  016110                  MEM     BR,SELB
 (1)              001673      MICPC=MICPC+1
 (1)  016110  062620          <MOVE!WRMEM!BR!<SELB>>
1845  016112          NTRS1:  LDMA    IMM,NXTSP
 (1)              001674      MICPC=MICPC+1
 (1)  016112  010241          <MOVE!LDMAR!IMM!<NXTSP&377>>
1846  016114                  LDMA    MEMX,SELB!SPX!SPO
 (1)              001675      MICPC=MICPC+1
 (1)  016114  053220          <MOVE!LDMAR!MEMX!<SELB!SPX!SPO>>
1847  016116                  MEMINC  IMM,201
 (1)              001676      MICPC=MICPC+1
 (1)  016116  016601          <MOVE!WRMEM!INCMAR!IMM!<201>>
1848  016120                  MEM     IMM,<<RTHRS>>
 (1)              001677      MICPC=MICPC+1
 (1)  016120  002574          <MOVE!WRMEM!IMM!<<RTHRS>>>
1849  016122                  LDMA    IMM,NXTSP
 (1)              001700      MICPC=MICPC+1
 (1)  016122  010241          <MOVE!LDMAR!IMM!<NXTSP&377>>
1850  016124                  MEM     IMM,INTSTK                      ;ASSUME QUEUE WRAP AROUND
 (1)              001701      MICPC=MICPC+1
 (1)  016124  002642          <MOVE!WRMEM!IMM!<INTSTK>>
1851  016126                  BRWRTE  IMM,<<MMEND-2>>
 (1)              001702      MICPC=MICPC+1
 (1)  016126  000776          <MOVE!WRTEBR!IMM!<<MMEND-2>>>
1852  016130                  CMP     BR,SPO
```

```
                001703
 (1)
 (1)  016130    060360          MICPC=MICPC+1
                                <SUBTC!BR!SP0>
1853  016132                    Z       NTRS2                           ;IT DID WRAP AROUND
                001704          MICPC=MICPC+1
 (1)
 (1)  016132    115707          <JUMP!ZCOND!<NTRS2-INIT&3000*4>!<NTRS2-INIT&777/2>>
1854  016134                    BRWRTE  IMM,2                           ;OFFSET TO NEXT PAIR
                001705          MICPC=MICPC+1
 (1)
 (1)  016134    000402          <MOVE!WRTEBR!IMM!<2>>
1855  016136                    MEM     BR,ADD!SP0                      ;UPDATE QUEUE POINTER
                001706          MICPC=MICPC+1
 (1)
 (1)  016136    062400          <MOVE!WRMEM!BR!<ADD!SP0>>
1856  016140    NTRS2:  BRWRTE  IMM,20
                001707          MICPC=MICPC+1
 (1)
 (1)  016140    000420          <MOVE!WRTEBR!IMM!<20>>
1857  016142                    SPBR    BR,AORB,SP1
                001710          MICPC=MICPC+1
 (1)
 (1)  016142    063701          <MOVE!SPBRX!BR!AORB!SP1>
1858  016144                    BRO     TAB1                    ;FLAGGED BY ERROR TYPE
                001711          MICPC=MICPC+1
 (1)
 (1)  016144    116342          <JUMP!BROCON!<TAB1-INIT&3000*4>!<TAB1-INIT&777/2>>
1859  016146    SNAK:   LDMA    IMM,ISP11
                001712          MICPC=MICPC+1
 (1)
 (1)  016146    010171          <MOVE!LDMAR!IMM!<ISP11&377>>
1860  016150                    SP      MEMX,SELB,SP11
                001713          MICPC=MICPC+1
 (1)
 (1)  016150    043231          <MOVE!SPX!MEMX!SELB!SP11>
1861  016152                    SP      BR,INCA,SP11                    ;INCREMENT MSG EXPECTED
                001714          MICPC=MICPC+1
 (1)
 (1)  016152    063071          <MOVE!SPX!BR!INCA!SP11>
1862  016154    SNAK1:  BRWRTE  IMM,1                           ;UNNUMB PENING BIT TO BR
                001715          MICPC=MICPC+1
 (1)
 (1)  016154    000401          <MOVE!WRTEBR!IMM!<1>>
1863  016156    SNAK2:  SP      BR,AORB,SP10                    ;UPDATE LINE STATUS WORD
                001716          MICPC=MICPC+1
 (1)
 (1)  016156    063310          <MOVE!SPX!BR!AORB!SP10>
1864  016160                    ALWAYS  FLUSH
                001717          MICPC=MICPC+1
 (1)
 (1)  016160    104415          <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
1865
1866  016162    EMTRIG: BRWRTE  IMM,24
                001720          MICPC=MICPC+1
 (1)
 (1)  016162    000424          <MOVE!WRTEBR!IMM!<24>>
1867  016164                    OUTPUT  BR,<SELB!OBA1>
                001721          MICPC=MICPC+1
 (1)
 (1)  016164    062226          <MOVE!WROUT!BR!<SELB!OBA1>>
1868  016166                    BRWRTE  IMM,0
                001722          MICPC=MICPC+1
 (1)
 (1)  016166    000400          <MOVE!WRTEBR!IMM!<0>>
1869  016170                    OUTPUT  BR,<SELB!OBA2>
                001723          MICPC=MICPC+1
 (1)
 (1)  016170    062227          <MOVE!WROUT!BR!<SELB!OBA2>>
1870  016172                    SPBR    IBUS,BM873,SP0          ;READ BM873 ADDRESS---
                001724          MICPC=MICPC+1
 (1)
 (1)  016172    023740          <MOVE!SPBRX!IBUS!BM873!SP0>
1871  016174                    OUTPUT  BR,SELB!OUTDA1          ;SET UP LOW BYTE OF ADDRESS
                001725          MICPC=MICPC+1
 (1)
```

```
  (1)  016174  062222                <MOVE!WROUT!BR!<SELB!OUTDA1>>
1872   016176                       BRWRTE  IMM,366                 ;HIGH BYTE BASE FOR ROM BOOT
  (1)         001726                MICPC=MICPC+1
  (1)  016176  000766                <MOVE!WRTEBR!IMM!<366>>
1873   016200                       OUTPUT  BR,SELB!OUTDA2          ;
  (1)         001727                MICPC=MICPC+1
  (1)  016200  062223                <MOVE!WROUT!BR!<SELB!OUTDA2>>
1874   016202               EM6:    BRWRTE  IMM,21                  ;MASK FOR TIMER AND ALSO TO START NPR
  (1)         001730                MICPC=MICPC+1
  (1)  016202  000421                <MOVE!WRTEBR!IMM!<21>>
1875   016204                       OUT     BR,<SELB!OBR>
  (1)         001731                MICPC=MICPC+1
  (1)  016204  061231                <MOVE!WROUTX!BR!<SELB!OBR>>
1876   016206                       OUT     BR,<SELB!ONPR>
  (1)         001732                MICPC=MICPC+1
  (1)  016206  061230                <MOVE!WROUTX!BR!<SELB!ONPR>>
1877   016210               EM1:    BRWRTE  IBUS,NPR                ;READ NPR CONTROL
  (1)         001733                MICPC=MICPC+1
  (1)  016210  120600                <MOVE!WRTEBR!IBUS!<NPR>>
1878   016212                       BRO     CKTIME
  (1)         001734                MICPC=MICPC+1
  (1)  016212  102372                <JUMP!BROCON!<CKTIME-INIT&3000*4>!<CKTIME-INIT&777/2>>
1879   016214                       MEMADR  RM1                     ;IF NPR DONE
  (1)         001735                MICPC=MICPC+1
  (1)  016214  002662                <MOVE!WRMEM!<RM1-INIT&777/2>>
1880   016216                       ALWAYS  ACLOW
  (1)         001736                MICPC=MICPC+1
  (1)  016216  100765                <JUMP!ALCOND!<ACLOW-INIT&3000*4>!<ACLOW-INIT&777/2>>
1881   016220               TABUPD: SPBR    IBUS,RCVCON,SP0         ;READ RECEIVER CONTROL REG
  (1)         001737                MICPC=MICPC+1
  (1)  016220  023640                <MOVE!SPBRX!IBUS!RCVCON!SP0>
1882   016222                       BRWRTE  BR,ADD!SP0              ;SHIFT LEFT
  (1)         001740                MICPC=MICPC+1
  (1)  016222  060400                <MOVE!WRTEBR!BR!<ADD!SP0>>
1883   016224                       BR7     IDLE                    ;RECEIVE ACTIVE--IDLE
  (1)         001741                MICPC=MICPC+1
  (1)  016224  103445                <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1884   016226               TAB1:   LDMA    IMM,IMG10
  (1)         001742                MICPC=MICPC+1
  (1)  016226  010156                <MOVE!LDMAR!IMM!<IMG10&377>>
1885   016230                       BRWRTE  IMM,2
  (1)         001743                MICPC=MICPC+1
  (1)  016230  000402                <MOVE!WRTEBR!IMM!<2>>
1886   016232                       MEMINC  BR,AANDB!SP10           ;SAVE BIT 1 OF SP10
  (1)         001744                MICPC=MICPC+1
  (1)  016232  076670                <MOVE!WRMEM!INCMAR!BR!<AANDB!SP10>>
1887   016234                       MEMINC  BR,SELA!SP11            ;SAVE SP11
  (1)         001745                MICPC=MICPC+1
  (1)  016234  076611                <MOVE!WRMEM!INCMAR!BR!<SELA!SP11>>
1888   016236                       MEMINC  BR,SELA!SP12            ;SAVE SP12
  (1)         001746                MICPC=MICPC+1
  (1)  016236  076612                <MOVE!WRMEM!INCMAR!BR!<SELA!SP12>>
1889   016240                       MEMINC  BR,SELA!SP17            ;SAVE SP17
  (1)         001747                MICPC=MICPC+1
  (1)  016240  076617                <MOVE!WRMEM!INCMAR!BR!<SELA!SP17>>
1890                                 ;
```

```
1891  016242
 (1)            001750
 (1)  016242    000461
1892
1893  016244
 (1)            001751
 (1)  016244    060363
1894  016246
 (1)            001752
 (1)  016246    101445
1895
1896  016250
 (1)            001753
 (1)  016250    043233
1897  016252
 (1)  016252
 (2)            001754
 (2)  016252    002775
1898  016254
 (1)            001755
 (1)  016254    003004
1899
1900
1901  016256
 (1)            001756
 (1)  016256    010017
1902  016260
 (1)            001757
 (1)  016260    104443
1903  016262
 (1)            001760
 (1)  016262    000402
1904  016264
 (1)            001761
 (1)  016264    063004
1905  016266
 (1)            001762
 (1)  016266    023140
1906  016270
 (1)            001763
 (1)  016270    052006
1907  016272
 (1)            001764
 (1)  016272    023160
1908  016274
 (1)            001765
 (1)  016274    062107
1909  016276
 (1)            001766
 (1)  016276    105373
1910
1911  016300
 (1)            001767
 (1)  016300    070604
1912  016302
 (1)            001770
```

```
        STATE   RB2             ;NOTE: THE BRG CANNOT BE CHANGED FROM THIS
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RB2-INIT&777/2>>
                                ;POINT UNTIL THE BRANCH TO RB0
        CMP     BR,SP3
        MICPC=MICPC+1
        <SUBTC!BR!SP3>
        Z       IDLE
        MICPC=MICPC+1
        <JUMP!ZCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        .
        SP      MEMX,SELB,SP13
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP13>
        PSTATE  TBU1                    ;NEW PORT STATE ADDRESS
        MEM     IMM,<<TBU1-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<TBU1-INIT&777/2>>>
        SP      IMM,4,SP4               ;INITIALIZE COUNT
        MICPC=MICPC+1
        <MOVE!SPX!IMM!4!SP4>
                                        ;NOTE: FIRST 6 RAM LOCATIONS ARE NOT WRITTEN
                                        ;TO CORE TABLE.
        LDMA    IMM,BASE                ;MAR NOW POINTS TO BASE
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<BASE&377>>
        ALWAYS  RB0             ;THE BRG MUST BE PRESET TO STATE RB2
        MICPC=MICPC+1
        <JUMP!ALCOND!<RB0-INIT&3000*4>!<RB0-INIT&777/2>>
EC2:    BRWRTE  IMM,2                   ;INCREMENT COUNT/TEST
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        SP      BR,ADD,SP4
        MICPC=MICPC+1
        <MOVE!SPX!BR!ADD!SP4>
        SP      IBUS,IOBA1,SP0          ;POINT TO NEXT ADDRESS
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA1!SP0>
        OUTPUT  BR,ADD!OBA1
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<ADD!OBA1>>
        SP      IBUS,IOBA2,SP0
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA2!SP0>
        OUTPUT  BR,AC!OBA2
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<AC!OBA2>>
        C       TABMXT
        MICPC=MICPC+1
        <JUMP!CCOND!<TABMXT-INIT&3000*4>!<TABMXT-INIT&777/2>>

ECX:    BRWRTE  BR!LDMAR,SELA!SP4       ;READ COUNTER
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
        BR7     20$                     ;ALL DONE
        MICPC=MICPC+1
```

```
  (1)  016302  117774                <JUMP!BR7CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
 1913  016304                       OUTPUT  MEMX!INCMAR,SELB!OUTDA1          ;STORE COUNTS OF ERRORS
  (1)         001771                MICPC=MICPC+1
  (1)  016304  056222                <MOVE!WROUT!MEMX!INCMAR!<SELB!OUTDA1>>
 1914  016306                       OUTPUT  MEMX!INCMAR,SELB!OUTDA2
  (1)         001772                MICPC=MICPC+1
  (1)  016306  056223                <MOVE!WROUT!MEMX!INCMAR!<SELB!OUTDA2>>
 1915  016310                       ALWAYS  RK8
  (1)         001773                MICPC=MICPC+1
  (1)  016310  104641                <JUMP!ALCOND!<RK8-INIT&3000*4>!<RK8-INIT&777/2>>
 1916
 1917  016312                 20$:  LDMA    IMM,PRTST
  (1)         001774                MICPC=MICPC+1
  (1)  016312  010162                <MOVE!LDMAR!IMM!<PRTST&377>>
 1918  016314                       MEM     BR,SELA!SP13
  (1)         001775                MICPC=MICPC+1
  (1)  016314  062613                <MOVE!WRMEM!BR!<SELA!SP13>>
 1919  016316                       ALWAYS  RM1
  (1)         001776                MICPC=MICPC+1
  (1)  016316  114662                <JUMP!ALCOND!<RM1-INIT&3000*4>!<RM1-INIT&777/2>>
 1920  016320                       $ZERO
  (1)         001777                MICPC=MICPC+1
  (1)  016320  000000                000000
 1921         000001                .END


. ABS.  016322     000

ERRORS DETECTED:  0

.DDCMP/CRF/DS:CRF←DMCNEW,LOW,DDCNEW
RUN-TIME:  5  8.0 SECONDS
RUN-TIME RATIO: 70/13=5.1
CORE USED:  6K  (11 PAGES)

 1668                                01400
```

```
                                02800
                                02900   HIMAP:                  ;HIGH SPEED (LOCAL) MICRO-CODE
 1669
 1670  016322
```

```
                        .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
004000                 PAGE1=4000
010000                 PAGE2=10000
014000                 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
                          ;
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                                   .SBTTL          INDEXED BRANCH INSTRUCTIONS
                          ;
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                                   .SBTTL          MOVE INSTRUCTIONS
                          ;
        000000            MOVE=0           ;MOVE OPCODE
                          ;
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
```

```
                        .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                  ORCVCO=12                       ;RCVR CONTROL
000013                  OMODEM=13                       ;MODEM CONTROL
000014                  SYNC=14                         ;SYN REGISTER
000017                  OLUMAN=17                       ;LINE UNIT MAINT.
```

```
                        .SBTTL PROTOCOL DEPENDANT MACROS
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;

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
                        .=16322
016322          INIT:   SP      BR,SELB,SPO             ;CLEAR   SPO
                        MICPC=MICPC+1
                        <MOVE!SPX!BR!SELB!SPO>
                        SP      BR,SELB,SP3             ;PAGE ONE TRANSFER ADDRESS
                        MICPC=MICPC+1
                        <MOVE!SPX!BR!SELB!SP3>
                        SP      BR,SELB,SP17            ;CLEAR SP17
                        MICPC=MICPC+1
                        <MOVE!SPX!BR!SELB!SP17>
                        OUT     BR,<SELA!OINCON>        ;ZERO THE IN CONTROL CSR
                        MICPC=MICPC+1
                        <MOVE!WROUTX!BR!<SELA!OINCON>>
                        OUT     BR,<SELA!OOCON>         ;ZERO THE OUT CONTROL CSR
                        MICPC=MICPC+1
                        <MOVE!WROUTX!BR!<SELA!OOCON>>
                        SP      IMM,370,SP10            ;WRITE 5 ONE BITS TO THE HIGH ORDER
                        MICPC=MICPC+1
                        <MOVE!SPX!IMM!370!SP10>
                                                        ;BITS OF SP10
016336          5$:     SP      BR,AA,SP10              ;SHIFT SP10 LEFT SETTING  RRY THE
                        MICPC=MICPC+1
                        <MOVE!SPX!BR!AA!SP10>
                                                        ;FIRST 5 TIMES THRU THE LOOP
                        MEMINC  BR,ADDC!SP3             ;WRITE A ONE TO THE FIRST 5 MEMORY
                        MICPC=MICPC+1
                        <MOVE!WRMEM!INCMAR!BR!<ADDC!SP3>>
                                                        ;LOCATIONS AND ZERO THE REST
                        SP      BR,INCA,SPO             ;INCREMENT COUNTER
                        MICPC=MICPC+1
                        <MOVE!SPX!BR!INCA!SPO>
                        Z       10$                     ;ALL DONE
                        MICPC=MICPC+1
                        <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
                        ALWAYS  5$                      ;KEEP GOING
                        MICPC=MICPC+1
                        <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
016350          10$:    SPBR    IMM,1,SP1               ;WRITE A 1 TO THE BR AND SP1
                        MICPC=MICPC+1
                        <MOVE!SPBRX!IMM!1!SP1>
                        SP      BR,SELB,SP11            ;WRITE A 1 TO SP11
                        MICPC=MICPC+1
                        <MOVE!SPX!BR!SELB!SP11>
                        SP      BR,SELB,SP12            ;WRITE A 1 TO SP12
                        MICPC=MICPC+1
                        <MOVE!SPX!BR!SELB!SP12>
                        LDMA    IMM,TYPTAB              ;POINT MAR TO TYPE TABLE
                        MICPC=MICPC+1
                        <MOVE!LDMAR!IMM!<TYPTAB&377>>
```

```
503  016360
(1)            000017
(1)  016360    000626
504  016362
(1)            000020
(1)  016362    062234
505  016364
(1)            000021
(1)  016364    016403
506  016366
(1)            000022
(1)  016366    002402
507  016370
(1)            000023
(1)  016370    057235
508  016372
(1)            000024
(1)  016372    016406
509  016374
(1)            000025
(1)  016374    016407
510  016376
(1)            000026
(1)  016376    016401
511  016400
(1)            000027
(1)  016400    010210
512  016402
(1)  016402
(2)            000030
(2)  016402    016460
513  016404
(1)  016404
(2)            000031
(2)  016404    016533
514  016406
(1)            000032
(1)  016406    010067
515  016410
(1)            000033
(1)  016410    016471
516  016412
(1)            000034
(1)  016412    002471
517  016414
(1)            000035
(1)  016414    043236
518  016416
(1)            000036
(1)  016416    010022
519  016420
(1)            000037
(1)  016420    016424
520  016422
(1)            000040
(1)  016422    002424
```

```
        BRWRTE  IMM,226                 ;WRITE SYNC TO MEMORY
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<226>>
        OUTPUT  BR,SELB!SYNC    ;LOAD THE SYNC REGISTER
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!SYNC>>
        MEMINC  IMM,3                   ;REP
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<3>>
        MEM     IMM,2                   ;NAK
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<2>>
        SP      MEMX!INCMAR,SELB,SP15           ;SET STARTING COUNT
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!INCMAR!SELB!SP15>
        MEMINC  IMM,6                   ;START
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<6>>
        MEMINC  IMM,7                   ;STACK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<7>>
        MEMINC  IMM,1                   ;ACK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<1>>
        LDMA    IMM,TABST               ;POINT TO TABLE UPDATE STATE
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<TABST&377>>
        PSTATI  I3                      ;INITIALIZE IT
        MEMINC  IMM,<<I3-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<<I3-INIT&777/2>>>
        PSTATI  NIDLE2                  ;INITIALIZE PORT STATUS
        MEMINC  IMM,<<NIDLE2-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<<NIDLE2-INIT&777/2>>>
        LDMA    IMM,STC                 ;LOAD ADDRESS OF LAST TMT CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        MEMINC  IMM,TML1                ;STORE ADDRESS OF FIRST TMT LINK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<TML1>>
        MEM     IMM,TML1
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<TML1>>
        SP      MEMX,SELB,SP16          ;INITIALIZE LAST XMIT POINTER
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP16>
        LDMA    IMM,SRC                 ;LOAD ADDRESS OF LAST RECV CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<SRC&377>>
        MEMINC  IMM,RCL1                ;SET UP ADDRESS OF FIRST RECV LINK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<RCL1>>
        MEM     IMM,RCL1
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<RCL1>>
```

```
521  016424
(1)                 000041
(1)  016424  043234
522  016426
(1)                 000042
(1)  016426  010240
523  016430
(1)                 000043
(1)  016430  016642
524  016432
(1)                 000044
(1)  016432  002642
525  016434
(1)                 000045
(1)  016434  000600
526  016436
(1)                 000046
(1)  016436  061221
527
529  016440
(1)                 000047
(1)  016440  110740
530
531  016442
(1)                 000050
(1)  016442  063223
```

```
        SP      MEMX,SELB,SP14
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP14>
        LDMA    IMM,NXTINT              ;ADDRESS OF NEXT INTERRUPT POINTER TO MAR
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NXTINT&377>>
        MEMINC  IMM,INTSTK                      ;INITIALIZE NEXT INTERRUPT POINTER
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<INTSTK>>
        MEM     IMM,INTSTK                      ;INITIALIZE INSERTION POINTER
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<INTSTK>>
        BRWRTE  IMM,200                 ;WRITE THE RUN BIT TO THE BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        OUT     BR,<SELB!OMAIN>         ;WRITE  THE RUN BIT TO MAINT CSR
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<SELB!OMAIN>>
                                        ;FALL INTO IDLE LOOP
        ALWAYS  TEOM2
        MICPC=MICPC+1
        <JUMP!ALCOND!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
        .
REXIT:  SP      BR,SELB,SP3
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP3>
```

```
                                .SBTTL  IDLE--PROGRAM IDLE LOOP
                                ;PROGRAM IDLE LOOP
                                ;DISPATCHES TO APPROPRIATE SERVICE ROUTINES
                                ;USES STATE POINTERS FOR TMT,RCV,CSR ACTIVITY
                                ;
016444                  IDLE:   BRWRTE  BR,<SELA!SP10>          ;READ TRANSMIT STATUS WORD FROM SP10 TO BR
(1)     000051                  MICPC=MICPC+1
(1)     016444  060610          <MOVE!WRTEBR!BR!<SELA!SP10>>
016446                          BR1     TMTDA                   ;IF DATA TO SEND-- BRANCH
(1)     000052                  MICPC=MICPC+1
(1)     016446  112400          <JUMP!BR1CON!<TMTDA-INIT&3000*4>!<TMTDA-INIT&777/2>>
016450                          BR0     TMTDA                   ;IF DATA TO SEND-- BRANCH
(1)     000053                  MICPC=MICPC+1
(1)     016450  112000          <JUMP!BR0CON!<TMTDA-INIT&3000*4>!<TMTDA-INIT&777/2>>
016452                  I1:     BRWRTE  IBUS,RCVCON             ;READ LINE UNIT RECEIVE CONTROL WORD
(1)     000054                  MICPC=MICPC+1
(1)     016452  020640          <MOVE!WRTEBR!IBUS!<RCVCON>>
016454                          .BR4    BR,SELA,SP3!PAGE1       ;BRANCH BASED UPON RECV STATE
(1)     000055                  MICPC=MICPC+1
(1)     016454  167203          <JUMP! BR4CON!BR!SELA!SP3!PAGE1>
016456                  I2:     LDMA    IMM,TABST               ;POINT TO TABLE UPDATE STATE
(1)     000056                  MICPC=MICPC+1
(1)     016456  010210          <MOVE!LDMAR!IMM!<TABST&377>>
016460                          .ALWAY  MEMX,SELB,0
(1)     000057                  MICPC=MICPC+1
(1)     016460  140620          <JUMP!ALCOND!MEMX!SELB!0>
016462                  I3:
016462                          STATE   TMTA+2                  ;GET IDLE TRANSMIT STATE + 1
(1)     000060                  MICPC=MICPC+1
(1)     016462  000404          <MOVE!WRTEBR!IMM!<TMTA+2-INIT&777/2>>
016464                          NOP     BR,SUB,SP2              ;SUBTRACT FROM CURRENT STATE
(1)     000061                  MICPC=MICPC+1
(1)     016464  060342          <BR!SUB!SP2>
016466                          C       TMTDA                   ;NON-IDLE STATE
(1)     000062                  MICPC=MICPC+1
(1)     016466  111000          <JUMP!CCOND!<TMTDA-INIT&3000*4>!<TMTDA-INIT&777/2>>
016470                  IDLE0:  SPBR    IBUS,UBBR,SP0           ;TIMER EXPIRES?
(1)     000063                  MICPC=MICPC+1
(1)     016470  123620          <MOVE!SPBRX!IBUS!UBBR!SP0>
016472                          BR4     TIMSRV
(1)     000064                  MICPC=MICPC+1
(1)     016472  113255          <JUMP!BR4CON!<TIMSRV-INIT&3000*4>!<TIMSRV-INIT&777/2>>
016474                          SP      IBUS,RCVCON,SP0         ;READ THE RECEIVE CONTROL REGISTER
(1)     000065                  MICPC=MICPC+1
(1)     016474  023240          <MOVE!SPX!IBUS!RCVCON!SP0>
016476                          BRWRTE  BR,AA!SP0               ;SHIFT IT LEFT
(1)     000066                  MICPC=MICPC+1
(1)     016476  060520          <MOVE!WRTEBR!BR!<AA!SP0>>
016500                          BR7     I1                      ;RECEIVE ACTIVE, DON'T DO PORT STATUS
(1)     000067                  MICPC=MICPC+1
(1)     016500  103454          <JUMP!BR7CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
016502                          LDMA    IMM,PRTST               ;ADDRESS PORT STATE
(1)     000070                  MICPC=MICPC+1
(1)     016502  010211          <MOVE!LDMAR!IMM!<PRTST&377>>
016504                          .ALWAY  MEMX,SELB,0             ;INDEX
(1)     000071                  MICPC=MICPC+1
```

```
    (1)  016504  140620                <JUMP!ALCOND!MEMX!SELB!0>
```

```
                                .SBTTL  BASSRV---- BASE SERVICE ROUTINE
016506                  BASSRV: PSTATE  NIDLE2
016506                          MEM     IMM,<<NIDLE2-INIT&777/2>>
                                MICPC=MICPC+1
000072                          <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
002533
016510                          LDMA    IMM,BASE                        ;CLEAR TO MAR SO IT POINTS TO BASE POINT
                                MICPC=MICPC+1
000073                          <MOVE!LDMAR!IMM!<BASE&377>>
010017
016512                          MEMINC  IBUS,PORT1              ;READ CSR4
                                MICPC=MICPC+1
000074                          <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
136500
016514                          MEMINC  IBUS,PORT2              ;READ CSR5
                                MICPC=MICPC+1
000075                          <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
136520
016516                          MEM     IBUS,PORT4
                                MICPC=MICPC+1
000076                          <MOVE!WRMEM!IBUS!<PORT4>>
122560
016520                          SP      IBUS,INCON,SPO          ;READ INPUT CONTROL CSR
                                MICPC=MICPC+1
000077                          <MOVE!SPX!IBUS!INCON!SPO>
123000
016522                          BRWRTE  IMM,100                 ;CLEAR THE BR
                                MICPC=MICPC+1
000100                          <MOVE!WRTEBR!IMM!<100>>
000500
016524                          OUT     BR,<AANDB!OINCON>       ;CLEAR THE INCONTROL CSR
                                MICPC=MICPC+1
000101                          <MOVE!WROUTX!BR!<AANDB!OINCON>>
061260
016526                          OUTPUT  IMM,<120!OMODEM>        ;MASK FOR HDX AND DTR
                                MICPC=MICPC+1
000102                          <MOVE!WROUT!IMM!<120!OMODEM>>
002133
016530                          BRWRTE  MEMX,SELB               ;READ SEL6
                                MICPC=MICPC+1
000103                          <MOVE!WRTEBR!MEMX!<SELB>>
040620
016532                          BR4     RESUME                  ;IF SET RESUME
                                MICPC=MICPC+1
000104                          <JUMP!BR4CON!<RESUME-INIT&3000*4>!<RESUME-INIT&777/2>>
103113
016534                          LDMA    IMM,T                   ;LOAD ADDRESS OF TYPE FIELD
                                MICPC=MICPC+1
000105                          <MOVE!LDMAR!IMM!<T&377>>
010151
016536                          MEMINC  IMM,6                   ;WRITE START TYPE TO MEMORY
                                MICPC=MICPC+1
000106                          <MOVE!WRMEM!INCMAR!IMM!<6>>
016406
016540                          MEM     IMM,300                 ;WRITE SELECT AND FINAL TO MEMORY
                                MICPC=MICPC+1
000107                          <MOVE!WRMEM!IMM!<300>>
002700
016542                          SP      BR,DECA,SP1     ;TURN OFF INIT MODE
                                MICPC=MICPC+1
000110                          <MOVE!SPX!BR!DECA!SP1>
063161
016544                  BS1:    BRWRTE  IMM,241         ;SET OK TO SEND,STARTMODE AND UNNUM PENDING
                                MICPC=MICPC+1
000111                          <MOVE!WRTEBR!IMM!<241>>
000641
016546                          ALWAYS  SA3
                                MICPC=MICPC+1
000112                          <JUMP!ALCOND!<SA3-INIT&3000*4>!<SA3-INIT&777/2>>
110737
016550                  RESUME: SP      IMM,SP4,4               ;SET UP SP4 FOR COUNTING NPRS
                                MICPC=MICPC+1
000113                          <MOVE!SPX!IMM!SP4!4>
003004
```

```
   584  016552                  SP      BR,INCA,SP10            ;SET UNNUMB MESSAGE PENDING TO
   (1)          000114          MICPC=MICPC+1
   (1)  016552  063070          <MOVE!SPX!BR!INCA!SP10>
   585                                                          ;TRICK TRANSMITTER CODE
   586  016554                  LDMA    IMM,BASE                ;ADDRESS BASE TABLE ADDRESS
   (1)          000115          MICPC=MICPC+1
   (1)  016554  010017          <MOVE!LDMAR!IMM!<BASE&377>>
   587  016556                  STATE   FUDGE                   ;SET TMTR STATE TO ENTER TABLE UPDATE
   (1)          000116          MICPC=MICPC+1
   (1)  016556  000743          <MOVE!WRTEBR!IMM!<FUDGE-INIT&777/2>>
   588  016560                  ALWAYS  TBO                     ;GO SET UP MXT BITS AND ADRESS OF BASE FOR NPRS
   (1)          000117          MICPC=MICPC+1
   (1)  016560  110455          <JUMP!ALCOND!<TBO-INIT&3000*4>!<TBO-INIT&777/2>>
   589  016562          BS2:    LDMA    IMM,IMG10
   (1)          000120          MICPC=MICPC+1
   (1)  016562  010154          <MOVE!LDMAR!IMM!<IMG10&377>>
   590  016564                  SP      MEMX!INCMAR,AORB,SP10           ;RESTORE BIT 1 OF SP10
   (1)          000121          MICPC=MICPC+1
   (1)  016564  057310          <MOVE!SPX!MEMX!INCMAR!AORB!SP10>
   591  016566                  SP      MEMX!INCMAR,SELB,SP11           ;RESTORE SP11
   (1)          000122          MICPC=MICPC+1
   (1)  016566  057231          <MOVE!SPX!MEMX!INCMAR!SELB!SP11>
   592  016570                  SP      MEMX!INCMAR,SELB,SP12           ;RESTORE SP12
   (1)          000123          MICPC=MICPC+1
   (1)  016570  057232          <MOVE!SPX!MEMX!INCMAR!SELB!SP12>
   593  016572                  SP      MEMX!INCMAR,SELB,SP14           ;RESTORE SP14
   (1)          000124          MICPC=MICPC+1
   (1)  016572  057234          <MOVE!SPX!MEMX!INCMAR!SELB!SP14>
   594  016574                  SP      MEMX!INCMAR,SELB,SP16           ;RESTORE SP16
   (1)          000125          MICPC=MICPC+1
   (1)  016574  057236          <MOVE!SPX!MEMX!INCMAR!SELB!SP16>
   595  016576                  SP      MEMX,SELB,SP17          ;RESTORE        SP17
   (1)          000126          MICPC=MICPC+1
   (1)  016576  043237          <MOVE!SPX!MEMX!SELB!SP17>
   596  016600                  SP      BR,DECA,SP10            ;TURN OFF UNNUM MESSAGE PENDING AND
   (1)          000127          MICPC=MICPC+1
   (1)  016600  063170          <MOVE!SPX!BR!DECA!SP10>
   597                                                          ;ZERO THE BRG
   598  016602                  SP      BR,DECA,SP1             ;CLEAR INIT MODE
   (1)          000130          MICPC=MICPC+1
   (1)  016602  063161          <MOVE!SPX!BR!DECA!SP1>
   599  016604                  BRWRTE  IMM,200                 ;SET OK TO SEND
   (1)          000131          MICPC=MICPC+1
   (1)  016604  000600          <MOVE!WRTEBR!IMM!<200>>
   600  016606                  ALWAYS  SA3
   (1)          000132          MICPC=MICPC+1
   (1)  016606  110737          <JUMP!ALCOND!<SA3-INIT&3000*4>!<SA3-INIT&777/2>>
```

```
  602
  603  016610
  (1)               000133
  (1)  016610  060601
  605  016612
  (1)               000134
  (1)  016612  103141
  610  016614
  (1)               000135
  (1)  016614  123400
  611  016616
  (1)               000136
  (1)  016616  001620
  612  016620
  (1)               000137
  (1)  016620  103146
  613
  614
  616  016622
  (1)               000140
  (1)  016622  100451
  618  016624
  639  016624
  (1)  016624
  (2)               000141
  (2)  016624  002614
  640  016626
  (1)               000142
  (1)  016626  100451

                .SBTTL  NIDLE2---NO CSR ACTIVITY STATE
NIDLE2: BRWRTE  BR,SELA!SP1             ;READ PORT STATUS WORD
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR4     NIDLE5                  ;INTERRUPT PENDING?---BRANCH
        MICPC=MICPC+1
        <JUMP!BR4CON!<NIDLE5-INIT&3000*4>!<NIDLE5-INIT&777/2>>
        SPBR    IBUS,INCON,SP0          ;READ INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!INCON!SP0>
        BRSHFT                          ;SHIFT IT RIGHT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     INWAT1                  ;IF RQI SET -- BRANCH
        MICPC=MICPC+1
        <JUMP!BR4CON!<INWAT1-INIT&3000*4>!<INWAT1-INIT&777/2>>
                                        ;TO RE-READ THE IN CNTRL REGISTER TO AVOID
                                        ;A RACE IN MICRO-P READ/UNIBUS WRITE
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
NIDLE6:
NIDLE5: PSTATE  OUTINT                  ;SET STATE FOR INTERRUPT PROCESSING
        MEM     IMM,<<OUTINT-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<OUTINT-INIT&777/2>>>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
                        .SBTTL INWAIT---WAIT FOR RQI TO CLEAR
016630           INWAIT: SPBR    IBUS,INCON,SPO          ;READ INPUT CONTROL CSR
(1)  000143             MICPC=MICPC+1
(1)  016630  123400     <MOVE!SPBRX!IBUS!INCON!SPO>
016632                  BRWRTE  BR,<AA!SPO>             ;SHIFT IT LEFT
(1)  000144             MICPC=MICPC+1
(1)  016632  060520     <MOVE!WRTEBR!BR!<AA!SPO>>
016634                  BR7     NIDLE3                  ;INTERRUPT ENABLE HAS BEEN SET
(1)  000145             MICPC=MICPC+1
(1)  016634  103550     <JUMP!BR7CON!<NIDLE3-INIT&3000*4>!<NIDLE3-INIT&777/2>>
016636           INWAT1: SPBR    IBUS,INCON,SPO          ;READ THE INPUT CONTROL CSR
(1)  000146             MICPC=MICPC+1
(1)  016636  123400     <MOVE!SPBRX!IBUS!INCON!SPO>
016640                  BR7     INWAT2                  ;READY IN STILL SET
(1)  000147             MICPC=MICPC+1
(1)  016640  103557     <JUMP!BR7CON!<INWAT2-INIT&3000*4>!<INWAT2-INIT&777/2>>
016642           NIDLE3: PSTATE  INWAT1                  ;UPDATE STATE TO INPUT
(1)  016642             MEM     IMM,<<INWAT1-INIT&777/2>>
(2)  000150             MICPC=MICPC+1
(2)  016642  002546     <MOVE!WRMEM!IMM!<<INWAT1-INIT&777/2>>>
016644                  BRWRTE  BR,AA!SPO               ;SHIFT CSR LEFT
(1)  000151             MICPC=MICPC+1
(1)  016644  060520     <MOVE!WRTEBR!BR!<AA!SPO>>
016646                  BR7     ININT
(1)  000152             MICPC=MICPC+1
(1)  016646  117460     <JUMP!BR7CON!<ININT-INIT&3000*4>!<ININT-INIT&777/2>>
016650                  PSTATE  INWAIT                  ;UPDATE STATE POINTER TO NO INTERRUPT GENERATED
(1)  016650             MEM     IMM,<<INWAIT-INIT&777/2>>
(2)  000153             MICPC=MICPC+1
(2)  016650  002543     <MOVE!WRMEM!IMM!<<INWAIT-INIT&777/2>>>
016652           NIDLE4: BRWRTE  IMM,200
(1)  000154             MICPC=MICPC+1
(1)  016652  000600     <MOVE!WRTEBR!IMM!<200>>
016654                  OUT     BR,AORB!OINCON          ;SET THE RDYI
(1)  000155             MICPC=MICPC+1
(1)  016654  061300     <MOVE!WROUTX!BR!<AORB!OINCON>>
016656                  ALWAYS  IDLE
(1)  000156             MICPC=MICPC+1
(1)  016656  100451     <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>

016660           INWAT2: BRSHFT                          ;SHIFT THE BR RIGHT
(1)  000157             MICPC=MICPC+1
(1)  016660  001620     <MOVE!SHFTBR!WRTEBR!SELB>
016662                  BR4     IDLE
(1)  000160             MICPC=MICPC+1
(1)  016662  103051     <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
016664                  PSTATE  INSRV                   ;SET NEXT STATE TO INPUT SERVICE
(1)  016664             MEM     IMM,<<INSRV-INIT&777/2>>
(2)  000161             MICPC=MICPC+1
(2)  016664  002563     <MOVE!WRMEM!IMM!<<INSRV-INIT&777/2>>>
016666                  ALWAYS  IDLE
(1)  000162             MICPC=MICPC+1
(1)  016666  100451     <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
016670           INSRV:  SPBR    IBUS,INCON,SPO          ;READ THE INPUT CONTROL CSR
(1)  000163             MICPC=MICPC+1
```

```
(1)  016670  123400                    <MOVE!SPBRX!IBUS!INCON!SPO>
668  016672                             BR1     30$                     ;--SENSE OR BASE
(1)            000164                   MICPC=MICPC+1
(1)  016672  102600                     <JUMP!BR1CON!<30$-INIT&3000*4>!<30$-INIT&777/2>>
669  016674                             BR0     10$                     ;CNTL I
(1)            000165                   MICPC=MICPC+1
(1)  016674  102172                     <JUMP!BR0CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
670  016676                             BRSHFT                          ;MUST BE BA/CC-SHIFT FOR IN OR OUT
(1)            000166                   MICPC=MICPC+1
(1)  016676  001620                     <MOVE!SHFTBR!WRTEBR!SELB>
671  016700                             BR1     15$
(1)            000167                   MICPC=MICPC+1
(1)  016700  102574                     <JUMP!BR1CON!<15$-INIT&3000*4>!<15$-INIT&777/2>>
672  016702                             PSTATE  TBASRV                  ;TRANSMITTER
(1)  016702                             MEM     IMM,<<TBASRV-INIT&777/2>>
(2)            000170                   MICPC=MICPC+1
(2)  016702  002700                     <MOVE!WRMEM!IMM!<<TBASRV-INIT&777/2>>>
673  016704                             ALWAYS  20$
(1)            000171                   MICPC=MICPC+1
(1)  016704  100575                     <JUMP!ALCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
674  016706                    10$:     PSTATE  CTLSRV
(1)  016706                             MEM     IMM,<<CTLSRV-INIT&777/2>>
(2)            000172                   MICPC=MICPC+1
(2)  016706  002657                     <MOVE!WRMEM!IMM!<<CTLSRV-INIT&777/2>>>
675  016710                             ALWAYS  20$
(1)            000173                   MICPC=MICPC+1
(1)  016710  100575                     <JUMP!ALCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
676  016712                    15$:     PSTATE  RBASRV
(1)  016712                             MEM     IMM,<<RBASRV-INIT&777/2>>
(2)            000174                   MICPC=MICPC+1
(2)  016712  002721                     <MOVE!WRMEM!IMM!<<RBASRV-INIT&777/2>>>
677  016714                    20$:     BRWRTE  BR,SELA!SP1             ;INIT MODE
(1)            000175                   MICPC=MICPC+1
(1)  016714  060601                     <MOVE!WRTEBR!BR!<SELA!SP1>>
678  016716                             BR0     PROCER                  ;IF INIT MODE--ERROR
(1)            000176                   MICPC=MICPC+1
(1)  016716  102201                     <JUMP!BR0CON!<PROCER-INIT&3000*4>!<PROCER-INIT&777/2>>
679  016720                             ALWAYS  IDLE
(1)            000177                   MICPC=MICPC+1
(1)  016720  100451                     <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
680  016722                    30$:     BR0     INSRV1                          ;IF BASE---PROCESS
(1)            000200                   MICPC=MICPC+1
(1)  016722  102211                     <JUMP!BR0CON!<INSRV1-INIT&3000*4>!<INSRV1-INIT&777/2>>
681  016724                    PROCER:  PSTATE  NIDLE2                  ;RESET PORT STATUS
(1)  016724                             MEM     IMM,<<NIDLE2-INIT&777/2>>
(2)            000201                   MICPC=MICPC+1
(2)  016724  002533                     <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
682  016726                             BRWRTE  IMM,100                 ;CLEAR INPUT CONTROL CSR
(1)            000202                   MICPC=MICPC+1
(1)  016726  000500                     <MOVE!WRTEBR!IMM!<100>>
683  016730                             OUT     BR,AANDB!0INCON         ;;
(1)            000203                   MICPC=MICPC+1
(1)  016730  061260                     <MOVE!WROUTX!BR!<AANDB!0INCON>>
684  016732                             LDMA    IMM,<<RTHRS+3>>         ;ADDRESS ERROR LINK
(1)            000204                   MICPC=MICPC+1
(1)  016732  010177                     <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
```

```
  685  016734
  (1)            000205
  (1)  016734    016402
  686  016736
  (1)            000206
  (1)  016736    002400
  687  016740
  (1)            000207
  (1)  016740    042233
  688  016742
  (1)            000210
  (1)  016742    114524
  689  016744
  (1)            000211
  (1)  016744    060601
  690  016746
  (1)            000212
  (1)  016746    102072
  691  016750
  (1)            000213
  (1)  016750    100601

                 MEMINC  IMM,2
                 MICPC=MICPC+1
                 <MOVE!WRMEM!INCMAR!IMM!<2>>
                 MEM     IMM,0
                 MICPC=MICPC+1
                 <MOVE!WRMEM!IMM!<0>>
                 OUTPUT  MEMX,SELB!OMODEM        ;CLEAR DATA TERMINAL READY
                 MICPC=MICPC+1
                 <MOVE!WROUT!MEMX!<SELB!OMODEM>>
                 ALWAYS  RCEXX                   ;POST THE ERROR - FATAL
                 MICPC=MICPC+1
                 <JUMP!ALCOND!<RCEXX-INIT&3000*4>!<RCEXX-INIT&777/2>>
INSRV1:          BRWRTE  BR,SELA!SP1                     ;INIT MODE?
                 MICPC=MICPC+1
                 <MOVE!WRTEBR!BR!<SELA!SP1>>
                 BRO     BASSRV
                 MICPC=MICPC+1
                 <JUMP!BROCON!<BASSRV-INIT&3000*4>!<BASSRV-INIT&777/2>>
                 ALWAYS  PROCER                  ;NO - PROCEDURE ERROR
                 MICPC=MICPC+1
                 <JUMP!ALCOND!<PROCER-INIT&3000*4>!<PROCER-INIT&777/2>>
```

```
                        .SBTTL  OUTINT---SET UP OUTPUT INTERRUPT [RDYO]
                    OUTINT:
                        PSTATE  PINT2
                        MEM     IMM,<<PINT2-INIT&777/2>>
                        MICPC=MICPC+1
                        <MOVE!WRMEM!IMM!<<PINT2-INIT&777/2>>>
                                                ;COMPLETION
                        LDMA    IMM,NXTINT              ;ADDRESS OF NEXT INTERRUPT POINTER
                        MICPC=MICPC+1
                        <MOVE!LDMAR!IMM!<NXTINT&377>>
                        LDMA    MEMX,SELB               ;NEXT INTERRUPT
                        MICPC=MICPC+1
                        <MOVE!LDMAR!MEMX!<SELB>>
                        SP      IBUS,OCON,SPO           ;READ THE OUTPUT CONTROL CSR
                        MICPC=MICPC+1
                        <MOVE!SPX!IBUS!OCON!SPO>
                        OUT     <MEMX!INCMAR>,<AORB!OOCON>      ;WRITE THE OUT CONTROL CSR
                        MICPC=MICPC+1
                        <MOVE!WROUTX!MEMX!INCMAR!<AORB!OOCON>>
                        LDMA    MEMX,SELB                       ;ADDRESS LINK
                        MICPC=MICPC+1
                        <MOVE!LDMAR!MEMX!<SELB>>
                        BRWRTE  <BR!INCMAR>,<AA!SPO>            ;KICK PAST LINK STATUS BYTE
                        MICPC=MICPC+1
                        <MOVE!WRTEBR!BR!INCMAR!<AA!SPO>>
                                                                ;SHIFT CSRO IMAGE LEFT
                                                        ;***DO NOT CHANGE BR UNTIL BR7***
                        OUT     <MEMX!INCMAR>,<SELB!OPORT1>     ;WRITE LOW BYTE OF BA TO CSR
                        MICPC=MICPC+1
                        <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT1>>
                        OUT     <MEMX!INCMAR>,<SELB!OPORT2>     ;WRITE HIGH BYTE OF BA TO CSR
                        MICPC=MICPC+1
                        <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT2>>
                        OUT     <MEMX!INCMAR>,<SELB!OPORT4>     ;WRITE HIGH BYTE OF COUNT TO CSR
                        MICPC=MICPC+1
                        <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT4>>
                        OUT     <MEMX!INCMAR>,<SELB!OPORT3>     ;WRITE THE LOW BYTE OF COUNT
                        MICPC=MICPC+1
                        <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT3>>
                                                        ;***HERE IS BR7***
                        BR7     PE1                     ;INTERRUPT ENABLE IS SET
                        MICPC=MICPC+1
                        <JUMP!BR7CON!<PE1-INIT&3000*4>!<PE1-INIT&777/2>>
                                                        ;GENERATE AN INTERRUPT
                        ALWAYS  IDLE
                        MICPC=MICPC+1
                        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
                    PINT2:  PSTATE  OUTWAIT
                        MEM     IMM,<<OUTWAIT-INIT&777/2>>
                        MICPC=MICPC+1
                        <MOVE!WRMEM!IMM!<<OUTWAIT-INIT&777/2>>>
                        LDMA    IMM,NXTINT              ;ADDRESS NEXT INTERRUPT QUEUE
                        MICPC=MICPC+1
                        <MOVE!LDMAR!IMM!<NXTINT&377>>
                        SP      MEMX,SELB,SPO           ;COPY ADDRESS FOR NEXT INT TO SPO
                        MICPC=MICPC+1
```

```
(1)  017010  043220                    <MOVE!SPX!MEMX!SELB!SPO>
726  017012                            MEM     IMM,INTSTK              ;ASSUME WRAP AROUND CASE
(1)            000234                  MICPC=MICPC+1
(1)  017012  002642                    <MOVE!WRMEM!IMM!<INTSTK>>
727  017014                            BRWRTE  IMM,<<MMEND-2>>         ;ADDRESS OF LAST INT IN STACK
(1)            000235                  MICPC=MICPC+1
(1)  017014  000776                    <MOVE!WRTEBR!IMM!<<MMEND-2>>>
728  017016                            CMP     BR,SPO                  ;SHOULD WE WRAP
(1)            000236                  MICPC=MICPC+1
(1)  017016  060360                    <SUBTC!BR!SPO>
729  017020                            Z       5$                      ;YES--BRANCH
(1)            000237                  MICPC=MICPC+1
(1)  017020  101642                    <JUMP!ZCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
730  017022                            BRWRTE  IMM,2                   ;OFFSET FOR NEXT POINTER
(1)            000240                  MICPC=MICPC+1
(1)  017022  000402                    <MOVE!WRTEBR!IMM!<2>>
731  017024                            MEM     BR,ADD!SPO              ;UPDATE POINTER
(1)            000241                  MICPC=MICPC+1
(1)  017024  062400                    <MOVE!WRMEM!BR!<ADD!SPO>>
732  017026                    5$:     SP      MEMX,SELB,SPO           ;COPY POINTER TO SPO
(1)            000242                  MICPC=MICPC+1
(1)  017026  043220                    <MOVE!SPX!MEMX!SELB!SPO>
733  017030                            LDMA    IMM,NXTSP               ;PICK UP START OF IN QUEUE
(1)            000243                  MICPC=MICPC+1
(1)  017030  010241                    <MOVE!LDMAR!IMM!<NXTSP&377>>
734  017032                            CMP     MEMX,SPO                ;COMPARE TO END
(1)            000244                  MICPC=MICPC+1
(1)  017032  040360                    <SUBTC!MEMX!SPO>
735  017034                            Z       10$                     ;IF EQUAL--CLEAR INT PENDING
(1)            000245                  MICPC=MICPC+1
(1)  017034  101647                    <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
736  017036                            ALWAYS  IDLE
(1)            000246                  MICPC=MICPC+1
(1)  017036  100451                    <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
737  017040                    10$:    BRWRTE  IMM,357                 ;MASK TO CLEAR INT PENDING
(1)            000247                  MICPC=MICPC+1
(1)  017040  000757                    <MOVE!WRTEBR!IMM!<357>>
738  017042                    CLRIDL: SP      BR,AANDB,SP1
(1)            000250                  MICPC=MICPC+1
(1)  017042  063261                    <MOVE!SPX!BR!AANDB!SP1>
739  017044                            ALWAYS  IDLE
(1)            000251                  MICPC=MICPC+1
(1)  017044  100451                    <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
                                                .SBTTL  OUTWAI--WAIT FOR RDYO TO GO AWAY
741
742  017046                             OUTWAI: SPBR    IBUS,OCON,SPO           ;READ OUTPUT CONTROL CSR
(1)            000252                          MICPC=MICPC+1
(1)  017046    123440                          <MOVE!SPBRX!IBUS!OCON!SPO>
747  017050                                    BR7     IDLE
(1)            000253                          MICPC=MICPC+1
(1)  017050    103451                          <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
749  017052                                    BRWRTE  IMM,100         ;CLEAR CONTROL BITS
(1)            000254                          MICPC=MICPC+1
(1)  017052    000500                          <MOVE!WRTEBR!IMM!<100>>
750  017054                                    OUT     BR,OOCON!AANDB
(1)            000255                          MICPC=MICPC+1
(1)  017054    061262                          <MOVE!WROUTX!BR!<OOCON!AANDB>>
751  017056                                    ALWAYS  INS13
(1)            000256                          MICPC=MICPC+1
(1)  017056    100671                          <JUMP!ALCOND!<INS13-INIT&3000*4>!<INS13-INIT&777/2>>
```

```
753
754  017060
(1)                000257
(1)  017060  123560
755  017062
(1)                000260
(1)  017062  001620
756  017064
(1)                000261
(1)  017064  102754
757  017066
(1)                000262
(1)  017066  002113
758  017070
(1)                000263
(1)  017070  060600
759  017072
(1)                000264
(1)  017072  102273
760  017074
(1)                000265
(1)  017074  123000
761  017076
(1)                000266
(1)  017076  000500
762  017100
(1)                000267
(1)  017100  061260
763  017102
(1)                000270
(1)  017102  010211
764  017104
(1)  017104
(2)                000271
(2)  017104  002533
765  017106
(1)                000272
(1)  017106  100451
766
767  017110                CBOOT:  BRWRTE  IMM,200                 ;MASK FOR BOOT MODE
(1)                000273
(1)  017110  000600
768  017112
(1)                000274
(1)  017112  063301
769  017114
(1)                000275
(1)  017114  000604
770  017116
(1)                000276
(1)  017116  063230
771  017120
(1)                000277
(1)  017120  100665

                      .SBTTL  CTLSRV--CNTL I SERVICE
CTLSRV:  SPBR    IBUS,PORT4,SPO                 ;TO SPO
         MICPC=MICPC+1
         <MOVE!SPBRX!IBUS!PORT4!SPO>
         BRSHFT
         MICPC=MICPC+1
         <MOVE!SHFTBR!WRTEBR!SELB>
         BR1     HDSEL                          ;IF SET IS HALF DUPLEX
         MICPC=MICPC+1
         <JUMP!BRICON!<HDSEL-INIT&3000*4>!<HDSEL-INIT&777/2>>
         OUTPUT  IMM,<100!OMODEM>               ;MASK DTR, TURN OFF HDX
         MICPC=MICPC+1
         <MOVE!WROUT!IMM!<100!OMODEM>>
INS11:   BRWRTE  DP,<SELA!SPO>                  ;RESTORE THE CNTL WORD
         MICPC=MICPC+1
         <MOVE!WRTEBR!DP!<SELA!SPO>>
         BR0     CBOOT                          ;IF SET IS BOOT
         MICPC=MICPC+1
         <JUMP!BR0CON!<CBOOT-INIT&3000*4>!<CBOOT-INIT&777/2>>
INS12:   SP      IBUS,INCON,SPO                 ;READ THE INPUT CONTROL CSR
         MICPC=MICPC+1
         <MOVE!SPX!IBUS!INCON!SPO>
         BRWRTE  IMM,100                        ;ZERO THE BR REGISTER EXCEPT INT ENABLE
         MICPC=MICPC+1
         <MOVE!WRTEBR!IMM!<100>>
         OUT     BR,<AANDB!OINCON>              ;CLEAR IN CONTROL CSR
         MICPC=MICPC+1
         <MOVE!WROUTX!BR!<AANDB!OINCON>>
         LDMA    IMM,PRTST                      ;ADDRESS PORT STATE
         MICPC=MICPC+1
         <MOVE!LDMAR!IMM!<PRTST&377>>
INS13:   PSTATE  NIDLE2
         MEM     IMM,<<NIDLE2-INIT&777/2>>
         MICPC=MICPC+1
         <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
         ALWAYS  IDLE
         MICPC=MICPC+1
         <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
         .
         SP      BR,AORB,SP1                    ;IN PORT STATUS WORD
         MICPC=MICPC+1
         <MOVE!SPX!BR!AORB!SP1>
         BRWRTE  IMM,204                        ;MASK FOR OK TO SEND AND LINE IDLE
         MICPC=MICPC+1
         <MOVE!WRTEBR!IMM!<204>>
         SP      BR,SELB,SP10                   ;IN LINE STATUS
         MICPC=MICPC+1
         <MOVE!SPX!BR!SELB!SP10>
         ALWAYS  INS12
         MICPC=MICPC+1
         <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
```

```
                                 .SBTTL  TBASRV--TRANSMITTER BUFFER ADDRESS SERVICE
     017122          TBASRV:    LDMA    IMM,ETC                        ;GET POINTER TO END OF TMT CHAIN
(1)          000300             MICPC=MICPC+1
(1)  017122  010070             <MOVE!LDMAR!IMM!<ETC&377>>
     017124                     LDMA    MEMX,<SELB!SPX!SPO>            ;FIND THE LINK
(1)          000301             MICPC=MICPC+1
(1)  017124  053220             <MOVE!LDMAR!MEMX!<SELB!SPX!SPO>>
     017126                     MEMINC  IMM,1                          ;BUFFER ASSIGNED IN IN LINK FLAGS
(1)          000302             MICPC=MICPC+1
(1)  017126  016401             <MOVE!WRMEM!INCMAR!IMM!<1>>
     017130                     BRWRTE  <IMM!INCMAR>,TMLB                    ;POINT PAST NUMBER FIELD
(1)          000303             MICPC=MICPC+1
(1)  017130  014543             <MOVE!WRTEBR!IMM!INCMAR!<TMLB>>
                                                                       ;SET BR FOR ADDITION TO SPO
     017132                     MEMINC  IBUS,PORT1
(1)          000304             MICPC=MICPC+1
(1)  017132  136500             <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
     017134                     MEMINC  IBUS,PORT2
(1)          000305             MICPC=MICPC+1
(1)  017134  136520             <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
     017136                     MEMINC  IBUS,PORT4
(1)          000306             MICPC=MICPC+1
(1)  017136  136560             <MOVE!WRMEM!INCMAR!IBUS!<PORT4>>
     017140                     MEMINC  IBUS,PORT3
(1)          000307             MICPC=MICPC+1
(1)  017140  136540             <MOVE!WRMEM!INCMAR!IBUS!<PORT3>>
     017142                     LDMA    IMM,ETC
(1)          000310             MICPC=MICPC+1
(1)  017142  010070             <MOVE!LDMAR!IMM!<ETC&377>>
     017144                     MEM     IMM,TML1                       ;ASSUME QUEUE WRAP AROUND
(1)          000311             MICPC=MICPC+1
(1)  017144  002471             <MOVE!WRMEM!IMM!<TML1>>
     017146                     CMP     BR,SPO                         ;END OF CHAIN?
(1)          000312             MICPC=MICPC+1
(1)  017146  060360             <SUBTC!BR!SPO>
     017150                     Z       10$                            ;IF YES--BRANCH
(1)          000313             MICPC=MICPC+1
(1)  017150  101716             <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
     017152                     BRWRTE  IMM,6                          ;QUEUE ENTRY LENGTH
(1)          000314             MICPC=MICPC+1
(1)  017152  000406             <MOVE!WRTEBR!IMM!<6>>
     017154                     MEM     BR,ADD!SPO                     ;UPDATE THE END POINTER IN MEMORY
(1)          000315             MICPC=MICPC+1
(1)  017154  062400             <MOVE!WRMEM!BR!<ADD!SPO>>
     017156          10$:       BRWRTE  IMM,2                ;NUMBERED MSG PENDING MASK
(1)          000316             MICPC=MICPC+1
(1)  017156  000402             <MOVE!WRTEBR!IMM!<2>>
     017160                     SP      BR,AORB,SP10         ;UPDATE LINE STATUS
(1)          000317             MICPC=MICPC+1
(1)  017160  063310             <MOVE!SPX!BR!AORB!SP10>
     017162                     ALWAYS  INS12
(1)          000320             MICPC=MICPC+1
(1)  017162  100665             <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
```

```
                                .SBTTL  RBASRV--RECEIVE BUFFER ADDRESS SERVICE
793
794  017164                     RBASRV: LDMA    IMM,ERC                         ;ADDRES END OF RECEIVE CHAIN
(1)            000321                  MICPC=MICPC+1
(1)  017164    010023                  <MOVE!LDMAR!IMM!<ERC&377>>
795  017166                            LDMA    MEMX,<SELB!SPX!SP0>             ;GET THE POINTER TO LINK
(1)            000322                  MICPC=MICPC+1
(1)  017166    053220                  <MOVE!LDMAR!MEMX!<SELB!SPX!SP0>>
796  017170                            MEMINC  IMM,1
(1)            000323                  MICPC=MICPC+1
(1)  017170    016401                  <MOVE!WRMEM!INCMAR!IMM!<1>>
797  017172                            MEMINC  IBUS,PORT1
(1)            000324                  MICPC=MICPC+1
(1)  017172    136500                  <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
798  017174                            MEMINC  IBUS,PORT2
(1)            000325                  MICPC=MICPC+1
(1)  017174    136520                  <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
799  017176                            MEMINC  IBUS,PORT4
(1)            000326                  MICPC=MICPC+1
(1)  017176    136560                  <MOVE!WRMEM!INCMAR!IBUS!<PORT4>>
800  017200                            MEMINC  IBUS,PORT3
(1)            000327                  MICPC=MICPC+1
(1)  017200    136540                  <MOVE!WRMEM!INCMAR!IBUS!<PORT3>>
801                                     ;;;;NOTE INVERTED ORDER OF PORT 3 AND PORT4
802  017202                            LDMA    IMM,ERC
(1)            000330                  MICPC=MICPC+1
(1)  017202    010023                  <MOVE!LDMAR!IMM!<ERC&377>>
803  017204                            MEM     IMM,RCL1                        ;ASSUME WRAP AROUND CASE
(1)            000331                  MICPC=MICPC+1
(1)  017204    002424                  <MOVE!WRMEM!IMM!<RCL1>>
804  017206                            BRWRTE  IMM,RCL7                        ;GET ADDRESS OF END OF CAHIN AREA
(1)            000332                  MICPC=MICPC+1
(1)  017206    000462                  <MOVE!WRTEBR!IMM!<RCL7>>
805  017210                            CMP     BR,SP0                                  ;CHECK FOR END
(1)            000333                  MICPC=MICPC+1
(1)  017210    060360                  <SUBTC!BR!SP0>
806  017212                            Z       INS12                                   ;IF EQUAL BRANCH
(1)            000334                  MICPC=MICPC+1
(1)  017212    101665                  <JUMP!ZCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
807  017214                            BRWRTE  IMM,5                           ;CALCULATE ADDRESS OF NEXT LINK
(1)            000335                  MICPC=MICPC+1
(1)  017214    000405                  <MOVE!WRTEBR!IMM!<5>>
808  017216                            MEM     BR,ADD!SP0                      ;..
(1)            000336                  MICPC=MICPC+1
(1)  017216    062400                  <MOVE!WRMEM!BR!<ADD!SP0>>
809  017220                            ALWAYS  INS12                                   ;EXIT
(1)            000337                  MICPC=MICPC+1
(1)  017220    100665                  <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
810  017222                     RA1:   BRWRTE  IMM,317                         ;MASK TO CLEAR START MODE AND CLR ACTIVE
(1)            000340                  MICPC=MICPC+1
(1)  017222    000717                  <MOVE!WRTEBR!IMM!<317>>
811  017224                            SPBR    BR,AANDB,SP10                   ;CLEAR BIT IN LINE STATUS WORD
(1)            000341                  MICPC=MICPC+1
(1)  017224    063670                  <MOVE!SPBRX!BR!AANDB!SP10>
812  017226                     RA3:   BRWRTE  IMM,0                           ;CLEAR BR
(1)            000342                  MICPC=MICPC+1
(1)  017226    000400                  <MOVE!WRTEBR!IMM!<0>>
```

```
  813  017230                        SP      BR,SELB,SP13            ;SET NUMB MESSAGE TYPE IN SP13
  (1)          000343                MICPC=MICPC+1
  (1)  017230  063233                <MOVE!SPX!BR!SELB!SP13>
  814  017232                        STATE   RCVB                    ;CHANGE RECEIVE STATE POINTER TO STATE B
  (1)          000344                MICPC=MICPC+1
  (1)  017232  000424                <MOVE!WRTEBR!IMM!<RCVB-INIT&777/2>>
  815  017234                        ALWAYS  REXIT
  (1)          000345                MICPC=MICPC+1
  (1)  017234  100450                <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
  816                                ;
  818  017236                ACK:    BRWRTE  BR,AA!SP10              ;READ LINE STATUS SHIFTING LEFT
  (1)          000346                MICPC=MICPC+1
  (1)  017236  060530                <MOVE!WRTEBR!BR!<AA!SP10>>
  819  017240                        BR4     5$                      ;IF START RECD--CLEAR START MODE
  (1)          000347                MICPC=MICPC+1
  (1)  017240  103351                <JUMP!BR4CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
  820  017242                        ALWAYS  IDLE
  (1)          000350                MICPC=MICPC+1
  (1)  017242  100451                <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
  821  017244                5$:     BRWRTE  IMM,327                 ;CLEAR START MODE
  (1)          000351                MICPC=MICPC+1
  (1)  017244  000727                <MOVE!WRTEBR!IMM!<327>>
  822  017246                        SP      BR,AANDB,SP10           ;IN LINE STATUS
  (1)          000352                MICPC=MICPC+1
  (1)  017246  063270                <MOVE!SPX!BR!AANDB!SP10>
  823  017250                        ALWAYS  RD5
  (1)          000353                MICPC=MICPC+1
  (1)  017250  104507                <JUMP!ALCOND!<RD5-INIT&3000*4>!<RD5-INIT&777/2>>
```

```
   826  017252
   (1)            000354
   (1)  017252    000500  HDSEL:  BRWRTE  IMM,100                         ;HD MASK TO BR
                          MICPC=MICPC+1
                          <MOVE!WRTEBR!IMM!<100>>
   827  017254                  SP      BR,AORB,SP10                     ;UPDATE PORT STATUS WORD
   (1)            000355         MICPC=MICPC+1
   (1)  017254    063310         <MOVE!SPX!BR!AORB!SP10>
   828  017256                  ALWAYS  INS11
   (1)            000356         MICPC=MICPC+1
   (1)  017256    100663         <JUMP!ALCOND!<INS11-INIT&3000*4>!<INS11-INIT&777/2>>
   829                    ;
   830  017260            PE1:  BRWRTE  IMM,300                  ;MASK FOR INTERUPT AND VECTOR THROUGH X04
   (1)            000357         MICPC=MICPC+1
   (1)  017260    000700         <MOVE!WRTEBR!IMM!<300>>
   831  017262                  SP      IBUS,UBBR,SP0            ;READ BR CONTROL REG
   (1)            000360         MICPC=MICPC+1
   (1)  017262    123220         <MOVE!SPX!IBUS!UBBR!SP0>
   832  017264                  OUT     BR,<AORB!OBR>            ;INTERRUPT
   (1)            000361         MICPC=MICPC+1
   (1)  017264    061311         <MOVE!WROUTX!BR!<AORB!OBR>>
   834  017266                  ALWAYS  IDLE
   (1)            000362         MICPC=MICPC+1
   (1)  017266    100451         <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
   839                    ;
   840  017270            HALTED: MEMADR  EM6
   (1)            000363         MICPC=MICPC+1
   (1)  017270    002722         <MOVE!WRMEM!<EM6-INIT&777/2>>
   841                                                             ;FALL INTO ACLOW
   842  017272            ACLOW:  BRWRTE  IMM,2                    ;CAUSE AN AC LOW
   (1)            000364         MICPC=MICPC+1
   (1)  017272    000402         <MOVE!WRTEBR!IMM!<2>>
   843  017274                  OUT     BR,<SELB!OBR>
   (1)            000365         MICPC=MICPC+1
   (1)  017274    061231         <MOVE!WROUTX!BR!<SELB!OBR>>
   844  017276            5$:   BRWRTE  IBUS,UBBR                  ;WAIT FOR IT TO COMPLETE
   (1)            000366         MICPC=MICPC+1
   (1)  017276    120620         <MOVE!WRTEBR!IBUS!<UBBR>>
   845  017300                  BR1     5$
   (1)            000367         MICPC=MICPC+1
   (1)  017300    102766         <JUMP!BR1CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
   846  017302                  .ALWAY  MEMX,SELB,PAGE3
   (1)            000370         MICPC=MICPC+1
   (1)  017302    154620         <JUMP!ALCOND!MEMX!SELB!PAGE3>
   847  017304            CKTIME: BRWRTE  IBUS,UBBR                ;READ BR CONTROL REG
   (1)            000371         MICPC=MICPC+1
   (1)  017304    120620         <MOVE!WRTEBR!IBUS!<UBBR>>
   848  017306                  BR4     HALTED
   (1)            000372         MICPC=MICPC+1
   (1)  017306    103363         <JUMP!BR4CON!<HALTED-INIT&3000*4>!<HALTED-INIT&777/2>>
   849  017310                  ALWAYS  EM1
   (1)            000373         MICPC=MICPC+1
   (1)  017310    114725         <JUMP!ALCOND!<EM1-INIT&3000*4>!<EM1-INIT&777/2>>
   850                    ;
   851  017312            TBU1:   BRWRTE  IBUS,NPR
   (1)            000374         MICPC=MICPC+1
   (1)  017312    120600         <MOVE!WRTEBR!IBUS!<NPR>>
   852  017314                  BR0     IDLE
```

```
(1)             000375          MICPC=MICPC+1
(1)  017314     102051          <JUMP!BROCON!<IDLE-INIT&3C00*4>!<IDLE-INIT&777/2>>
853  017316                     ALWAYS  EC2
(1)             000376          MICPC=MICPC+1
(1)  017316     114752          <JUMP!ALCOND!<EC2-INIT&3000*4>!<EC2-INIT&777/2>>
854  017320                     $ZERO
(1)             000377          MICPC=MICPC+1
(1)  017320     000000          000000
855
```

```
                        .=INIT+1000
                        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.
 017322         RCVA:   SP      IBUS,RCVDAT,SPO         ;READ RECEIVE CHARACTER TO SPO
(1)  000400             MICPC=MICPC+1
(1)  017322  023200     <MOVE!SPX!IBUS!RCVDAT!SPO>
 017324                 BRWRTE  BR,SELA!SP1             ;READ PORT STATUS WORD
(1)  000401             MICPC=MICPC+1
(1)  017324  060601     <MOVE!WRTEBR!BR!<SELA!SP1>>
 017326                 BRO     5$                      ;IF INIT MODE---ONLY BOOT OK
(1)  000402             MICPC=MICPC+1
(1)  017326  106012     <JUMP!BROCON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
 017330                 BR7     5$                      ;IF BOOT MODE---ONLY BOOT OK
(1)  000403             MICPC=MICPC+1
(1)  017330  107412     <JUMP!BR7CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
 017332                 BRWRTE  IMM,201                 ;SOH TO BR
(1)  000404             MICPC=MICPC+1
(1)  017332  000601     <MOVE!WRTEBR!IMM!<201>>
 017334                 CMP     BR,SPO                  ;COMPARE BR TO SPO
(1)  000405             MICPC=MICPC+1
(1)  017334  060360     <SUBTC!BR!SPO>
 017336                 Z       RA1                     ;IF EQUAL-IS NUMBERED MESSAGE
(1)  000406             MICPC=MICPC+1
(1)  017336  101740     <JUMP!ZCOND!<RA1-INIT&3000*4>!<RA1-INIT&777/2>>
 017340                 BRWRTE  IMM,5                   ;ENQ TO BR
(1)  000407             MICPC=MICPC+1
(1)  017340  000405     <MOVE!WRTEBR!IMM!<5>>
 017342                 CMP     BR,SPO                  ;COMPARE ENQ TO SPO
(1)  000410             MICPC=MICPC+1
(1)  017342  060360     <SUBTC!BR!SPO>
 017344                 Z       RA2                     ;IF EQUAL-IS UNNUMBERED MESSAGE
(1)  000411             MICPC=MICPC+1
(1)  017344  105422     <JUMP!ZCOND!<RA2-INIT&3000*4>!<RA2-INIT&777/2>>
 017346         5$:     BRWRTE  IMM,220                 ;DLE TO BR
(1)  000412             MICPC=MICPC+1
(1)  017346  000620     <MOVE!WRTEBR!IMM!<220>>
 017350                 CMP     BR,SPO                  ;COMPARE DLE TO SPO
(1)  000413             MICPC=MICPC+1
(1)  017350  060360     <SUBTC!BR!SPO>
 017352                 Z       BOOT                    ;IF EQUAL IS BOOT
(1)  000414             MICPC=MICPC+1
(1)  017352  105756     <JUMP!ZCOND!<BOOT-INIT&3000*4>!<BOOT-INIT&777/2>>
 017354         FLUSH:  OUTPUT  IMM,<200!ORCVC0>        ;FLUSH INPUT SILO
(1)  000415             MICPC=MICPC+1
(1)  017354  002212     <MOVE!WROUT!IMM!<200!ORCVC0>>
 017                                                    ;(LOW ORDER BITS READ ONLY)
 017356                 BRWRTE  IMM,357                 ;MASK TO CLEAR--CLEAR ACTIVE
(1)  000416             MICPC=MICPC+1
(1)  017356  000757     <MOVE!WRTEBR!IMM!<357>>
 017360                 SP      BR,AANDB,SP10           ;IN LINE STATUS WORD
(1)  000417             MICPC=MICPC+1
(1)  017360  063270     <MOVE!SPX!BR!AANDB!SP10>
 017362         RM1:    STATE   RCVA
```

```
(1)             000420          MICPC=MICPC+1
(1)   017362    000400          <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
885   017364                    ALWAYS  REXIT
(1)             000421          MICPC=MICPC+1
(1)   017364    100450          <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
887   017366           RA2:     STATE   RCVI                    ;CHANGE RECEIVE STATE TO I
(1)             000422          MICPC=MICPC+1
(1)   017366    000665          <MOVE!WRTEBR!IMM!<RCVI-INIT&777/2>>
889   017370                    ALWAYS  REXIT
(1)             000423          MICPC=MICPC+1
(1)   017370    100450          <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
```

```
                        .SBTTL  RCVB--ROUTINE TO HANDLE FIRST CHARACTER OF COUNT FIELD
                        ;ENTERED FROM IDLE LOOP
                        ;STORES COUNT FIELD AND SETS UP RCVC AS NEXT STATE
                        ;
017372                  RCVB:
017372                          SP      IBUS,RCVDAT,SP4         ;READ CHARACTER TO SP4
(1)     000424                  MICPC=MICPC+1
(1)     017372  023204          <MOVE!SPX!IBUS!RCVDAT!SP4>
017374                          LDMA    BR,<SELA!SP14>                  ;LOAD MAR WITH ADDRESS OF CURRENT BA
(1)     000425                  MICPC=MICPC+1
(1)     017374  070214          <MOVE!LDMAR!BR!<SELA!SP14>>
017376                          BRWRTE  MEMX,INCMAR!SELB                ;READ FLAGS BYTE
(1)     000426                  MICPC=MICPC+1
(1)     017376  054620          <MOVE!WRTEBR!MEMX!<INCMAR!SELB>>
017400                          BRO     RB1                             ;RECV BUFFER ASSIGNED---CONTINUE
(1)     000427                  MICPC=MICPC+1
(1)     017400  106041          <JUMP!BROCON!<RB1-INIT&3000*4>!<RB1-INIT&777/2>>
017402                          BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
(1)     000430                  MICPC=MICPC+1
(1)     017402  060601          <MOVE!WRTEBR!BR!<SELA!SP1>>
017404                          BR7     RB3                             ;MAINT MODE
(1)     000431                  MICPC=MICPC+1
(1)     017404  107437          <JUMP!BR7CON!<RB3-INIT&3000*4>!<RB3-INIT&777/2>>
017406                          LDMA    IMM,T                   ;ERROR--LOAD TYPE FIELD ADDRESS IN MAR
(1)     000432                  MICPC=MICPC+1
(1)     017406  010151          <MOVE!LDMAR!IMM!<T&377>>
017410                          MEMINC  IMM,2                   ;LOAD NAK TYPE
(1)     000433                  MICPC=MICPC+1
(1)     017410  016402          <MOVE!WRMEM!INCMAR!IMM!<2>>
017412                          MEM     IMM,310                 ;LOAD SUB-TYPE NO BUFFERS
(1)     000434                  MICPC=MICPC+1
(1)     017412  002710          <MOVE!WRMEM!IMM!<310>>
017414                          LDMA    IMM,NTLS
(1)     000435                  MICPC=MICPC+1
(1)     017414  010012          <MOVE!LDMAR!IMM!<NTLS&377>>
017416                          ALWAYS  RH5                     ;BRANCH TO SEND NAK ROUTINE
(1)     000436                  MICPC=MICPC+1
(1)     017416  104552          <JUMP!ALCOND!<RH5-INIT&3000*4>!<RH5-INIT&777/2>>
017420                  RB3:    BRWRTE  IMM,4                   ;MASK FOR NO BUFFER AVAILABLE
(1)     000437                  MICPC=MICPC+1
(1)     017420  000404          <MOVE!WRTEBR!IMM!<4>>
017422                          SP      BR,AORB,SP1             ;SET THE FLAG
(1)     000440                  MICPC=MICPC+1
(1)     017422  063301          <MOVE!SPX!BR!AORB!SP1>
017424                  RB1:    STATE   RCVC
(1)     000441                  MICPC=MICPC+1
(1)     017424  000461          <MOVE!WRTEBR!IMM!<RCVC-INIT&777/2>>
017426                  RB0:    SP      BR,SELB,SP3
(1)     000442                  MICPC=MICPC+1
(1)     017426  063223          <MOVE!SPX!BR!SELB!SP3>
017430                          OUTPUT  <MEMX!INCMAR>,<SELB!OBA1>       ;OUTPUT LOW ORDER BYTE OF ADDRESS
(1)     000443                  MICPC=MICPC+1
(1)     017430  056226          <MOVE!WROUT!MEMX!INCMAR!<SELB!OBA1>>
017432                          OUTPUT  MEMX!INCMAR,<SELB!OBA2> ;OUTPUT HIGH BYTE OF ADDRESS
(1)     000444                  MICPC=MICPC+1
(1)     017432  056227          <MOVE!WROUT!MEMX!INCMAR!<SELB!OBA2>>
```

```
 918  017434
(1)             000445
(1)   017434    123220          SP      IBUS,UBBR,SPO           ;READ THE BUS REQ REGISTER
                                MICPC=MICPC+1
                                <MOVE!SPX!IBUS!UBBR!SPO>
 919  017436
(1)             000446          BRWRTE  IMM,101                 ;MASK OFF ALL BUT NXM AND VEC4 BITS
(1)   017436    000501          MICPC=MICPC+1
                                <MOVE!WRTEBR!IMM!<101>>
 920  017440
(1)             000447          SP      BR,AANDB,SPO            ;AND SAVE IN SPO
(1)   017440    063260          MICPC=MICPC+1
                                <MOVE!SPX!BR!AANDB!SPO>
 921  017442
(1)             000450          SP      IMM,300,SP5             ;MASK TO ISOLATE EX. MEM BITS
(1)   017442    003305          MICPC=MICPC+1
                                <MOVE!SPX!IMM!300!SP5>
 922                                                            ;NOTE THIS REALLY WRITES A 305 BUT THE
 923                                                            ;5 GETS SHIFTED OUT
 924  017444
(1)             000451          BRWRTE  MEMX,AANDB!SP5          ;MASK ALL BUT EX. MEM BITS
(1)   017444    040665          MICPC=MICPC+1
                                <MOVE!WRTEBR!MEMX!<AANDB!SP5>>
 925  017446                     BRSHFT                          ;SHIFT THEM INTO THE CORRECT POSITION
(1)             000452          MICPC=MICPC+1
(1)   017446    001620          <MOVE!SHFTBR!WRTEBR!SELB>
 926  017450                     BRSHFT
(1)             000453          MICPC=MICPC+1
(1)   017450    001620          <MOVE!SHFTBR!WRTEBR!SELB>
 927  017452                     BRSHFT
(1)             000454          MICPC=MICPC+1
(1)   017452    001620          <MOVE!SHFTBR!WRTEBR!SELB>
 928  017454                     BRSHFT
(1)             000455          MICPC=MICPC+1
(1)   017454    001620          <MOVE!SHFTBR!WRTEBR!SELB>
 929  017456                     OUT     BR,AORB!OBR                     ;WRITE EX MEM BITS OUT
(1)             000456          MICPC=MICPC+1
(1)   017456    061311          <MOVE!WROUTX!BR!<AORB!OBR>>
 930  017460                     ALWAYS  IDLE
(1)             000457          MICPC=MICPC+1
(1)   017460    100451          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 931  017462             RB2:    ALWAYS  I2
(1)             000460          MICPC=MICPC+1
(1)   017462    100456          <JUMP!ALCOND!<I2-INIT&3000*4>!<I2-INIT&777/2>>
```

```
                        .SBTTL  RCVC--ROUTINE TO HANDLE SECOND CHARACTER OF COUNT FIELD, SELECT AND FINA
                        ;ENTERED FROM IDLE LOOP
                        ;INTERPRETS SELECT AND FINAL
                        ;CHECKS FOR COUNT TOO LARGE
                        ;
 933
 934
 935
 936
 937
 938  017464           RCVC:
 943  017464                   SP      IBUS,RCVDAT,SP5         ;GET CHARACTER
(1)                       000461 MICPC=MICPC+1
(1)   017464   023205    <MOVE!SPX!IBUS!RCVDAT!SP5>
 944  017466                   BRWRTE  IMM,200                 ;SEPARATE SELECT BIT FROM COUNT
(1)                       000462 MICPC=MICPC+1
(1)   017466   000600    <MOVE!WRTEBR!IMM!<200>>
 945  017470                   BRWRTE  BR,AANDB!SP5
(1)                       000463 MICPC=MICPC+1
(1)   017470   060665    <MOVE!WRTEBR!BR!<AANDB!SP5>>
 946  017472                   SP      BR,AORB,SP10
(1)                       000464 MICPC=MICPC+1
(1)   017472   063310    <MOVE!SPX!BR!AORB!SP10>
 947  017474                   LDMA    IMM,BC                  ;LOAD MAR TO BYTE COUNT
(1)                       000465 MICPC=MICPC+1
(1)   017474   010167    <MOVE!LDMAR!IMM!<BC&377>>
 948  017476                   MEMINC  BR,SELA!SP4             ;SAVE LOW BYTE
(1)                       000466 MICPC=MICPC+1
(1)   017476   076604    <MOVE!WRMEM!INCMAR!BR!<SELA!SP4>>
 949  017500                   MEMINC  BR,SELA!SP5             ;AND NOW HIGH BYTE
(1)                       000467 MICPC=MICPC+1
(1)   017500   076605    <MOVE!WRMEM!INCMAR!BR!<SELA!SP5>>
 951  017502           RC5:    STATE   RCVD                    ;SET NEXT STATE TO D
(1)                       000470 MICPC=MICPC+1
(1)   017502   000472    <MOVE!WRTEBR!IMM!<RCVD-INIT&777/2>>
 952  017504                   ALWAYS  REXIT
(1)                       000471 MICPC=MICPC+1
(1)   017504   100450    <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
```

```
954
955
956  017506                          RCVD:   .SBTTL  RCVD--ROUTINE TO HANDLE RESPONSE FIELD FOR NUMBERED MESSAGES
                                             ;
                                             STATE   RCVE
(1)             000472               
                                             MICPC=MICPC+1
(1)  017506     000513               
                                             <MOVE!WRTEBR!IMM!<RCVE-INIT&777/2>>
957  017510                          RD2:    SP      BR,SELB,SP3             ;SAVE THE STATE
(1)             000473               
                                             MICPC=MICPC+1
(1)  017510     063223               
                                             <MOVE!SPX!BR!SELB!SP3>
958  017512                                  SPBR    IBUS,RCVDAT,SPO         ;INPUT THE CHARACTER
(1)             000474               
                                             MICPC=MICPC+1
(1)  017512     023600               
                                             <MOVE!SPBRX!IBUS!RCVDAT!SPO>
959  017514                                  BRWRTE  BR,SUB!SP17             ;COMPARE NEW R TO LAST R
(1)             000475               
                                             MICPC=MICPC+1
(1)  017514     060757               
                                             <MOVE!WRTEBR!BR!<SUB!SP17>>
960  017516                                  BR7     10$                     ;IF NEW IS GREATER---PROCESS
(1)             000476               
                                             MICPC=MICPC+1
(1)  017516     107500               
                                             <JUMP!BR7CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
961  017520                                  ALWAYS  IDLE
(1)             000477               
                                             MICPC=MICPC+1
(1)  017520     100451               
                                             <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
962  017522                          10$:    BRWRTE  BR,SELA!SP1             ;READ STATUS BYTE
(1)             000500               
                                             MICPC=MICPC+1
(1)  017522     060601               
                                             <MOVE!WRTEBR!BR!<SELA!SP1>>
963  017524                                  BR7     IDLE                    ;MAINT. MODE - GET OUT
(1)             000501               
                                             MICPC=MICPC+1
(1)  017524     103451               
                                             <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
964  017526                                  BRWRTE  BR,SELA!SP10
(1)             000502               
                                             MICPC=MICPC+1
(1)  017526     060610               
                                             <MOVE!WRTEBR!BR!<SELA!SP10>>
965  017530                                  BRSHFT
(1)             000503               
                                             MICPC=MICPC+1
(1)  017530     001620               
                                             <MOVE!SHFTBR!WRTEBR!SELB>
966  017532                                  BR4     IDLE
(1)             000504               
                                             MICPC=MICPC+1
(1)  017532     103051               
                                             <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
967  017534                                  LDMA    IMM,ISP17               ;ADDRESS LAST ACKED IMAGE
(1)             000505               
                                             MICPC=MICPC+1
(1)  017534     010153               
                                             <MOVE!LDMAR!IMM!<ISP17&377>>
968  017536                                  MEM     BR,SELA!SPO             ;COPY THE CHAR
(1)             000506               
                                             MICPC=MICPC+1
(1)  017536     062600               
                                             <MOVE!WRMEM!BR!<SELA!SPO>>
969  017540                          RD5:    BRWRTE  IMM!LDMAR,REPST         ;SET UP COUNT FOR TIMER
(1)             000507               
                                             MICPC=MICPC+1
(1)  017540     010403               
                                             <MOVE!WRTEBR!IMM!LDMAR!<REPST>>
970                                                                          ;****DEPENDENT ON REPST = 2
971  017542                                  MEM     IMM,1                   ;RESET REP THRESHOLD
(1)             000510               
                                             MICPC=MICPC+1
(1)  017542     002401               
                                             <MOVE!WRMEM!IMM!<1>>
972  017544                                  SP      BR,SELB,SP15            ;RESET THE COUNT
(1)             000511               
                                             MICPC=MICPC+1
(1)  017544     063235               
                                             <MOVE!SPX!BR!SELB!SP15>
973  017546                                  ALWAYS  IDLE
(1)             000512               
                                             MICPC=MICPC+1
(1)  017546     100451               
                                             <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
                        .SBTTL  RCVE--ROUTINE TO HANDLE N FIELD OF NUMBERED MESSAGE
                        ;
RCVE:   BRWRTE  BR,SELA!SP1             ;READ THE STATUS BYTE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR7     RCVQ
        MICPC=MICPC+1
        <JUMP!BR7CON!<RCVQ-INIT&3000*4>!<RCVQ-INIT&777/2>>
        BRWRTE  IBUS,RCVDAT             ;INPUT THE CHARACTER
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<RCVDAT>>
        CMP     BR,SP11
        MICPC=MICPC+1
        <SUBTC!BR!SP11>
        Z       5$
        MICPC=MICPC+1
        <JUMP!ZCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        SP      BR,DECA,SP13            ;FORCE MSG TYPE TO -1
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP13>
        ALWAYS  RE2
        MICPC=MICPC+1
        <JUMP!ALCOND!<RE2-INIT&3000*4>!<RE2-INIT&777/2>>
5$:     SP      BR,INCA,SP11            ;UPDATE R FIELD
        MICPC=MICPC+1
        <MOVE!SPX!BR!INCA!SP11>
RE2:    STATE   RCVF                    ;NEXT RECEIVE STATE IS F
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVF-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
```

```
 988
 989  017574                          .SBTTL  RCVF--ROUTINE TO IGNORE ADDRESS
RCVF:   SP      BR,DECA,SP4             ;DECREMENTLOW BYTE OF COUNT
(1)             000525                  MICPC=MICPC+1
(1)     017574  063164                  <MOVE!SPX!BR!DECA!SP4>
 990  017576                            C       RCVF0                   ;NO OVERFLOW
(1)             000526                  MICPC=MICPC+1
(1)     017576  105130                  <JUMP!CCOND!<RCVF0-INIT&3000*4>!<RCVF0-INIT&777/2>>
 991  017600                            SP      BR,DECA,SP5             ;OVERFLOW - DECREMENT HIGH BYTE
(1)             000527                  MICPC=MICPC+1
(1)     017600  063165                  <MOVE!SPX!BR!DECA!SP5>
 992  017602  RCVF0:  STATE   RCVG
(1)             000530                  MICPC=MICPC+1
(1)     017602  000533                  <MOVE!WRTEBR!IMM!<RCVG-INIT&777/2>>
 993  017604  RCVF1:  NOP     IBUS,RCVDAT,0           ;INPUT CHARACTER - AND DISCARD
(1)             000531                  MICPC=MICPC+1
(1)     017604  020200                  <IBUS!RCVDAT!0>
 994  017606                            ALWAYS  REXIT
(1)             000532                  MICPC=MICPC+1
(1)     017606  100450                  <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
 995                                     ;
 996                                     .SBTTL  RCVG--ROUTINE TO IGNORE CRC1
 997                                     ;
 998  017610  RCVG:   STATE   RCVH                    ;NEXT STATE IS RCVH
(1)             000533                  MICPC=MICPC+1
(1)     017610  000535                  <MOVE!WRTEBR!IMM!<RCVH-INIT&777/2>>
 999  017612                            ALWAYS  RCVF1
(1)             000534                  MICPC=MICPC+1
(1)     017612  104531                  <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
```

```
                                                .SBTTL  RCVH--ROUTINE TO HANDLE CRC2 AND TO DISPATCH NUMBERED AND UNNUMBERED TYP
                                                ;
1001                                    RCVH:
1002                                            SP      IBUS,RCVDAT,SPO                 ;GET CHAR IN SPO
1003  017614                                    MICPC=MICPC+1
1004  017614                                    <MOVE!SPX!IBUS!RCVDAT!SPO>
  (1)           000535                          BRWRTE  IBUS,RCVCON                     ;READ RECVR CONTROL REGISTER
  (1)  017614   023200                          MICPC=MICPC+1
1005  017616                                    <MOVE!WRTEBR!IBUS!<RCVCON>>
  (1)           000536                          BRO     TDON1                           ;IF BCC MATCH SET CRC IS GOOD
  (1)  017616   020640                          MICPC=MICPC+1
1006  017620                                    <JUMP!BROCON!<TDON1-INIT&3000*4>!<TDON1-INIT&777/2>>
  (1)           000537                          BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
  (1)  017620   116165                          MICPC=MICPC+1
1007  017622                                    <MOVE!WRTEBR!BR!<SELA!SP1>>
  (1)           000540                          BR7     RHX                             ;MAINT MODE
  (1)  017622   060601                          MICPC=MICPC+1
1008  017624                                    <JUMP!BR7CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
  (1)           000541                          BRWRTE  DP,<SELA!SP10>                  ;READ PORT STATUS WORD TO BR
  (1)  017624   107740                          MICPC=MICPC+1
1009  017626                                    <MOVE!WRTEBR!DP!<SELA!SP10>>
  (1)           000542                          BRSHFT
  (1)  017626   060610                          MICPC=MICPC+1
1010  017630                                    <MOVE!SHFTBR!WRTEBR!SELB>
  (1)           000543                          BR4     SNAK1                           ;IF START MODE--PROCEED TO RESEND START
  (1)  017630   001620                          MICPC=MICPC+1
1011  017632                                    <JUMP!BR4CON!<SNAK1-INIT&3000*4>!<SNAK1-INIT&777/2>>
  (1)           000544                          LDMA    IMM,T                           ;ELSE BCC ERROR--LOAD ADDRESS OF TYPE FI
  (1)  017632   117307                          MICPC=MICPC+1
1012  017634                                    <MOVE!LDMAR!IMM!<T&377>>
  (1)           000545                          MEMINC  IMM,2                           ;WRITE NAK TYPE
  (1)  017634   010151                          MICPC=MICPC+1
1013  017636                                    <MOVE!WRMEM!INCMAR!IMM!<2>>
  (1)           000546                          MEMINC  IMM,301                         ;WRITE HEADER BCC ERROR SUBTYPE
  (1)  017636   016402                          MICPC=MICPC+1
1014  017640                                    <MOVE!WRMEM!INCMAR!IMM!<301>>
  (1)           000547                          MEM     BR,SELA!SP17                    ;RESTORE LAST ACKED  IMAGE
  (1)  017640   016701                          MICPC=MICPC+1
1015  017642                                    <MOVE!WRMEM!BR!<SELA!SP17>>
  (1)           000550                          LDMA    IMM,NHDS                        ;ADDRESS CUM ERROR COUNTER
  (1)  017642   062617                          MICPC=MICPC+1
1016  017644                                    <MOVE!LDMAR!IMM!<NHDS&377>>
  (1)           000551                  RH5:    SP      MEMX,SELB,SPO                   ;WRITE IT TO SPO
  (1)  017644   010013                          MICPC=MICPC+1
1017  017646                                    <MOVE!SPX!MEMX!SELB!SPO>
  (1)           000552                          MEM     BR,INCA!SPO                     ;INCREMENT IT
  (1)  017646   043220                          MICPC=MICPC+1
1018  017650                                    <MOVE!WRMEM!BR!<INCA!SPO>>
  (1)           000553                          LDMA    IMM,NAKST                       ;ADDRESS NAKS TMTED DYNAMIC
  (1)  017650   062460                          MICPC=MICPC+1
1019  017652                                    <MOVE!LDMAR!IMM!<NAKST&377>>
  (1)           000554                          BRWRTE  MEMX,SELB                       ;WRITE IT TO BR
  (1)  017652   010001                          MICPC=MICPC+1
1020  017654                                    <MOVE!WRTEBR!MEMX!<SELB>>
  (1)           000555                          BSHFTB                                  ;SHIFT IT RIGHT
  (1)  017654   040620                          MICPC=MICPC+1
1021  017656
  (1)           000556
```

```
  (1)  017656  061620          <MOVE!SHFTBR!SELB!BR>
1022   017660                 MEM     BR,SELB                 ;UPDATE IT
  (1)          000557         MICPC=MICPC+1
  (1)  017660  062620         <MOVE!WRMEM!BR!<SELB>>
1023   017662                 BRQ     NTHRES                  ;BRANCH IF THRESHOLD EXCEEDED
  (1)          000560         MICPC=MICPC+1
  (1)  017662  116256         <JUMP!BRQCON!<NTHRES-INIT&3000*4>!<NTHRES-INIT&777/2>>
1024   017664                 ALWAYS  SNAK
  (1)          000561         MICPC=MICPC+1
  (1)  017664  114704         <JUMP!ALCOND!<SNAK-INIT&3000*4>!<SNAK-INIT&777/2>>
1025   017666         RH3:    BRWRTE  DP,<DECA!SP13>          ;LOAD TYPE RECEIVED--DECREMENTING
  (1)          000562         MICPC=MICPC+1
  (1)  017666  060573         <MOVE!WRTEBR!DP!<DECA!SP13>>
1026   017670                 Z       RH1                     ;IF ALUOUT IS ALL ONES IS NUMBERED MSG
  (1)          000563         MICPC=MICPC+1
  (1)  017670  115467         <JUMP!ZCOND!<RH1-INIT&3000*4>!<RH1-INIT&777/2>>
1027   017672                 RSTATE  RCVA
  (1)          000564         MICPC=MICPC+1
  (1)  017672  000400         <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
  (1)          000565         MICPC=MICPC+1
  (1)  017674  063223         <MOVE!SPX!BR!SELB!SP3>
1028   017676                 BRWRTE  DP,<SELA!SP10>          ;LOAD LINE STATUS WORD IN BR
  (1)          000566         MICPC=MICPC+1
  (1)  017676  060610         <MOVE!WRTEBR!DP!<SELA!SP10>>
1034   017700                 OUTPUT  IMM,<200!ORCVCO>
  (1)          000567         MICPC=MICPC+1
  (1)  017700  002212         <MOVE!WROUT!IMM!<200!ORCVCO>>
1036   017702                 BRSHFT                          ;SHIFT RIGHT
  (1)          000570         MICPC=MICPC+1
  (1)  017702  001620         <MOVE!SHFTBR!WRTEBR!SELB>
1037   017704                 BR4     10$
  (1)          000571         MICPC=MICPC+1
  (1)  017704  107177         <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1038   017706                 LDMA    IMM,TYPTAB              ;ADDRESS TYPE TABLE
  (1)          000572         MICPC=MICPC+1
  (1)  017706  010162         <MOVE!LDMAR!IMM!<TYPTAB&377>>
1039   017710                 CMP     <MEMX!INCMAR>,SP13
  (1)          000573         MICPC=MICPC+1
  (1)  017710  054373         <SUBTC!MEMX!INCMAR!SP13>
1040   017712                 Z       REP
  (1)          000574         MICPC=MICPC+1
  (1)  017712  115411         <JUMP!ZCOND!<REP-INIT&3000*4>!<REP-INIT&777/2>>
1041   017714                 CMP     <MEMX!INCMAR>,SP13
  (1)          000575         MICPC=MICPC+1
  (1)  017714  054373         <SUBTC!MEMX!INCMAR!SP13>
1042   017716                 Z       NAK
  (1)          000576         MICPC=MICPC+1
  (1)  017716  115445         <JUMP!ZCOND!<NAK-INIT&3000*4>!<NAK-INIT&777/2>>
1043   017720         10$:    LDMA    IMM,TYPSTT              ;SET POINTER TO START TYPE
  (1)          000577         MICPC=MICPC+1
  (1)  017720  010164         <MOVE!LDMAR!IMM!<TYPSTT&377>>
1044   017722                 CMP     <MEMX!INCMAR>,SP13
  (1)          000600         MICPC=MICPC+1
  (1)  017722  054373         <SUBTC!MEMX!INCMAR!SP13>
1045   017724                 Z       START
  (1)          000601         MICPC=MICPC+1
```

```
  (1)  017724  115420                <JUMP!ZCOND!<START-INIT&3000*4>!<START-INIT&777/2>>
1046
1047  017726                         CMP     <MEMX!INCMAR>,SP13              ;STACK TYPE
  (1)          000602                MICPC=MICPC+1
  (1)  017726  054373                <SUBTC!MEMX!INCMAR!SP13>
1048  017730                         Z       STACK
  (1)          000603                MICPC=MICPC+1
  (1)  017730  115432                <JUMP!ZCOND!<STACK-INIT&3000*4>!<STACK-INIT&777/2>>
1049  017732                         CMP     <MEMX!INCMAR>,SP13      ;ACK TYPE
  (1)          000604                MICPC=MICPC+1
  (1)  017732  054373                <SUBTC!MEMX!INCMAR!SP13>
1050  017734                         Z       ACK
  (1)          000605                MICPC=MICPC+1
  (1)  017734  101746                <JUMP!ZCOND!<ACK-INIT&3000*4>!<ACK-INIT&777/2>>
1051  017736                         ALWAYS  IDLE                            ;OTHERWISE IGNORE--MUST BE OBS MSG
  (1)          000606                MICPC=MICPC+1
  (1)  017736  100451                <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
                                ;*****************TIME CRITICAL CODE-- CHANGE WITH GREAT CARE*********************
                                        .SBTTL  RCVK01--ROUTINE TO HANDLE FIRST BYTE ODD RECEIVE
1065                    RCVK01: SPBR    IBUS,NPR,SPO                    ;READ NPR REGISTER
1066                            MICPC=MICPC+1
1067  017740                    <MOVE!SPBRX!IBUS!NPR!SPO>
 (1)          000607
 (1)  017740  123600
1068  017742                    BRO     IDLE
 (1)          000610            MICPC=MICPC+1
 (1)  017742  102051            <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1069  017744                    BRWRTE  IMM,200                         ;MASK FOR CO(BYTE TRANSFER)
 (1)          000611            MICPC=MICPC+1
 (1)  017744  000600            <MOVE!WRTEBR!IMM!<200>>
1071  017746                    SP      BR,AORB,SPO
 (1)          000612            MICPC=MICPC+1
 (1)  017746  063300            <MOVE!SPX!BR!AORB!SPO>
1076  017750                    STATE   RKE1
 (1)          000613            MICPC=MICPC+1
 (1)  017750  000653            <MOVE!WRTEBR!IMM!<RKE1-INIT&777/2>>
1077  017752                    ALWAYS  RCVK02
 (1)          000614            MICPC=MICPC+1
 (1)  017752  104620            <JUMP!ALCOND!<RCVK02-INIT&3000*4>!<RCVK02-INIT&777/2>>
1078                                    .SBTTL  RCVK0--PROCESS ODD CHARACTER
1079  017754                    RCVK0:  SPBR    IBUS,NPR,SPO                    ;IS AN NPR GOING
 (1)          000615            MICPC=MICPC+1
 (1)  017754  123600            <MOVE!SPBRX!IBUS!NPR!SPO>
1081  017756                    BRO     RK66                            ;IF SO, REITERATE ODD AND EXIT
 (1)          000616            MICPC=MICPC+1
 (1)  017756  106247            <JUMP!BROCON!<RK66-INIT&3000*4>!<RK66-INIT&777/2>>
1086  017760                    STATE   RCVKE
 (1)          000617            MICPC=MICPC+1
 (1)  017760  000625            <MOVE!WRTEBR!IMM!<RCVKE-INIT&777/2>>
1087  017762                    RCVK02: SP      BR,SELB,SP3                     ;SET STATE
 (1)          000620            MICPC=MICPC+1
 (1)  017762  063223            <MOVE!SPX!BR!SELB!SP3>
1088  017764                    OUTPUT  IBUS,RCVDAT!OUTDA2              ;OUTPUT A CHAR
 (1)          000621            MICPC=MICPC+1
 (1)  017764  022203            <MOVE!WROUT!IBUS!<RCVDAT!OUTDA2>>
1089  017766                    RK8:    BRWRTE  IMM,21                  ;SET OUT NPR (C1) AND NPR REQ
 (1)          000622            MICPC=MICPC+1
 (1)  017766  000421            <MOVE!WRTEBR!IMM!<21>>
1093  017770                    RK7:    OUT     BR,<AORB!ONPR>                  ;WRITE NPR REGISTER
 (1)          000623            MICPC=MICPC+1
 (1)  017770  061310            <MOVE!WROUTX!BR!<AORB!ONPR>>
1094  017772                    ALWAYS  IDLE
 (1)          000624            MICPC=MICPC+1
 (1)  017772  100451            <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
1096
1097  017774                    SBTTL  RCVKE--HANDLE EVEN BYTES
                        RCVKE:  BRWRTE  IBUS,NPR                        ;READ NPR CONTROL REGISTER
 (1)          000625            MICPC=MICPC+1
 (1)  017774  120600            <MOVE!WRTEBR!IBUS!<NPR>>
1099  017776                    BR4     RK4                             ;IF  RECV NPR--BRANCH
 (1)          000626            MICPC=MICPC+1
 (1)  017776  107251            <JUMP!BR4CON!<RK4-INIT&3000*4>!<RK4-INIT&777/2>>
1104  020000              RK5:  SP      IBUS,IOBA1,SPO                    ;READ LOW BYTE OF BA TO SP
 (1)          000627            MICPC=MICPC+1
 (1)  020000  023140            <MOVE!SPX!IBUS!IOBA1!SPO>
1105  020002                    OUTPUT  DP,<INCA!OBA1>                  ;WRITE INCREMENTED BA
 (1)          000630            MICPC=MICPC+1
 (1)  020002  062066            <MOVE!WROUT!DP!<INCA!OBA1>>
1106  020004              RK50: SP      BR,DECA,SP4                       ;DECREMENT CHARACTER COUNT
 (1)          000631            MICPC=MICPC+1
 (1)  020004  063164            <MOVE!SPX!BR!DECA!SP4>
1107  020006                    C       10$                             ;NO OVERFLOW
 (1)          000632            MICPC=MICPC+1
 (1)  020006  105235            <JUMP!CCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1108  020010                    SP      BR,DECA,SP5                     ;OVERFLOW - DECREMENT HIGH BYTE
 (1)          000633            MICPC=MICPC+1
 (1)  020010  063165            <MOVE!SPX!BR!DECA!SP5>
1109  020012                    Z       RL3                             ;BYTE COUNT ZERO
 (1)          000634            MICPC=MICPC+1
 (1)  020012  105711            <JUMP!ZCOND!<RL3-INIT&3000*4>!<RL3-INIT&777/2>>
1110  020014              10$:  OUTPUT  IBUS,<RCVDAT!OUTDA1>            ;READ CHARACTER AND WRITE IT
 (1)          000635            MICPC=MICPC+1
 (1)  020014  022202            <MOVE!WROUT!IBUS!<RCVDAT!OUTDA1>>
1111  020016                    SP      IBUS,IOBA1,SPO                  ;READ INCREMENTED BA
 (1)          000636            MICPC=MICPC+1
 (1)  020016  023140            <MOVE!SPX!IBUS!IOBA1!SPO>
1112  020020                    OUTPUT  DP,<INCA!OBA1>                  ;WRITE INCREMENTED BA
 (1)          000637            MICPC=MICPC+1
 (1)  020020  062066            <MOVE!WROUT!DP!<INCA!OBA1>>
1113  020022                    C       ICBA22                          ;IF CARRY INC BA HIGH
 (1)          000640            MICPC=MICPC+1
 (1)  020022  115035            <JUMP!CCOND!<ICBA22-INIT&3000*4>!<ICBA22-INIT&777/2>>
1114  020024              RK3:  SP      BR,DECA,SP4                       ;DECREMENT THE COUNT OF BYTES
 (1)          000641            MICPC=MICPC+1
 (1)  020024  063164            <MOVE!SPX!BR!DECA!SP4>
1115  020026                    C       RK6                             ;NO OVERFLOW
 (1)          000642            MICPC=MICPC+1
 (1)  020026  105245            <JUMP!CCOND!<RK6-INIT&3000*4>!<RK6-INIT&777/2>>
1116  020030                    SP      BR,DECA,SP5                     ;DECREMENT HIGH BYTE OF COUNT
 (1)          000643            MICPC=MICPC+1
 (1)  020030  063165            <MOVE!SPX!BR!DECA!SP5>
1117  020032                    Z       RL4                             ;BYTE COUNT ZERO
 (1)          000644            MICPC=MICPC+1
 (1)  020032  111772            <JUMP!ZCOND!<RL4-INIT&3000*4>!<RL4-INIT&777/2>>
1119  020034              RK6:  BRWRTE  IBUS,RCVCON                       ;READ RECEIVER CONTROL REGISTER
 (1)          000645            MICPC=MICPC+1
 (1)  020034  020640            <MOVE!WRTEBR!IBUS!<RCVCON>>
1120  020036                    BR4     RCVKO                           ;IF ANOTHER CHARACTER--PROCESS
 (1)          000646            MICPC=MICPC+1
 (1)  020036  107215            <JUMP!BR4CON!<RCVKO-INIT&3000*4>!<RCVKO-INIT&777/2>>
1121  020040              RK66: STATE   RCVKO
```

```
  (1)           000647
  (1)  020040   000615                MICPC=MICPC+1
1122   020042                        <MOVE!WRTEBR!IMM!<RCVKO-INIT&777/2>>
                                     ALWAYS  REXIT
  (1)           000650                MICPC=MICPC+1
  (1)  020042   100450                <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
1123   020044                RK4:    BRO     IDLE
  (1)           000651                MICPC=MICPC+1
  (1)  020044   102051                <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1124   020046                        ALWAYS  RK5                     ;IF NO NPR --PROCESS
  (1)           000652                MICPC=MICPC+1
  (1)  020046   104627                <JUMP!ALCOND!<RK5-INIT&3000*4>!<RK5-INIT&777/2>>
1130                                 ;
1131   020050                RKE1:   SP      IBUS,NPR,SPO            ;READ NPR REGISTER
  (1)           000653                MICPC=MICPC+1
  (1)  020050   123200                <MOVE!SPX!IBUS!NPR!SPO>
1133   020052                        BRO     IDLE                    ;NPR STILL IN PROGRESS
  (1)           000654                MICPC=MICPC+1
  (1)  020052   102051                <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1135   020054                        BRWRTE  IMM,177                 ;MASK FOR ALL BUT CO
  (1)           000655                MICPC=MICPC+1
  (1)  020054   000577                <MOVE!WRTEBR!IMM!<177>>
1136   020056                        OUT     BR,<AANDB!ONPR>         ;TURN OFF ALL BUT CO
  (1)           000656                MICPC=MICPC+1
  (1)  020056   061270                <MOVE!WROUTX!BR!<AANDB!ONPR>>
1137   020060                        ALWAYS  RK50
  (1)           000657                MICPC=MICPC+1
  (1)  020060   104631                <JUMP!ALCOND!<RK50-INIT&3000*4>!<RK50-INIT&777/2>>
1138                                ;*****************END OF TIME CRITICAL PATH*************************************
1139
1140   020062                RCVKEO: SP      IBUS,RCVDAT,SPO         ;READ CHARACTER AND SAVE IN SPO
  (1)           000660                MICPC=MICPC+1
  (1)  020062   023200                <MOVE!SPX!IBUS!RCVDAT!SPO>
1141   020064                        OUTPUT  BR,<SELA!OUTDA1>        ;SEND NONSENSE CHARACTER
  (1)           000661                MICPC=MICPC+1
  (1)  020064   062202                <MOVE!WROUT!BR!<SELA!OUTDA1>>
1142   020066                        BRWRTE  BR,SELA!SP1             ;READ STATUS BYTE
  (1)           000662                MICPC=MICPC+1
  (1)  020066   060601                <MOVE!WRTEBR!BR!<SELA!SP1>>
1143   020070                        BR7     PASWRD                  ;MAINT MODE - SEE IF RLD MESSAGE
  (1)           000663                MICPC=MICPC+1
  (1)  020070   117576                <JUMP!BR7CON!<PASWRD-INIT&3000*4>!<PASWRD-INIT&777/2>>
1144   020072                        ALWAYS  RK3                     ;OTHERWISE PROCESS NORMALLY
  (1)           000664                MICPC=MICPC+1
  (1)  020072   104641                <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
```

```
1146
1147  020074                    RCVI:   SP      IBUS,RCVDAT,SP13        ;STORE UNNUMBERED TYPE
 (1)          000665                    MICPC=MICPC+1
 (1)  020074  023213                    <MOVE!SPX!IBUS!RCVDAT!SP13>
1148  020076                            STATE   RCVJ                    ;NEXT STATE IS J
 (1)          000666                    MICPC=MICPC+1
 (1)  020076  000670                    <MOVE!WRTEBR!IMM!<RCVJ-INIT&777/2>>
1149  020100                            ALWAYS  REXIT
 (1)          000667                    MICPC=MICPC+1
 (1)  020100  100450                    <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
1150                                    ;
```

```
 1152
 1153  020102
 1158  020102
  (1)              000670
  (1)  020102      023205
 1159  020104
  (1)              000671
  (1)  020104      000600
 1160  020106
  (1)              000672
  (1)  020106      060665
 1161  020110
  (1)              000673
  (1)  020110      063310
 1163  020112
  (1)              000674
  (1)  020112      000676
 1164  020114
  (1)              000675
  (1)  020114      100450
```

```
        .SBTTL  RCVJ--ROUTINE TO HANDLE SUBTYPE FIELD,SELECT AND FINAL
RCVJ:
        SP      IBUS,RCVDAT,SP5         ;GET CHARACTER
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!RCVDAT!SP5>
        BRWRTE  IMM,200                 ;CONDITIONALLY SET BIT
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        BRWRTE  BR,AANDB!SP5
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AANDB!SP5>>
        SP      BR,AORB,SP10
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        STATE   RCVR                    ;NEXT STATE IS N
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVR-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
```

```
1166
1167
1168
1169  020116                        .SBTTL  RCVR--UNNUMBERED MESSAGE RESPONSE FIELD
                                    ;ENTERED FROM IDLE LOOP
                                    ;
RCVR:   BRWRTE  IMM,3               ;REP MESSAGE TYPE TO BR
  (1)           000676              MICPC=MICPC+1
  (1)   020116  000403              <MOVE!WRTEBR!IMM!<3>>
1170  020120                        NOP     BR,SUB,SP13         ;IS TYPE ACK OR NAK
  (1)           000677              MICPC=MICPC+1
  (1)   020120  060353              <BR!SUB!SP13>
1171  020122                        STATE   RCVQ                ;NEXT STATE IS RCVQ
  (1)           000700              MICPC=MICPC+1
  (1)   020122  000703              <MOVE!WRTEBR!IMM!<RCVQ-INIT&777/2>>
1172                                                            ;***NOTE THIS INSTR DOES NOT CLOCK "C"
1173  020124                        C       RCVF1               ;IF NOT IGNORE
  (1)           000701              MICPC=MICPC+1
  (1)   020124  105131              <JUMP!CCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
1174  020126                        ALWAYS  RD2                 ;DO RANGE CHECKS
  (1)           000702              MICPC=MICPC+1
  (1)   020126  104473              <JUMP!ALCOND!<RD2-INIT&3000*4>!<RD2-INIT&777/2>>
```

```
 1176                                   .SBTTL  RCVQ--UNNUMBERED MESSAGE--NUMBER FIELD
 1177                                   ;ENTER  FROM IDLE
 1178                                   ;
 1179  020130                  RCVQ:    STATE   RCVF                    ;NEXT STATE IS ADDRESS
  (1)           000703                 MICPC=MICPC+1
  (1)  020130   000525                 <MOVE!WRTEBR!IMM!<RCVF-INIT&777/2>>
 1190  020132                          ALWAYS  RCVF1
  (1)           000704                 MICPC=MICPC+1
  (1)  020132   104531                 <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
```

```
1182
1183                                        .SBTTL  RCVL--PROCESS CRC3
                                            ;ENTERED FROM  IDLE LOOP
1184  020134                        RCVL:   SPBR    IBUS,NPR,SPO                    ;READ NPR CONTROL
  (1)           000705                      MICPC=MICPC+1
  (1)  020134   123600                      <MOVE!SPBRX!IBUS!NPR!SPO>
1186  020136                              BR4     RL1                             ;RCV NPR BRANCH
  (1)           000706                      MICPC=MICPC+1
  (1)  020136   107314                      <JUMP!BR4CON!<RL1-INIT&3000*4>!<RL1-INIT&777/2>>
1191  020140                        RL2:    BRWRTE  IMM,176                         ;MASK TO TURN OFF CO
  (1)           000707                      MICPC=MICPC+1
  (1)  020140   000576                      <MOVE!WRTEBR!IMM!<176>>
1192  020142                              OUT     BR,AANDB!ONPR
  (1)           000710                      MICPC=MICPC+1
  (1)  020142   061270                      <MOVE!WROUTX!BR!<AANDB!ONPR>>
1193                                        .
1194  020144                        RL3:    NOP     IBUS,RCVDAT,0           ;INPUT CHARACTER AND DISCARD
  (1)           000711                      MICPC=MICPC+1
  (1)  020144   020200                      <IBUS!RCVDAT!0>
1195  020146                              STATE   RCVM
  (1)           000712                      MICPC=MICPC+1
  (1)  020146   000716                      <MOVE!WRTEBR!IMM!<RCVM-INIT&777/2>>
1196  020150                              ALWAYS  REXIT
  (1)           000713                      MICPC=MICPC+1
  (1)  020150   100450                      <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
1197                                        .
1199  020152                        RL1:    BRO     IDLE                            ;NPR GOING --GET OUT
  (1)           000714                      MICPC=MICPC+1
  (1)  020152   102051                      <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1200  020154                              ALWAYS  RL2
  (1)           000715                      MICPC=MICPC+1
  (1)  020154   104707                      <JUMP!ALCOND!<RL2-INIT&3000*4>!<RL2-INIT&777/2>>
1202                                        ;
```

```
1204
1205
1206
1207
1208
1209  020156                                    .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
                                        RCVM:   BRWRTE  IBUS,UBBR                       ;READ UNIBUS BR REGISTER
  (1)           000716                          MICPC=MICPC+1
  (1)  020156   120620                          <MOVE!WRTEBR!IBUS!<UBBR>>
1210  020160                                    BRO     NXMERR                          ;NON-EXISTANT MEMORY
  (1)           000717                          MICPC=MICPC+1
  (1)  020160   106351                          <JUMP!BROCON!<NXMERR-INIT&3000*4>!<NXMERR-INIT&777/2>>
1211  020162                                    SP      IBUS,RCVDAT,SPO                 ;READ CRC CHARACTER
  (1)           000720                          MICPC=MICPC+1
  (1)  020162   023200                          <MOVE!SPX!IBUS!RCVDAT!SPO>
1212  020164                                    BRWRTE  IBUS,RCVCON                     ;READ RECEIVER CONTROL REGISTER
  (1)           000721                          MICPC=MICPC+1
  (1)  020164   020640                          <MOVE!WRTEBR!IBUS!<RCVCON>>
1213  020166                                    BRO     RCVM1                           ;IF CRC GOOD -- PROCESS
  (1)           000722                          MICPC=MICPC+1
  (1)  020166   116214                          <JUMP!BROCON!<RCVM1-INIT&3000*4>!<RCVM1-INIT&777/2>>
1214  020170                                    BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
  (1)           000723                          MICPC=MICPC+1
  (1)  020170   060601                          <MOVE!WRTEBR!BR!<SELA!SP1>>
1215  020172                                    BR7     RHX                             ;CRC ERROR IN BOOT MODE - FLUSH
  (1)           000724                          MICPC=MICPC+1
  (1)  020172   107740                          <JUMP!BR7CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
1216  020174                                    LDMA    IMM,T                           ;ELSE SEND NAK --DATA ERROR
  (1)           000725                          MICPC=MICPC+1
  (1)  020174   010151                          <MOVE!LDMAR!IMM!<T&377>>
1217  020176                                    MEMINC  IMM,2                           ;NAK TYPE
  (1)           000726                          MICPC=MICPC+1
  (1)  020176   016402                          <MOVE!WRMEM!INCMAR!IMM!<2>>
1218  020200                                    MEMINC  IMM,302                         ;DATA ERROR SUBTYPE
  (1)           000727                          MICPC=MICPC+1
  (1)  020200   016702                          <MOVE!WRMEM!INCMAR!IMM!<302>>
1219  020202                                    LDMA    IMM,NDATS
  (1)           000730                          MICPC=MICPC+1
  (1)  020202   010014                          <MOVE!LDMAR!IMM!<NDATS&377>>
1220  020204                                    ALWAYS  RH5                             ;SEND NAK
  (1)           000731                          MICPC=MICPC+1
  (1)  020204   104552                          <JUMP!ALCOND!<RH5-INIT&3000*4>!<RH5-INIT&777/2>>
1221
1222  020206                                RCVM0:  LDMA    IMM,<<RTHRS+3>>                 ;POINT TO ERROR WORD
  (1)           000732                          MICPC=MICPC+1
  (1)  020206   010177                          <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
1223  020210                                    BRWRTE  IMM,10                          ;MAINT MESSAGE ERROR
  (1)           000733                          MICPC=MICPC+1
  (1)  020210   000410                          <MOVE!WRTEBR!IMM!<10>>
1224  020212                                    ALWAYS RCEXY                            ;GIVE FATAL ERROR
  (1)           000734                          MICPC=MICPC+1
  (1)  020212   114522                          <JUMP!ALCOND!<RCEXY-INIT&3000*4>!<RCEXY-INIT&777/2>>
```

```
                                        .SBTTL  EM2--PROCESS RLD MESSAGE
                                        ;ENTERED FROM IDLE LOOP
                                        ;IF RLD PASSWORD CHECKS TRIGGER THE BOOT ROM

020214                          EM2:    BRWRTE  IBUS,RCVDAT             ;READ THE CHAR
        000735                          MICPC=MICPC+1
020214  020600                          <MOVE!WRTEBR!IBUS!<RCVDAT>>
020216                                  CMP     BR,SP13                 ;IS IT A MATCH
        000736                          MICPC=MICPC+1
020216  060373                          <SUBTC!BR!SP13>
020220                                  Z       EM3
        000737                          MICPC=MICPC+1
020220  105746                          <JUMP!ZCOND!<EM3-INIT&3000*4>!<EM3-INIT&777/2>>
                                                                        ;FALL INTO RHX
020222                          RHX:    BRWRTE  BR,AA!SP1               ;READ STATUS BYTE SHIFTED LEFT
        000740                          MICPC=MICPC+1
020222  060521                          <MOVE!WRTEBR!BR!<AA!SP1>>
020224                                  BR4     10$                     ;DLE RECEIVED IN NORMAL MODE
        000741                          MICPC=MICPC+1
020224  107343                          <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
020226                                  ALWAYS  FLUSH                   ;ALREADY IN MAINT MODE
        000742                          MICPC=MICPC+1
020226  104415                          <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
020230                          10$:    BRWRTE  IMM,163                 ;MASK TO CLEAR ALL MAINT RELATED BITS
        000743                          MICPC=MICPC+1
020230  000563                          <MOVE!WRTEBR!IMM!<163>>
020232                                  SP      BR,AANDB,SP1            ;CLEAR THEM
        000744                          MICPC=MICPC+1
020232  063261                          <MOVE!SPX!BR!AANDB!SP1>
020234                                  ALWAYS  FLUSH
        000745                          MICPC=MICPC+1
020234  104415                          <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>

020236                          EM3:    SP      BR,DECA,SP4             ;DECREMENT CHARACTER COUNT BY ONE
        000746                          MICPC=MICPC+1
020236  063164                          <MOVE!SPX!BR!DECA!SP4>
020240                                  Z       EMTRIG                  ;TRIGGER AC LOW
        000747                          MICPC=MICPC+1
020240  115712                          <JUMP!ZCOND!<EMTRIG-INIT&3000*4>!<EMTRIG-INIT&777/2>>
020242                                  ALWAYS  IDLE
        000750                          MICPC=MICPC+1
020242  100451                          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
1246
1247  020244
 (1)              000751
 (1)  020244  010177
1248  020246
 (1)              000752
 (1)  020246  016401
1249  020250
 (1)              000753
 (1)  020250  002400
1250  020252
 (1)              000754
 (1)  020252  043230
1251  020254
 (1)              000755
 (1)  020254  114524
```

```
                .SBTTL NXMERR ---NON EXISTANT MEMORY HANDLER
NXMERR: LDMA    IMM,<<RTHRS+3>>          ;ADDRESS ERROR LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
        MEMINC  IMM,1
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<1>>
        MEM     IMM,0                    ;NXM ERROR BIT
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<0>>
        SP      MEMX,SELB,SP10           ;CLEAR STATUS
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP10>
        ALWAYS  RCEXX
        MICPC=MICPC+1
        <JUMP!ALCOND!<RCEXX-INIT&3000*4>!<RCEXX-INIT&777/2>>
```

```
1273
1274  020256                        ;
 (1)          000756
 (1)  020256  060601  BOOT:  BRWRTE  BR,SELA!SP1          ;SEE IF IN MAINT. MODE
                             MICPC=MICPC+1
                             <MOVE!WRTEBR!BR!<SELA!SP1>>
1275  020260                        BR7     RA3            ;BRANCH IF SO AND TREAT DLE LIKE NUM. MSG.
 (1)          000757
 (1)  020260  103742                MICPC=MICPC+1
                             <JUMP!BR7CON!<RA3-INIT&3000*4>!<RA3-INIT&777/2>>
1276  020262                        BRWRTE  IMM,210          ;MASK TO SET MAINT MODE AND DLE RECV'D
 (1)          000760
 (1)  020262  000610                MICPC=MICPC+1
                             <MOVE!WRTEBR!IMM!<210>>
1277  020264                        SP      BR,AORB,SP1      ;SET THE BITS
 (1)          000761
 (1)  020264  063301                MICPC=MICPC+1
                             <MOVE!SPX!BR!AORB!SP1>
1278  020266                        ALWAYS  RA3              ;TREAT LIKE NUMBERED MESSAGE
 (1)          000762
 (1)  020266  100742                MICPC=MICPC+1
                             <JUMP!ALCOND!<RA3-INIT&3000*4>!<RA3-INIT&777/2>>
1279  020270  RESEXT:  BRWRTE  IMM,4                    ;ADD TO MXT BITS
 (1)          000763
 (1)  020270  000404                MICPC=MICPC+1
                             <MOVE!WRTEBR!IMM!<4>>
1280  020272                        SP      BR,ADD,SPO
 (1)          000764
 (1)  020272  063000                MICPC=MICPC+1
                             <MOVE!SPX!BR!ADD!SPO>
1281  020274                        ALWAYS  TH3X
 (1)          000765
 (1)  020274  110601                MICPC=MICPC+1
                             <JUMP!ALCOND!<TH3X-INIT&3000*4>!<TH3X-INIT&777/2>>
1282  020276  TABMXT:  BRWRTE  IMM,4                    ;INCREMENT MXT
 (1)          000766
 (1)  020276  000404                MICPC=MICPC+1
                             <MOVE!WRTEBR!IMM!<4>>
1283  020300                        SP      IBUS,UBBR,SPO            ;READ BR CONTROL
 (1)          000767
 (1)  020300  123220                MICPC=MICPC+1
                             <MOVE!SPX!IBUS!UBBR!SPO>
1284  020302                        OUT     BR,ADD!OBR
 (1)          000770
 (1)  020302  061011                MICPC=MICPC+1
                             <MOVE!WROUTX!BR!<ADD!OBR>>
1285  020304                        ALWAYS  ECX
 (1)          000771
 (1)  020304  114761                MICPC=MICPC+1
                             <JUMP!ALCOND!<ECX-INIT&3000*4>!<ECX-INIT&777/2>>
1286
1287  020306                ;
                            RTHRES:  BRWRTE  IMM,2
 (1)          000772
 (1)  020306  000402                MICPC=MICPC+1
                             <MOVE!WRTEBR!IMM!<2>>
1288  020310                        ALWAYS  ERRXX
 (1)          000773
 (1)  020310  114663                MICPC=MICPC+1
                             <JUMP!ALCOND!<ERRXX-INIT&3000*4>!<ERRXX-INIT&777/2>>
1291  020312                        $ZERO
 (2)          000774
 (2)  020312  000000                MICPC=MICPC+1
                             000000
 (1)  020314                        $ZERO
 (2)          000775
 (2)  020314  000000                MICPC=MICPC+1
                             000000
 (1)  020316                        $ZERO
 (2)          000776
 (2)  020316  000000                MICPC=MICPC+1
                             000000
 (1)  020320                        $ZERO
 (2)          000777
 (2)  020320  000000                MICPC=MICPC+1
                             000000
```

```
1293            020322
1294            000777
1295
1296
1297  020322                    TMTDA:  BRWRTE  IBUS,TMTCON                    ;READ TRANMITTER CONTROL REGISTER
 (1)            001000                  MICPC=MICPC+1
 (1)  020322    020620                  <MOVE!WRTEBR!IBUS!<TMTCON>>
1298  020324                            .BR4    DP,SELA,<2!PAGE2>               ;IF READY PROCEED
 (1)            001001                  MICPC=MICPC+1
 (1)  020324    173202                  <JUMP! BR4CON!DP!SELA!2!PAGE2>
1299  020326                            ALWAYS  I1                              ;ELSE IDLE
 (1)            001002                  MICPC=MICPC+1
 (1)  020326    100454                  <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
```

```
1301
1302
1303  020330
1309  020330
 (1)            001003
 (1)  020330    060610
1310  020332
 (1)            001004
 (1)  020332    112007
1311  020334
 (1)            001005
 (1)  020334    001620
1312  020336
 (1)            001006
 (1)  020336    103063
1318  020340
 (1)            001007
 (1)  020340    060610
1319  020342
 (1)            001010
 (1)  020342    113412
1320  020344
 (1)            001011
 (1)  020344    100454
1321  020346
 (1)            001012
 (1)  020346    020660
1322  020350
 (1)            001013
 (1)  020350    001620
1323  020352
 (1)            001014
 (1)  020352    103054
1324  020354
 (1)            001015
 (1)  020354    000773
1325  020356
 (1)            001016
 (1)  020356    063270
1326  020360
 (1)            001017
 (1)  020360    000424
 (1)            001020
 (1)  020362    063222
1327  020364
 (1)            001021
 (1)  020364    000412
1328  020366
 (1)            001022
 (1)  020366    063226
1329  020370
 (1)            001023
 (1)  020370    100454
1330  020372
 (1)            001024
 (1)  020372    063166
```

```
        .SBTTL  TMTA--FIRST CHARACTER OF HEADER
        ;
TMTA:
        BRWRTE  BR,SELA!SP10                    ;REREAD STATUS
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP10>>
        BRO     NUMSYN                          ;IF UNNUMBPENDING  -- SEND IT
        MICPC=MICPC+1
        <JUMP!BROCON!<NUMSYN-INIT&3000*4>!<NUMSYN-INIT&777/2>>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     IDLE0                           ;IF START MODE--EXIT
        MICPC=MICPC+1
        <JUMP!BR4CON!<IDLE0-INIT&3000*4>!<IDLE0-INIT&777/2>>
NUMSYN: BRWRTE  BR,<SELA!SP10>                  ;READ LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP10>>
        BR7     5$                              ;IF OK TO SEND--PROCEED
        MICPC=MICPC+1
        <JUMP!BR7CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        ALWAYS  I1                              ;ELSE--IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
5$:     BRWRTE  IBUS,MODEM                      ;ARE WE STILL SENDING?
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<MODEM>>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     I1                              ;RTS SET? IF SO WE ARE--STALL
        MICPC=MICPC+1
        <JUMP!BR4CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
        BRWRTE  IMM,373                         ;MASK TO TURN OFFLINE IDLE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<373>>
        SP      BR,AANDB,SP10                   ;IN LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP10>
        TSTATE  TMTA1
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTA1-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        BRWRTE  IMM,12
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<12>>
        SP      BR,SELB,SP6                     ;STORE IN SP6
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP6>
        ALWAYS  I1                              ;BACK TO IDLE LOOP
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
TMTA1:  SP      BR,DECA,SP6                     ;DECREMENT SYN COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP6>
```

```
 1331  020374                          Z       TMTEXT
  (1)          001025                 MICPC=MICPC+1
  (1)  020374  111432                 <JUMP!ZCOND!<TMTEXT-INIT&3000*4>!<TMTEXT-INIT&777/2>>
 1332  020376                          OUTPUT  IMM,<1!OTMTCO>                  ;WRITE SOM TO TMTR CONTRL
  (1)          001026                 MICPC=MICPC+1
  (1)  020376  002011                 <MOVE!WROUT!IMM!<1!OTMTCO>>
 1333  020400                          BRWRTE  IMM,226                         ;SYNC CHAR
  (1)          001027                 MICPC=MICPC+1
  (1)  020400  000626                 <MOVE!WRTEBR!IMM!<226>>
 1334  020402                          OUTPUT  BR,<SELB!TMTDAT>                ;SEND THE CHARACTER
  (1)          001030                 MICPC=MICPC+1
  (1)  020402  062230                 <MOVE!WROUT!BR!<SELB!TMTDAT>>
 1335  020404                          ALWAYS  I1
  (1)          001031                 MICPC=MICPC+1
  (1)  020404  100454                 <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 1337  020406                  TMTEXT: BRWRTE  BR,<SELA!SP10>                  ;UNNUMB MESSGE?
  (1)          001032                 MICPC=MICPC+1
  (1)  020406  060610                 <MOVE!WRTEBR!BR!<SELA!SP10>>
 1338  020410                          BRO     TMTUN                           ;IF SO --BRANCH
  (1)          001033                 MICPC=MICPC+1
  (1)  020410  112043                 <JUMP!BROCON!<TMTUN-INIT&3000*4>!<TMTUN-INIT&777/2>>
 1339  020412                          TSTATE  TMTB
  (1)          001034                 MICPC=MICPC+1
  (1)  020412  000451                 <MOVE!WRTEBR!IMM!<TMTB-INIT&777/2>>
  (1)          001035                 MICPC=MICPC+1
  (1)  020414  063222                 <MOVE!SPX!BR!SELB!SP2>
 1340  020416                          BRWRTE  BR,SELA!SP1                     ;ARE WE IN BOOT MODE
  (1)          001036                 MICPC=MICPC+1
  (1)  020416  060601                 <MOVE!WRTEBR!BR!<SELA!SP1>>
 1341  020420                          BR7     TMTBT                   ;IF SO SEND DLE
  (1)          001037                 MICPC=MICPC+1
  (1)  020420  113447                 <JUMP!BR7CON!<TMTBT-INIT&3000*4>!<TMTBT-INIT&777/2>>
 1342  020422                          BRWRTE  IMM,201                         ;ELSE STORE SOH
  (1)          001040                 MICPC=MICPC+1
  (1)  020422  000601                 <MOVE!WRTEBR!IMM!<201>>
 1343  020424                  TMTA5:  OUTPUT  BR,<SELB!TMTDAT>                ;IN TMT SILO
  (1)          001041                 MICPC=MICPC+1
  (1)  020424  062230                 <MOVE!WROUT!BR!<SELB!TMTDAT>>
 1344  020426                          ALWAYS  I1
  (1)          001042                 MICPC=MICPC+1
  (1)  020426  100454                 <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
 1345  020430                  TMTUN:  TSTATE  TMTI
  (1)          001043                 MICPC=MICPC+1
  (1)  020430  000610                 <MOVE!WRTEBR!IMM!<TMTI-INIT&777/2>>
  (1)          001044                 MICPC=MICPC+1
  (1)  020432  063222                 <MOVE!SPX!BR!SELB!SP2>
 1346  020434                          BRWRTE  IMM,5                           ;ENQ TO BR
  (1)          001045                 MICPC=MICPC+1
  (1)  020434  000405                 <MOVE!WRTEBR!IMM!<5>>
 1347  020436                          ALWAYS  TMTA5
  (1)          001046                 MICPC=MICPC+1
  (1)  020436  110441                 <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
 1348  020440                  TMTBT:  BRWRTE  IMM,220                 ;WRITE A DLE TO BR
  (1)          001047                 MICPC=MICPC+1
  (1)  020440  000620                 <MOVE!WRTEBR!IMM!<220>>
 1349  020442                          ALWAYS  TMTA5                   ;SEND IT
```

```
(1)             001050          MICPC=MICPC+1
(1)  020442  110441             <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
```

```
                        .SBTTL  TMTB--OUTPUT FIRST CHAR OF COUNT
                        ;
1373
1374
1375  020444            TMTB:   LDMA    BR,SELA!SP16                    ;GETPOINTER TO NEXT TMT LINK
 (1)                    MICPC=MICPC+1
 (1)  020444  070216    <MOVE!LDMAR!BR!<SELA!SP16>>
1376  020446            MEMINC  IMM,3                                   ;WRITE MSG TMTED TO FLAGS
 (1)                    MICPC=MICPC+1
 (1)  020446  016403    <MOVE!WRMEM!INCMAR!IMM!<3>>
1377  020450            MEMINC  BR,SELA!SP12                            ;PICK UP MSGNO
 (1)                    MICPC=MICPC+1
 (1)  020450  076612    <MOVE!WRMEM!INCMAR!BR!<SELA!SP12>>
1378  020452            STATE   TMTC                                    ;ADDRESS TMTR STATE
 (1)                    MICPC=MICPC+1
 (1)  020452  000476    <MOVE!WRTEBR!IMM!<TMTC-INIT&777/2>>
1379  020454            TBO:    SP      BR,SELB,SP2                     ;UPDATE IT
 (1)                    MICPC=MICPC+1
 (1)  020454  063222    <MOVE!SPX!BR!SELB!SP2>
1380  020456            OUTPUT  <MEMX!INCMAR>,SELB!IBA1                 ;WRITELOWBYTEOFADDRESS
 (1)                    MICPC=MICPC+1
 (1)  020456  056224    <MOVE!WROUT!MEMX!INCMAR!<SELB!IBA1>>
1381  020460            OUTPUT  <MEMX!INCMAR>,SELB!IBA2                 ;WRITE HIGH BYTE OF ADDRESS
 (1)                    MICPC=MICPC+1
 (1)  020460  056225    <MOVE!WROUT!MEMX!INCMAR!<SELB!IBA2>>
1382  020462            SP      MEMX,SELB,SP7   ;HIGH BYTE OF COUNT TO SP7
 (1)                    MICPC=MICPC+1
 (1)  020462  043227    <MOVE!SPX!MEMX!SELB!SP7>
                                                ;WAIT TO MASK OFF MEM EXT. BITS
1383
1384  020464            SP      IBUS,NPR,SPO
 (1)                    MICPC=MICPC+1
 (1)  020464  123200    <MOVE!SPX!IBUS!NPR!SPO>
1385  020466            BRWRTE  IMM,220
 (1)                    MICPC=MICPC+1
 (1)  020466  000620    <MOVE!WRTEBR!IMM!<220>>
1386  020470            SP      BR,AANDB,SPO
 (1)                    MICPC=MICPC+1
 (1)  020470  063260    <MOVE!SPX!BR!AANDB!SPO>
1387  020472            SP      IMM,300,SP6                             ;MASK FOR MXT
 (1)                    MICPC=MICPC+1
 (1)  020472  003306    <MOVE!SPX!IMM!300!SP6>
1388  020474            BRWRTE  MEMX!INCMAR,AANDB!SP6                   ;TURN OFF CC2
 (1)                    MICPC=MICPC+1
 (1)  020474  054666    <MOVE!WRTEBR!MEMX!INCMAR!<AANDB!SP6>>
1389  020476            OUTPUT  MEMX,SELB!TMTDAT                        ;ALSO WRITE COUNT TO TMTR SILO
 (1)                    MICPC=MICPC+1
 (1)  020476  042230    <MOVE!WROUT!MEMX!<SELB!TMTDAT>>
1390  020500            BRSHFT                                          ;SHIFT BITS INTO CORRECT POSITION
 (1)                    MICPC=MICPC+1
 (1)  020500  001620    <MOVE!SHFTBR!WRTEBR!SELB>
1391  020502            BRSHFT
 (1)                    MICPC=MICPC+1
 (1)  020502  001620    <MOVE!SHFTBR!WRTEBR!SELB>
1392  020504            BRSHFT
 (1)                    MICPC=MICPC+1
 (1)  020504  001620    <MOVE!SHFTBR!WRTEBR!SELB>
1393  020506            BRSHFT
 (1)                    MICPC=MICPC+1
```

```
   (1)  020506  001620          <MOVE!SHFTBR!WRTEBR!SELB>
1394    020510                  OUT     BR,AORB!ONPR
   (1)          001073          MICPC=MICPC+1
   (1)  020510  061310          <MOVE!WROUTX!BR!<AORB!ONPR>>
1395    020512                  SPBR    MEMX,SELB,SP6           ;LOWBYTE OF COUNT TO SP6
   (1)          001074          MICPC=MICPC+1
   (1)  020512  043626          <MOVE!SPBRX!MEMX!SELB!SP6>
1400    020514                  ALWAYS  I1
   (1)          001075          MICPC=MICPC+1
   (1)  020514  100454          <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1402                            ;
```

```
1404
1405
1406  020516
 (1)                001076
 (1)   020516       000477
1407  020520
 (1)                001077
 (1)   020520       063667
1408  020522
 (1)                001100
 (1)   020522       062230
1409  020524
 (1)                001101
 (1)   020524       000543
1410  020526
 (1)                001102
 (1)   020526       060376
1411  020530
 (1)                001103
 (1)   020530       111511
1412  020532
 (1)                001104
 (1)   020532       000406
1413  020534
 (1)                001105
 (1)   020534       063016
1414  020536
1420  020536
 (1)                001106
 (1)   020536       000514
 (1)                001107
 (1)   020540       063222
1421  020542
 (1)                001110
 (1)   020542       100454
1423  020544
 (1)                001111
 (1)   020544       000471
1424  020546
 (1)                001112
 (1)   020546       063236
1425  020550
 (1)                001113
 (1)   020550       110506
1426
```

```
        .SBTTL  TMTC--OUTPUT SECOND CHAR OF COUNT
        ;
TMTC:   BRWRTE  IMM,77                          ;MASK TO CLEAR MXT BITS
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<77>>
        SPBR    BR,AANDB,SP7                    ;CLEAR THEM
        MICPC=MICPC+1
        <MOVE!SPBRX!BR!AANDB!SP7>
        OUTPUT  DP,<SELB!TMTDAT>                ;WRITE TO TMT SILO
        MICPC=MICPC+1
        <MOVE!WROUT!DP!<SELB!TMTDAT>>
        BRWRTE  IMM,TML8                                ;GET WRAPAROUND ADDRESS
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TML8>>
        CMP     BR,SP16                         ;WRAPAORUND
        MICPC=MICPC+1
        <SUBTC!BR!SP16>
        Z       10$
        MICPC=MICPC+1
        <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
        BRWRTE  IMM,6                           ;OFFSET TO NEXT LINK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
        SP      BR,ADD,SP16                     ;UPDATE THE POINTER
        MICPC=MICPC+1
        <MOVE!SPX!BR!ADD!SP16>
5$:
        TSTATE  TMTD
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTD-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        ALWAYS  I1                      ;****OCTOBER 29, 1976
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
10$:    BRWRTE  IMM,TML1                                ;GO BACK TO FIRST LINK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TML1>>
        SP      BR,SELB,SP16
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP16>
        ALWAYS  5$
        MICPC=MICPC+1
        <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        ;
```

```
1428
1429  020552
 (1)              001114
 (1)  020552      000524
1430  020554
 (1)              001115
 (1)  020554      063166
1431  020556
 (1)              001116
 (1)  020556      111120
1432  020560
 (1)              001117
 (1)  020560      063167
1433  020562
 (1)              001120
 (1)  020562      010171
1434  020564
 (1)              001121
 (1)  020564      042230
1435  020566
 (1)              001122
 (1)  020566      063222
1436  020570
 (1)              001123
 (1)  020570      100454
```

```
        .SBTTL  TMTD--RESPONSE FIELD-NUMBERED MESSAGE
TMTD:   STATE   TMTE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTE-INIT&777/2>>
        SP      BR,DECA,SP6                     ;ADJUSRT COUNT FOR  TWO'S  COMPLEMENT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP6>
        C       TD2                             ;NO OVERFLOW
        MICPC=MICPC+1
        <JUMP!CCOND!<TD2-INIT&3000*4>!<TD2-INIT&777/2>>
        SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP7>
TD2:    LDMA    IMM,ISP11                       ;RESP FIELD ADDR TO MAR
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<ISP11&377>>
TD3:    OUTPUT  MEMX,SELB!TMTDAT                ;WRITE IT TO SILO
        MICPC=MICPC+1
        <MOVE!WROUT!MEMX!<SELB!TMTDAT>>
XEXIT2: SP      BR,SELB,SP2
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        ALWAYS  I1
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
```

```
 1438
 1439  020572
 1440  020572
  (1)              001124
  (1)  020572  123600
 1441  020574
  (1)              001125
  (1)  020574  102054
 1442  020576
  (1)              001126
  (1)  020576  060612
 1443  020600
  (1)              001127
  (1)  020600  062230
 1444  020602
  (1)              001130
  (1)  020602  000532
 1445  020604
  (1)              001131
  (1)  020604  110600
```

```
        .SBTTL  TMTE--NUMBER FIELD--NUMBERED MESSAGE
TMTE:
        SPBR    IBUS,NPR,SPO            ;READ NPR CONTROL REGISTER
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BRO     I1                      ;BUSY - GET OUT
        MICPC=MICPC+1
        <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
        BRWRTE  BR,SELA!SP12
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP12>>
        OUTPUT  BR,<SELB!TMTDAT>                ;WRITE IT TO THE SILO
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!TMTDAT>>
        STATE   TMTF
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTF-INIT&777/2>>
        ALWAYS  TH3
        MICPC=MICPC+1
        <JUMP!ALCOND!<TH3-INIT&3000*4>!<TH3-INIT&777/2>>
```

```
1447
1448
1449  020606
 (1)            001132
 (1)  020606    000537
1450  020610
 (1)            001133
 (1)  020610    063222
1451  020612
 (1)            001134
 (1)  020612    000401
1453  020614
 (1)            001135
 (1)  020614    062230
1454  020616
 (1)            001136
 (1)  020616    100454
```

```
                .SBTTL  TMTF--NUMBERED MSG ADDRESS FIELD
                ;
TMTF:           STATE   TF1
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<TF1-INIT&777/2>>
TF2:            SP      BR,SELB,SP2
                MICPC=MICPC+1
                <MOVE!SPX!BR!SELB!SP2>
                BRWRTE  IMM,1                           ;LOAD ADDRESS
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<1>>
TF3:            OUTPUT  BR,<SELB!TMTDAT>
                MICPC=MICPC+1
                <MOVE!WROUT!BR!<SELB!TMTDAT>>
                ALWAYS  I1
                MICPC=MICPC+1
                <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
```

```
1460
1461  020620
 (1)              001137
 (1)  020620      000402
1462  020622
 (1)              001140
 (1)  020622      062231
1463  020624
 (1)              001141
 (1)  020624      062230
1464  020626
 (1)              001142
 (1)  020626      020500
1465  020630
 (1)              001143
 (1)  020630      112162
1466  020632
 (1)              001144
 (1)  020632      000546
1467  020634
 (1)              001145
 (1)  020634      110563
```

```
        .SBTTL  TF1-NUMBERED MSG HEADER EOM
TF1:    BRWRTE  IMM,2                           ;EOM  MASK TO BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        OUTPUT  BR,<SELB!OTMTCO>                ;UPDATE TMTR CONTROL REGISTER
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!OTMTCO>>
        OUTPUT  BR,<SELB!TMTDAT>                ;OUTPUT A GARBAGE CHAR
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!TMTDAT>>
        BRWRTE  IBUS,IIBA1                      ;READ LOW ORDER FROM INBA
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<IIBA1>>
        BRO     TMTF1                           ;IF ODD BYTE--BRANCH
        MICPC=MICPC+1
        <JUMP!BROCON!<TMTF1-INIT&3000*4>!<TMTF1-INIT&777/2>>
        STATE   TMTH
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTH-INIT&777/2>>
        ALWAYS  XEXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
```

```
1469
1470
1471
1472  020636
 (1)                001146
 (1)   020636  123600
1474  020640
 (1)                001147
 (1)   020640  113151
1476  020642
 (1)                001150
 (1)   020642  102054
1477  020644
 (1)                001151
 (1)   020644  022010
1478  020646
 (1)                001152
 (1)   020646  023100
1479  020650
 (1)                001153
 (1)   020650  062064
1480  020652
 (1)                001154
 (1)   020652  063166
1481  020654
 (1)                001155
 (1)   020654  111160
1482  020656
 (1)                001156
 (1)   020656  063167
1483  020660
 (1)                001157
 (1)   020660  115407
1484  020662
1486  020662
 (1)                001160
 (1)   020662  020620
1487  020664
 (1)                001161
 (1)   020664  113165
1489  020666
 (1)                001162
 (1)   020666  000565
1494  020670
 (1)                001163
 (1)   020670  063222
1495  020672
 (1)                001164
 (1)   020672  100454
1497  020674
1502  020674
 (1)                001165
 (1)   020674  022030
1503  020676
 (1)                001166
 (1)   020676  023100
```

```
;*************TIME CRITICAL PATH--MODIFY WITH GREAT CARE
        .SBTTL  TMTH--ROUTINE TO OUTPUT DATA CHARACTERS
        ;
TMTH:   SPBR    IBUS,NPR,SPO                    ;READ NPR CONTROL
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BR4     5$                              ;IF RECV NPR --PROCESS
        MICPC=MICPC+1
        <JUMP!BR4CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        BRO     I1                              ;IF NPR IN PROGRESS --BRANCH
        MICPC=MICPC+1
        <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
5$:     OUTPUT  IBUS,<INDAT1!TMTDAT>            ;WRITE THE EVEN CHAR TO TMT SILO
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<INDAT1!TMTDAT>>
        SP      IBUS,IIBA1,SPO                  ;READ LOW BYTE OF BA TO SP
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IIBA1!SPO>
        OUTPUT  BR,<INCA!IBA1>                  ;OUTPUT INCREMENTED BA
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<INCA!IBA1>>
        SP      BR,DECA,SP6                     ;DECREMENT CHARACTER  COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP6>
        C       TH6                             ;NO OVERFLOW
        MICPC=MICPC+1
        <JUMP!CCOND!<TH6-INIT&3000*4>!<TH6-INIT&777/2>>
        SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP7>
        Z       HEH1                            ;BYTE  COUNT ZERO
        MICPC=MICPC+1
        <JUMP!ZCOND!<HEH1-INIT&3000*4>!<HEH1-INIT&777/2>>
TH6:
        BRWRTE  IBUS,TMTCON             ;READ TMTR CONTROL CSR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<TMTCON>>
        BR4     TH9                             ;IF MORE ROOM IN SILO--BRANCH
        MICPC=MICPC+1
        <JUMP!BR4CON!<TH9-INIT&3000*4>!<TH9-INIT&777/2>>
TMTF1:  STATE   TMTHO
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTHO-INIT&777/2>>
XEXIT:  SP      BR,SELB,SP2                     ;STORE NEW TRANSMIT STATE
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        ALWAYS  I1
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
TMTHO:
TH9:    OUTPUT  IBUS,<INDAT2!TMTDAT>            ;ODD CHAR TO SILO
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<INDAT2!TMTDAT>>
        SP      IBUS,IIBA1,SPO                  ;READ LOW BYTE TO BA
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IIBA1!SPO>
```

```
1504  020700
 (1)          001167
 (1)  020700  062064          OUTPUT  BR,<INCA!IBA1>                  ;OUTPUT THE INCREMENTED BA
                              MICPC=MICPC+1
                              <MOVE!WROUT!BR!<INCA!IBA1>>
1505  020702                  C       HOINCH
 (1)          001170          MICPC=MICPC+1
 (1)  020702  111377          <JUMP!CCOND!<HOINCH-INIT&3000*4>!<HOINCH-INIT&777/2>>
1506  020704          TH8:    SP      BR,DECA,SP6                     ;DECREMENT CHARACTERCOUNT
 (1)          001171          MICPC=MICPC+1
 (1)  020704  063166          <MOVE!SPX!BR!DECA!SP6>
1507  020706                  C       TH7                             ;NO OVERFLOW
 (1)          001172          MICPC=MICPC+1
 (1)  020706  111175          <JUMP!CCOND!<TH7-INIT&3000*4>!<TH7-INIT&777/2>>
1508  020710                  SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF  COUNT
 (1)          001173          MICPC=MICPC+1
 (1)  020710  063167          <MOVE!SPX!BR!DECA!SP7>
1509  020712                  Z       HEH1                            ;BYTE COUNT ZERO
 (1)          001174          MICPC=MICPC+1
 (1)  020712  115407          <JUMP!ZCOND!<HEH1-INIT&3000*4>!<HEH1-INIT&777/2>>
1510  020714          TH7:    SPBR    IBUS,NPR,SPO                    ;READ NPR REGISTER
 (1)          001175          MICPC=MICPC+1
 (1)  020714  123600          <MOVE!SPBRX!IBUS!NPR!SPO>
1512  020716                  BRO     TH2                             ;IF NPR BUSY WAIT TO GO
 (1)          001176          MICPC=MICPC+1
 (1)  020716  112205          <JUMP!BROCON!<TH2-INIT&3000*4>!<TH2-INIT&777/2>>
1514  020720                  STATE   TMTH
 (1)          001177          MICPC=MICPC+1
 (1)  020720  000546          <MOVE!WRTEBR!IMM!<TMTH-INIT&777/2>>
1515  020722          TH3:    SP      BR,SELB,SP2                     ;SAVE TSTATE
 (1)          001200          MICPC=MICPC+1
 (1)  020722  063222          <MOVE!SPX!BR!SELB!SP2>
1516  020724          TH3X:   BRWRTE  IMM,156                         ;CLEAR CO AND C1
 (1)          001201          MICPC=MICPC+1
 (1)  020724  000556          <MOVE!WRTEBR!IMM!<156>>
1517  020726                  SP      BR,AANDB,SPO                    ;CLEAR THE BITS
 (1)          001202          MICPC=MICPC+1
 (1)  020726  063260          <MOVE!SPX!BR!AANDB!SPO>
1518  020730                  OUT     BR,<INCA!ONPR>
 (1)          001203          MICPC=MICPC+1
 (1)  020730  061070          <MOVE!WROUTX!BR!<INCA!ONPR>>
1519  020732                  ALWAYS  I1
 (1)          001204          MICPC=MICPC+1
 (1)  020732  100454          <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1521  020734          TH2:    TSTATE  TH7
 (1)          001205          MICPC=MICPC+1
 (1)  020734  000575          <MOVE!WRTEBR!IMM!<TH7-INIT&777/2>>
 (1)          001206          MICPC=MICPC+1
 (1)  020736  063222          <MOVE!SPX!BR!SELB!SP2>
1522  020740                  ALWAYS  I1
 (1)          001207          MICPC=MICPC+1
 (1)  020740  100454          <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1524                    ;***************END TIME CRITICAL PATH*******************************
1525                            ;
```

```
1527
1528  020742
 (1)                001210
 (1)  020742  010151
1529  020744
 (1)                001211
 (1)  020744  043226
1530  020746
 (1)                001212
 (1)  020746  000614
1531  020750
 (1)                001213
 (1)  020750  110521
1532
1533
1534  020752
 (1)                001214
 (1)  020752  010152
1535  020754
 (1)                001215
 (1)  020754  000617
1536  020756
 (1)                001216
 (1)  020756  110521
```

```
                .SBTTL  TMTI--SEND UNNUMBERED TYPE FIELD
TMTI:   LDMA    IMM,T                           ;ADDRESS OF TYPE FIELD TO MAR
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        SP      MEMX,SELB,SP6                   ;COPY IT TO SP6
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP6>
        STATE   TMTJ
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTJ-INIT&777/2>>
        ALWAYS  TD3
        MICPC=MICPC+1
        <JUMP!ALCOND!<TD3-INIT&3000*4>!<TD3-INIT&777/2>>
        ;
        .SBTTL  TMTJ--SEND SUB-TYPE FIELD
TMTJ:   LDMA    IMM,ST                          ;ADDRESS OF SUB-TYPE FIELD TO MAR
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<ST&377>>
        STATE   TMTK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTK-INIT&777/2>>
        ALWAYS  TD3
        MICPC=MICPC+1
        <JUMP!ALCOND!<TD3-INIT&3000*4>!<TD3-INIT&777/2>>
```

```
1538                                        .SBTTL  TMTK--OUTPUT RESPONSE FIELD (UNNUMB MSG)
1539                                        ;
1540  020760                        TMTK:   BRWRTE  IMM,3                           ;WRITE A 3 TO BR
 (1)          001217                        MICPC=MICPC+1
 (1)  020760  000403                        <MOVE!WRTEBR!IMM!<3>>
1541  020762                                NOP     BR,SUB,SP6                      ;IF TYPE LESS THAN 3
 (1)          001220                        MICPC=MICPC+1
 (1)  020762  060346                        <BR!SUB!SP6>
1542  020764                                TSTATE  TMTL
 (1)          001221                        MICPC=MICPC+1
 (1)  020764  000625                        <MOVE!WRTEBR!IMM!<TMTL-INIT&777/2>>
 (1)          001222                        MICPC=MICPC+1
 (1)  020766  063222                        <MOVE!SPX!BR!SELB!SP2>
1543  020770                                C       TMTLO
 (1)          001223                        MICPC=MICPC+1
 (1)  020770  111232                        <JUMP!CCOND!<TMTLO-INIT&3000*4>!<TMTLO-INIT&777/2>>
1544  020772                                ALWAYS  TD2
 (1)          001224                        MICPC=MICPC+1
 (1)  020772  110520                        <JUMP!ALCOND!<TD2-INIT&3000*4>!<TD2-INIT&777/2>>
1545                                        ;
1546                                        .SBTTL  TMTL--UNNUMB MSG NUMBER FIELD
1547  020774                        TMTL:   TSTATE  TMTM
 (1)          001225                        MICPC=MICPC+1
 (1)  020774  000637                        <MOVE!WRTEBR!IMM!<TMTM-INIT&777/2>>
 (1)          001226                        MICPC=MICPC+1
 (1)  020776  063222                        <MOVE!SPX!BR!SELB!SP2>
1548  021000                                BRWRTE  IMM,3
 (1)          001227                        MICPC=MICPC+1
 (1)  021000  000403                        <MOVE!WRTEBR!IMM!<3>>
1549  021002                                CMP     BR,SP6                  ;IS MESSAGE REP
 (1)          001230                        MICPC=MICPC+1
 (1)  021002  060366                        <SUBTC!BR!SP6>
1550  021004                                Z       TMTL1                           ;YES
 (1)          001231                        MICPC=MICPC+1
 (1)  021004  111635                        <JUMP!ZCOND!<TMTL1-INIT&3000*4>!<TMTL1-INIT&777/2>>
1551  021006                        TMTLO:  BRWRTE  IMM,0                           ;ADDRESS CONTNAT OF ZERO
 (1)          001232                        MICPC=MICPC+1
 (1)  021006  000400                        <MOVE!WRTEBR!IMM!<0>>
1556  021010                                OUTPUT  BR,<SELB!TMTDAT>                ;SEND IT OUT
 (1)          001233                        MICPC=MICPC+1
 (1)  021010  062230                        <MOVE!WROUT!BR!<SELB!TMTDAT>>
1557  021012                                ALWAYS  I1                              ;BACK TO IDLE LOOP
 (1)          001234                        MICPC=MICPC+1
 (1)  021012  100454                        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1559                                        ;
1560  021014                        TMTL1:  BRWRTE  BR,DECA!SP12                    ;WRITE A RESPONSE
 (1)          001235                        MICPC=MICPC+1
 (1)  021014  060572                        <MOVE!WRTEBR!BR!<DECA!SP12>>
1561  021016                                ALWAYS  TMTA5
 (1)          001236                        MICPC=MICPC+1
 (1)  021016  110441                        <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
1562                                        ;
```

```
                        .SBTTL  TMTM--UNNUMB MSG--STATION ADDRESS
021020                  STATE   TNEOM
(1)     001237          MICPC=MICPC+1
(1)     021020  000641  <MOVE!WRTEBR!IMM!<TNEOM-INIT&777/2>>
021022                  ALWAYS  TF2
(1)     001240          MICPC=MICPC+1
(1)     021022  110533  <JUMP!ALCOND!<TF2-INIT&3000*4>!<TF2-INIT&777/2>>
021024                  BRWRTE  IMM,2                           ;END OF MESSAGE TO BR
(1)     001241          MICPC=MICPC+1
(1)     021024  000402  <MOVE!WRTEBR!IMM!<2>>
021026                  OUTPUT  BR,<SELB!OTMTCO>
(1)     001242          MICPC=MICPC+1
(1)     021026  062231  <MOVE!WROUT!BR!<SELB!OTMTCO>>
021030                  OUTPUT  BR,<SELB!TMTDAT>                ;OUTPUT A GARBAGE CHARACTER
(1)     001243          MICPC=MICPC+1
(1)     021030  062230  <MOVE!WROUT!BR!<SELB!TMTDAT>>
021032                  BRWRTE  IMM,4                           ;SET UP LINE HAS GONE IDLE MASK
(1)     001244          MICPC=MICPC+1
(1)     021032  000404  <MOVE!WRTEBR!IMM!<4>>
021034                  SPBR    BR,AORB,SP10                    ;UPDATE LINE STATUS WORD
(1)     001245          MICPC=MICPC+1
(1)     021034  063710  <MOVE!SPBRX!BR!AORB!SP10>
021036                  BRWRTE  BR,AA!SP10                      ;SHIFT STATUS LEFT
(1)     001246          MICPC=MICPC+1
(1)     021036  060530  <MOVE!WRTEBR!BR!<AA!SP10>>
021040                  BR7     10$                             ;IF HDX SET---BRANCH TO CLEAR OK TO SEND
(1)     001247          MICPC=MICPC+1
(1)     021040  113653  <JUMP!BR7CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
021042                  BRWRTE  IMM,376                         ;MASK TO TURN OFF UNNUMB PENDDING
(1)     001250          MICPC=MICPC+1
(1)     021042  000776  <MOVE!WRTEBR!IMM!<376>>
021044          5$:     SP      BR,AANDB,SP10                   ;MASK TO LINE STATUS WORD
(1)     001251          MICPC=MICPC+1
(1)     021044  063270  <MOVE!SPX!BR!AANDB!SP10>
021046                  ALWAYS  TEOM2
(1)     001252          MICPC=MICPC+1
(1)     021046  110740  <JUMP!ALCOND!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
021050          10$:    BRWRTE  IMM,176                         ;CLEAR OK TO SEND AND UNNUMB PENPENDING
(1)     001253          MICPC=MICPC+1
(1)     021050  000576  <MOVE!WRTEBR!IMM!<176>>
021052                  ALWAYS  5$
(1)     001254          MICPC=MICPC+1
(1)     021052  110651  <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
```

```
                        .SBTTL  TIMSRV--TIMEOUT ROUTINE--SENDS  REP
                        ;
                        .ENABLE LSB
021054          TIMSRV: BRWRTE  IMM,177                         ;MASK OFF  BR REQ
(1)                     MICPC=MICPC+1
(1)     021054  000577  <MOVE!WRTEBR!IMM!<177>>
                                                                ;RESET TIMER---SLICK MOVE
                                                                ;SINCE TIMER IS RESET BY WRITING
                                                                ;A 1 AND THE EXPIRATION   LOOKS
                                                                ;LIKE 1--VOILA
021056                  OUT     BR,<AANDB!OBR>                  ;AND THE BIT ON
(1)                     MICPC=MICPC+1
(1)     021056  061271  <MOVE!WROUTX!BR!<AANDB!OBR>>
021060                  BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
(1)                     MICPC=MICPC+1
(1)     021060  060601  <MOVE!WRTEBR!BR!<SELA!SP1>>
021062                  BRO     IDLE
(1)                     MICPC=MICPC+1
(1)     021062  102051  <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
021064                  BR7     IDLE                            ;IF IN MAINT. MODE DISABLE TIMER
(1)                     MICPC=MICPC+1
(1)     021064  103451  <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
021066                  SP      BR,DECA,SP15                    ;DECREMENT THE COUNTER
(1)                     MICPC=MICPC+1
(1)     021066  063175  <MOVE!SPX!BR!DECA!SP15>
021070                  Z       20$                             ;IF ALL ONES HAS EXPIRED
(1)                     MICPC=MICPC+1
(1)     021070  111670  <JUMP!ZCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
021072          10$:    BRWRTE  BR,SELA!SP10                    ;READ LINE STATUS
(1)                     MICPC=MICPC+1
(1)     021072  060610  <MOVE!WRTEBR!BR!<SELA!SP10>>
021074                  BR1     TABUPD                          ;NUMBERED MESSAGE IN PROGRESS
(1)                     MICPC=MICPC+1
(1)     021074  116731  <JUMP!BR1CON!<TABUPD-INIT&3000*4>!<TABUPD-INIT&777/2>>
021076                  BRO     TABUPD                          ;UNNUMBMSGIN PROGRESS
(1)                     MICPC=MICPC+1
(1)     021076  116331  <JUMP!BROCON!<TABUPD-INIT&3000*4>!<TABUPD-INIT&777/2>>
021100                  ALWAYS  IDLE                            ;ELSE BACK TO IDLE LOOP
(1)                     MICPC=MICPC+1
(1)     021100  100451  <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
021102          TIME1:
021102          20$:    BRWRTE  IMM,2                           ;
(1)                     MICPC=MICPC+1
(1)     021102  000402  <MOVE!WRTEBR!IMM!<2>>
021104                  SP      BR,SELB,SP15                    ;RESET THE TIMER TICK COUNT
(1)                     MICPC=MICPC+1
(1)     021104  063235  <MOVE!SPX!BR!SELB!SP15>
021106                  BRWRTE  IMM,201                         ;SET OK TO SEND AND
(1)                     MICPC=MICPC+1
(1)     021106  000601  <MOVE!WRTEBR!IMM!<201>>
021110                  SPBR    BR,AORB,SP10                    ;UNNUM MSG PENDING
(1)                     MICPC=MICPC+1
(1)     021110  063710  <MOVE!SPBRX!BR!AORB!SP10>
021112                  BRSHFT
(1)                     MICPC=MICPC+1
(1)     021112  001620  <MOVE!SHFTBR!WRTEBR!SELB>
```

```
1609  021114
 (1)              001275
 (1)  021114  103111          BR4     BS1                             ;IF IN START MODE--BRANCH
                              MICPC=MICPC+1
                              <JUMP!BR4CON!<BS1-INIT&3000*4>!<BS1-INIT&777/2>>
1610  021116                  BRWRTE  BR,DECA!SP12                    ;GET LAST NUMBER SENT
 (1)              001276      MICPC=MICPC+1
 (1)  021116  060572          <MOVE!WRTEBR!BR!<DECA!SP12>>
1611  021120                  CMP     BR,SP17                         ;COMPARE TO LAST ACKED
 (1)              001277      MICPC=MICPC+1
 (1)  021120  060377          <SUBTC!BR!SP17>
1612  021122                  Z       SNDACK                  ;IF EQ --SEND ACK
 (1)              001300      MICPC=MICPC+1
 (1)  021122  111733          <JUMP!ZCOND!<SNDACK-INIT&3000*4>!<SNDACK-INIT&777/2>>
1613  021124          TIME2:  LDMA    IMM,T                           ;LOAD ADDRESS OF TYPE FIELD IN UNNUMB SK
 (1)              001301      MICPC=MICPC+1
 (1)  021124  010151          <MOVE!LDMAR!IMM!<T&377>>
1614  021126                  MEMINC  IMM,3                           ;LOAD REP TYPE
 (1)              001302      MICPC=MICPC+1
 (1)  021126  016403          <MOVE!WRMEM!INCMAR!IMM!<3>>
1615  021130                  MEMINC  IMM,300                         ;ZERO THE SUB-TYPE
 (1)              001303      MICPC=MICPC+1
 (1)  021130  016700          <MOVE!WRMEM!INCMAR!IMM!<300>>
1616  021132                  LDMA    IMM,REPCS                       ;CUMULATIVE REPS RECD
 (1)              001304      MICPC=MICPC+1
 (1)  021132  010015          <MOVE!LDMAR!IMM!<REPCS&377>>
1617  021134                  SP      MEMX,SELB,SPO                   ;COPY IT TO SPO
 (1)              001305      MICPC=MICPC+1
 (1)  021134  043220          <MOVE!SPX!MEMX!SELB!SPO>
1618  021136                  MEM     BR,INCA!SPO                     ;INCREMENT IT
 (1)              001306      MICPC=MICPC+1
 (1)  021136  062460          <MOVE!WRMEM!BR!<INCA!SPO>>
1619  021140                  LDMA    IMM,REPST                       ;ADDRESS DYNAMIC REP COUNTER
 (1)              001307      MICPC=MICPC+1
 (1)  021140  010003          <MOVE!LDMAR!IMM!<REPST&377>>
1620  021142                  BRWRTE  MEMX,SELB                       ;COPY IT TO THE BR
 (1)              001310      MICPC=MICPC+1
 (1)  021142  040620          <MOVE!WRTEBR!MEMX!<SELB>>
1621  021144                  BSHFTB
 (1)              001311      MICPC=MICPC+1
 (1)  021144  061620          <MOVE!SHFTBR!SELB!BR>
1622  021146                  MEM     BR,SELB
 (1)              001312      MICPC=MICPC+1
 (1)  021146  062620          <MOVE!WRMEM!BR!<SELB>>
1623  021150                  BRO     RTHRES
 (1)              001313      MICPC=MICPC+1
 (1)  021150  106372          <JUMP!BROCON!<RTHRES-INIT&3000*4>!<RTHRES-INIT&777/2>>
1628  021152                  ALWAYS  IDLE
 (1)              001314      MICPC=MICPC+1
 (1)  021152  100451          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1629                          .DSABLE LSB
1630                          ;
```

```
1632  021154
 (1)                001315
 (1)  021154  120620
1633  021156
 (1)                001316
 (1)  021156  106351
1634  021160
 (1)                001317
 (1)  021160  000402
1635  021162
 (1)                001320
 (1)  021162  062231
1636  021164
 (1)                001321
 (1)  021164  062230
1637  021166
 (1)                001322
 (1)  021166  060601
1638  021170
 (1)                001323
 (1)  021170  113762
1639  021172
 (1)                001324
 (1)  021172  063072
1640  021174
 (1)                001325
 (1)  021174  070216
1641  021176
 (1)                001326
 (1)  021176  040620
1642  021200
 (1)                001327
 (1)  021200  112340
1643  021202
 (1)                001330
 (1)  021202  000775
1644  021204
 (1)                001331
 (1)  021204  063670
1645  021206
 (1)                001332
 (1)  021206  112340
1646
1647  021210
 (1)                001333
 (1)  021210  010151
1648  021212
 (1)                001334
 (1)  021212  016401
1649  021214
 (1)                001335
 (1)  021214  000405
1650  021216
 (1)                001336
 (1)  021216  016700
1651  021220

TEOM:   BRWRTE  IBUS,UBBR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<UBBR>>
        BRO     NXMERR                          ;NON-EXISTANT MEMORY
        MICPC=MICPC+1
        <JUMP!BROCON!<NXMERR-INIT&3000*4>!<NXMERR-INIT&777/2>>
        BRWRTE  IMM,2                           ;EOM TO BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        OUTPUT  BR,<SELB!OTMTCO>                ;WRITE TMTR CONTROL
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!OTMTCO>>
        OUTPUT  BR,<SELB!TMTDAT>                ;WRITE GARBAGE DATA
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!TMTDAT>>
        BRWRTE  BR,SELA!SP1                     ;CKECK FOR BOOT MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR7     BTEOM                           ;---IF SET IS MAINT MSG
        MICPC=MICPC+1
        <JUMP!BR7CON!<BTEOM-INIT&3000*4>!<BTEOM-INIT&777/2>>
        SP      BR,INCA,SP12                    ;INCREMENT THE MESSAGE NUMBER
        MICPC=MICPC+1
        <MOVE!SPX!BR!INCA!SP12>
TEOM1:  LDMA    BR,SELA!SP16                            ;ADDRESS LAST TMT LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!BR!<SELA!SP16>>
        BRWRTE  MEMX,SELB
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!<SELB>>
        BRO     TEOM2
        MICPC=MICPC+1
        <JUMP!BROCON!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
TEOM3:  BRWRTE  IMM,375                         ;TURN OFF MESSAGE PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<375>>
        SPBR    BR,AANDB,SP10                   ;
        MICPC=MICPC+1
        <MOVE!SPBRX!BR!AANDB!SP10>
        BRO     TEOM2                           ;IF UNNUMB PENDING--GO AWAY
        MICPC=MICPC+1
        <JUMP!BROCON!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
        .SBTTL  SNDACK--ROUTINE TO SEND AN ACK
SNDACK: LDMA    IMM,T
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,1
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<1>>
        BRWRTE  IMM,5
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<5>>
SA2:    MEMINC  IMM,300
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<300>>
SA3:    SP      BR,AORB,SP10
```

```
  (1)           001337                  MICPC=MICPC+1
  (1)  021220   063310                  <MOVE!SPX!BR!AORB!SP10>
1652                                    ;
1658  021222                    TEOM2:  †STATE  TMTA
  (1)           001340                  MICPC=MICPC+1
  (1)  021222   000403                  <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
  (1)           001341                  MICPC=MICPC+1
  (1)  021224   063222                  <MOVE!SPX!BR!SELB!SP2>
1659  021226                            ALWAYS  I1
  (1)           001342                  MICPC=MICPC+1
  (1)  021226   100454                  <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1661  021230                    FUDGE:  BRWRTE  IBUS,NPR                ;READ NPR CONTROL
  (1)           001343                  MICPC=MICPC+1
  (1)  021230   120600                  <MOVE!WRTEBR!IBUS!<NPR>>
1662  021232                            BRO     IDLE                    ;IF NPR GOING---LEAVE
  (1)           001344                  MICPC=MICPC+1
  (1)  021232   102051                  <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1663  021234                            BRWRTE  BR!LDMAR,SELA!SP4       ;LOAD THE MAR
  (1)           001345                  MICPC=MICPC+1
  (1)  021234   070604                  <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
1664  021236                            BR7     BS2                     ;IF SET - READ BACK ALL 200
  (1)           001346                  MICPC=MICPC+1
  (1)  021236   103520                  <JUMP!BR7CON!<BS2-INIT&3000*4>!<BS2-INIT&777/2>>
1665  021240                            MEMINC  IBUS,INDAT1             ;OTHERWISE RESTORE TWO BYTES
  (1)           001347                  MICPC=MICPC+1
  (1)  021240   036400                  <MOVE!WRMEM!INCMAR!IBUS!<INDAT1>>
1666  021242                            MEMINC  IBUS,INDAT2             ;..
  (1)           001350                  MICPC=MICPC+1
  (1)  021242   036420                  <MOVE!WRMEM!INCMAR!IBUS!<INDAT2>>
1667  021244                            BRWRTE  IMM,2                   ;UPDATE---UNIBUS ADDRESS
  (1)           001351                  MICPC=MICPC+1
  (1)  021244   000402                  <MOVE!WRTEBR!IMM!<2>>
1668  021246                            SP      BR,ADD,SP4              ;UPDATE NPR COUNTER
  (1)           001352                  MICPC=MICPC+1
  (1)  021246   063004                  <MOVE!SPX!BR!ADD!SP4>
1669  021250                            SP      IBUS,IIBA1,SP0          ;UPDATE ADDRESS LOW
  (1)           001353                  MICPC=MICPC+1
  (1)  021250   023100                  <MOVE!SPX!IBUS!IIBA1!SP0>
1670  021252                            OUTPUT  BR,ADD!IBA1
  (1)           001354                  MICPC=MICPC+1
  (1)  021252   062004                  <MOVE!WROUT!BR!<ADD!IBA1>>
1671  021254                            SP      IBUS,IIBA2,SP0          ;READ HIGH ADDRESS
  (1)           001355                  MICPC=MICPC+1
  (1)  021254   023120                  <MOVE!SPX!IBUS!IIBA2!SP0>
1672  021256                            OUTPUT  BR,AC!IBA2              ;UPDATE HIGH
  (1)           001356                  MICPC=MICPC+1
  (1)  021256   062105                  <MOVE!WROUT!BR!<AC!IBA2>>
1673  021260                            SP      IBUS,NPR,SP0            ;READ NPR REGISTER
  (1)           001357                  MICPC=MICPC+1
  (1)  021260   123200                  <MOVE!SPX!IBUS!NPR!SP0>
1674  021262                            C       RESEXT                  ;IF CARRY---UPDATE MXT
  (1)           001360                  MICPC=MICPC+1
  (1)  021262   105363                  <JUMP!CCOND!<RESEXT-INIT&3000*4>!<RESEXT-INIT&777/2>>
1675  021264                            ALWAYS  TH3X                    ;GO DO ANOTHER NPR
  (1)           001361                  MICPC=MICPC+1
  (1)  021264   110601                  <JUMP!ALCOND!<TH3X-INIT&3000*4>!<TH3X-INIT&777/2>>
```

```
1676  021266
 (1)              001362
 (1)  021266      000774
1677  021270
 (1)              001363
 (1)  021270      063270
1678  021272
 (1)              001364
 (1)  021272      063233
1679
1680  021274
 (1)              001365
 (1)  021274      010067
1681  021276
 (1)              001366
 (1)  021276      043220
1686  021300
 (1)              001367
 (1)  021300      000403
 (1)              001370
 (1)  021302      063222
1688  021304
 (1)              001371
 (1)  021304      114532
1689  021306
 (1)              001372
 (1)  021306      000705
 (1)              001373
 (1)  021310      063223
1690  021312
 (1)              001374
 (1)  021312      123200
1691  021314
 (1)              001375
 (1)  021314      000621
1692  021316
 (1)              001376
 (1)  021316      104623
1693  021320
 (1)              001377
 (1)  021320      023120
1694  021322
 (1)              001400
 (1)  021322      062065
1695  021324
 (1)              001401
 (1)  021324      115003
1696  021326
 (1)              001402
 (1)  021326      110571
1697
1698  021330
 (1)              001403
 (1)  021330      123200
1699  021332
 (1)              001404
```

```
BTEOM:  BRWRTE  IMM,374                 ;MASK FOR CLEAR MSG PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<374>>
        SP      BR,AANDB,SP10           ;TURN THEM OFF IN LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP10>
        SP      BR,SELB,SP13            ;STORE UNRECOGNIZABLE VALUE INTO SP13
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP13>
                                        ;SO "RH3" WILL EXIT BACK TO IDLE LOOP
        LDMA    IMM,STC                 ;ADDRESS START OF TMT  CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        SP      MEMX,SELB,SP0           ;COPY LINK ADDRESS
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP0>
        TSTATE  TMTA                    ;CHANGE XMIT STATE TO LINE IS IDLE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        ALWAYS  TDON2                   ;POST A DONE
        MICPC=MICPC+1
        <JUMP!ALCOND!<TDON2-INIT&3000*4>!<TDON2-INIT&777/2>>
RL4:    RSTATE  RCVL
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVL-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP3>
        SP      IBUS,NPR,SP0            ;READ NPR CONTROL REGISTER
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!NPR!SP0>
        BRWRTE  IMM,221
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<221>>
        ALWAYS  RK7
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK7-INIT&3000*4>!<RK7-INIT&777/2>>
HOINCH: SP      IBUS,IIBA2,SP0
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IIBA2!SP0>
        OUTPUT  BR,INCA!IBA2                    ;OUTPUT INCREMENTED BA
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<INCA!IBA2>>
        C       5$                              ;INCREMENT BYTEW COUNT
        MICPC=MICPC+1
        <JUMP!CCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        ALWAYS  TH8
        MICPC=MICPC+1
        <JUMP!ALCOND!<TH8-INIT&3000*4>!<TH8-INIT&777/2>>
        ;INCREMENT MXT BITS
5$:     SP      IBUS,NPR,SP0            ;READ NPR REG IWTH CURRENT MXT BITS
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!NPR!SP0>
        BRWRTE  IMM,4                   ;WRITE BIT TO ADD
        MICPC=MICPC+1
```

```
  (1)  021332  000404              <MOVE!WRTEBR!IMM!<4>>
1700   021334                      OUT     BR,<ADD!ONPR>           ;TURN ON PROPER MXT BITS
  (1)          001405              MICPC=MICPC+1
  (1)  021334  061010              <MOVE!WROUTX!BR!<ADD!ONPR>>
1701   021336                      ALWAYS  TH8
  (1)          001406              MICPC=MICPC+1
  (1)  021336  110571              <JUMP!ALCOND!<TH8-INIT&3000*4>!<TH8-INIT&777/2>>
1702                               ;
1704   021340              HEH1:   STATE   TEOM
  (1)          001407              MICPC=MICPC+1
  (1)  021340  000715              <MOVE!WRTEBR!IMM!<TEOM-INIT&777/2>>
1705   021342                      ALWAYS  XEXIT
  (1)          001410              MICPC=MICPC+1
  (1)  021342  110563              <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
1707                               ;
```

```
1713
1714  021344
 (1)            001411
 (1)  021344    010016
1715  021346
 (1)            001412
 (1)  021346    043220
1716  021350
 (1)            001413
 (1)  021350    062460
1717  021352
 (1)            001414
 (1)  021352    010151
1718  021354
 (1)            001415
 (1)  021354    016402
1719  021356
 (1)            001416
 (1)  021356    016703
1720  021360
 (1)            001417
 (1)  021360    114704
1721
1722
1723  021362
 (1)            001420
 (1)  021362    060610
1724  021364
 (1)            001421
 (1)  021364    001620
1725  021366
 (1)            001422
 (1)  021366    117026
1726
1727  021370
 (1)            001423
 (1)  021370    010177
1728  021372
 (1)            001424
 (1)  021372    000600
1729  021374
 (1)            001425
 (1)  021374    114522
1730  021376
 (1)            001426
 (1)  021376    010151
1731  021400
 (1)            001427
 (1)  021400    016407
1732  021402
 (1)            001430
 (1)  021402    000411
1733  021404
 (1)            001431
 (1)  021404    110736
1734
```

```
                .SBTTL  REP HANDLER
REP:    LDMA    IMM,REPCR               ;LOAD MAR ADDRESS WITH POINTER TO REPS RECD
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<REPCR&377>>
        SP      MEMX,SELB,SPD           ;READ NUMBER OF REPS RECD
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SPD>
        MEM     DP,<INCA!SPD>           ;INCREMNT REPS RECD
        MICPC=MICPC+1
        <MOVE!WRMEM!DP!<INCA!SPD>>
        LDMA    IMM,T                   ;LOAD ADDRESS OF TYPE FIELD
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,2                   ; LOAD NAK TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<2>>
        MEMINC  IMM,303                 ;LOAD REP RESPONSE SUB-TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<303>>
        ALWAYS  SNAK                    ;SEND AN UNNUMB MSG
        MICPC=MICPC+1
        <JUMP!ALCOND!<SNAK-INIT&3000*4>!<SNAK-INIT&777/2>>
        ;
                .SBTTL  START HANDLER
START:  BRWRTE  DP,<SELA!SP10>          ;READ LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!WRTEBR!DP!<SELA!SP10>>
        BRSHFT                          ;GET START MODE BIT IN TESTABLE POSITION
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     10$                     ;IF IN START MODE SET STACK
        MICPC=MICPC+1
        <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
                                        ;ELSE SET UP START ERROR
        LDMA    IMM,<<RTHRS+3>>
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
        BRWRTE  IMM,200
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        ALWAYS  RCEXY
        MICPC=MICPC+1
        <JUMP!ALCOND!<RCEXY-INIT&3000*4>!<RCEXY-INIT&777/2>>
10$:    LDMA    IMM,T                   ;SET UP ADDRESS OF TYPE FIELD
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,7                   ;WRITE STACK TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<7>>
        BRWRTE  IMM,11                  ;SET START RECD AND UNNUMB PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<11>>
        ALWAYS  SA2                     ;SEND THE UNNUMBERED MESSAGE
        MICPC=MICPC+1
        <JUMP!ALCOND!<SA2-INIT&3000*4>!<SA2-INIT&777/2>>
        ;
```

```
1735
1736  021406
 (1)            001432
 (1)  021406    000727
1737  021410
 (1)            001433
 (1)  021410    063270
1738  021412
 (1)            001434
 (1)  021412    110670

                SBTTL  STACK HANDLER
STACK:  BRWRTE  IMM,327                 ;MASK TO CLEAR START MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<327>>
        SP      BR,AANDB,SP10           ;CLEAR START MODE
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP10>
        ALWAYS  TIME1                   ;RESET TIMER AND IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<TIME1-INIT&3000*4>!<TIME1-INIT&777/2>>
```

```
1740  021414
 (1)            001435
 (1)  021414    023160
1741  021416
 (1)            001436
 (1)  021416    062067
1742  021420
 (1)            001437
 (1)  021420    115041
1743  021422
 (1)            001440
 (1)  021422    104641
1744
1745  021424
 (1)            001441
 (1)  021424    123220
1746  021426
 (1)            001442
 (1)  021426    000404
1747  021430
 (1)            001443
 (1)  021430    061011
1748  021432
 (1)            001444
 (1)  021432    104641
1753  021434
 (1)            001445
 (1)  021434    010011
1754  021436
 (1)            001446
 (1)  021436    043220
1755  021440
 (1)            001447
 (1)  021440    042460
1756  021442
 (1)            001450
 (1)  021442    010067
1757  021444
 (1)            001451
 (1)  021444    043236
1758  021446
 (1)            001452
 (1)  021446    060477
1759  021450
 (1)            001453
 (1)  021450    063232
1760  021452
 (1)            001454
 (1)  021452    000406
1761
1762  021454
 (1)            001455
 (1)  021454    063310
1763  021456
 (1)            001456
 (1)  021456    063235
```

```
ICBA22: SP      IBUS,IOBA2,SPO                  ;READTHEHIGH ORDERBITS OF BA TO SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA2!SPO>
        OUTPUT  DP,<INCA!OBA2>                  ;OUTPUT THE INCREMENTED COUNT
        MICPC=MICPC+1
        <MOVE!WROUT!DP!<INCA!OBA2>>
        C       5$                              ;IF CARRY SET INCREMENT THE MXTBITS
        MICPC=MICPC+1
        <JUMP!CCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        ALWAYS  RK3
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
        ;
5$:     SP      IBUS,UBBR,SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!UBBR!SPO>
        BRWRTE  IMM,4
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<4>>
        OUT     BR,<ADD!OBR>
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<ADD!OBR>>
        ALWAYS  RK3
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
NAK:    LDMA    IMM,NDATR                       ;CUMMULATIVE NAK COUNTER
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NDATR&377>>
        SP      MEMX,SELB,SPO                   ;READ IT
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SPO>
        MEM     MEMX,INCA!SPO                   ;INCREMENT THE COUNTER
        MICPC=MICPC+1
        <MOVE!WRMEM!MEMX!<INCA!SPO>>
        LDMA    IMM,STC                         ;ADDRESS START OF TMT CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        SP      MEMX,SELB,SP16                  ;COPY START OF CHAIN TO LAST XMIT POINTER
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP16>
        BRWRTE  BR,INCA!SP17                            ;GETLASTMESSAGE ACKED
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<INCA!SP17>>
        SP      BR,SELB,SP12                            ;COPY TO CURRENT NUMBER
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP12>
        BRWRTE  IMM,6                           ;WRITE NUMBERED MSG PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
                                                ;  AND LINE HAS GONE IDLE
        SP      BR,AORB,SP10                    ;SET IT IN LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        SP      BR,SELB,SP15                    ;RESET TIMER COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP15>
```

```
1764  021460
 (1)            001457
 (1)  021460    110725
1765  021462
 (1)            001460
 (1)  021462    000415
1766  021464
 (1)            001461
 (1)  021464    123220
1767  021466
 (1)            001462
 (1)  021466    063260
1768  021470
 (1)            001463
 (1)  021470    000600
1769  021472
 (1)            001464
 (1)  021472    061311
1770  021474
 (1)            001465
 (1)  021474    123000
1771  021476
 (1)            001466
 (1)  021476    100554
1772
```

```
        ALWAYS  IEOM1
        MICPC=MICPC+1
        <JUMP!ALCOND!<IEOM1-INIT&3000*4>!<IEOM1-INIT&777/2>>
ININT:  BRWRTE  IMM,15                  ;MASK FOR TURN OFF ALL BUT EXT MEM BITS + NXM
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<15>>
        SP      IBUS,UBBR,SPO           ;READ BR  CONTROL REGISTER
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!UBBR!SPO>
        SP      BR,AANDB,SPO            ;MASK OFF VECTOR TO X04
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SPO>
        BRWRTE  IMM,200                 ;MASK FOR INTERRUPT
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        OUT     BR,AORB!OBR             ;INTERRUPT
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AORB!OBR>>
        SP      IBUS,INCON,SPO          ;RESTORE INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!INCON!SPO>
        ALWAYS  NIDLE4
        MICPC=MICPC+1
        <JUMP!ALCOND!<NIDLE4-INIT&3000*4>!<NIDLE4-INIT&777/2>>
        ;
```

```
1783
1784  021500
 (1)            001467
 (1)  021500    000477
1785  021502
 (1)            001470
 (1)  021502    063265
1786  021504
 (1)            001471
 (1)  021504    070074
1787  021506
 (1)            001472
 (1)  021506    077220
1788  021510
 (1)            001473
 (1)  021510    074601
1789  021512
 (1)            001474
 (1)  021512    001620
1790  021514
 (1)            001475
 (1)  021514    116502
1791  021516
 (1)            001476
 (1)  021516    054660
1792  021520
 (1)            001477
 (1)  021520    060365
1793  021522
 (1)            001500
 (1)  021522    115113
1794  021524
 (1)            001501
 (1)  021524    115510
1795  021526
 (1)            001502
 (1)  021526    020540
1796  021530
 (1)            001503
 (1)  021530    116106
1801  021532
 (1)            001504
 (1)  021532    000660
1802  021534
 (1)            001505
 (1)  021534    100450
1803
1804  021536
 (1)            001506
 (1)  021536    000607
1805  021540
 (1)            001507
 (1)  021540    100450
1806
1807  021542
 (1)            001510
```

```
                ;FUGITIVE RECEIVE ROUTINES---DON'T FIT IN PAGE
RH1:    BRWRTE  IMM,77
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<77>>
        SP      BR,AANDB,SP5
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP5>
        LDMA    BR,<INCA!SP14>          ;LOAD ADDRESS OF CURRENT COUNT
        MICPC=MICPC+1
        <MOVE!LDMAR!BR!<INCA!SP14>>
        SP      BR!INCMAR,SELB,SPO      ;SAVE MASK
        MICPC=MICPC+1
        <MOVE!SPX!BR!INCMAR!SELB!SPO>
        BRWRTE  BR!INCMAR,SELA!SP1      ;READ STATUS BYTE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!INCMAR! SELA!SP1>>
        BRSHFT                          ;SHIFT IT RIGHT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR1     RH2                     ;NO BUFFER ASSIGNED IN MAINT MODE
        MICPC=MICPC+1
        <JUMP!BR1CON!<RH2-INIT&3000*4>!<RH2-INIT&777/2>>
        BRWRTE  MEMX!INCMAR,AANDB!SPO   ;GET HIGH BYTE COUNT BITS
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!INCMAR!<AANDB!SPO>>
        CMP     BR,SP5                  ;COMPARE HIGH ORDER BITS OF COUNT
        MICPC=MICPC+1
        <SUBTC!BR!SP5>
        C       RCFATL                  ;IF CARRY--TOO BIG ERROR
        MICPC=MICPC+1
        <JUMP!CCOND!<RCFATL-INIT&3000*4>!<RCFATL-INIT&777/2>>
        Z       RCLOW                   ;IF EQUAL COMPARE LOW ORDER BITS OF COUNT
        MICPC=MICPC+1
        <JUMP!ZCOND!<RCLOW-INIT&3000*4>!<RCLOW-INIT&777/2>>
RH2:    BRWRTE  IBUS,IOBA1                      ;READ LOW BYTE OF IN BA
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<IOBA1>>
        BRO     RCVODD                          ;IF SET IS ODD TRANSFER
        MICPC=MICPC+1
        <JUMP!BROCON!<RCVODD-INIT&3000*4>!<RCVODD-INIT&777/2>>
        STATE   RCVKEO
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVKEO-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
;
RCVODD: STATE   RCVKO1
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVKO1-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
;
RCLOW:  CMP     MEMX,SP4                        ;COMPARE LOW ORDER BITS OF COUNT
        MICPC=MICPC+1
```

```
                        (1)  021542  040364
1808  021544                <SUBTC!MEMX!SP4>
                        (1)  021544  001511
                              C       RCFATL                          ;CARRY--TOO BIG
                        (1)  021544  115113
1809  021546                MICPC=MICPC+1
                        (1)  021546  001512
                              <JUMP!CCOND!<RCFATL-INIT&3000*4>!<RCFATL-INIT&777/2>>
                        (1)  021546  114502
1810  021550                ALWAYS  RH2                             ;ELSE CONTINUE
                        (1)  021550  001513
                              MICPC=MICPC+1
                        (1)  021550  010151
1811  021552                <JUMP!ALCOND!<RH2-INIT&3000*4>!<RH2-INIT&777/2>>
                        (1)  021552  001514
RCFATL:                       LDMA    IMM,T
                        (1)  021552  016402
1812  021554                MICPC=MICPC+1
                        (1)  021554  001515
                              <MOVE!LDMAR!IMM!<T&377>>
                        (1)  021554  002711
1813  021556                MEMINC  IMM,2
                        (1)  021556  001516
                              MICPC=MICPC+1
                        (1)  021556  010175
1814  021560                <MOVE!WRMEM!INCMAR!IMM!<2>>
                        (1)  021560  001517
                              MEM     IMM,311
                        (1)  021560  036540
1815  021562                MICPC=MICPC+1
                        (1)  021562  001520
                              <MOVE!WRMEM!IMM!<311>>
                        (1)  021562  036560
1816  021564                LDMA    IMM,<<RTHRS+1>>                 ;ADDRESS ERROR LINK
                        (1)  021564  001521
                              MICPC=MICPC+1
                        (1)  021564  000420
1817  021566                <MOVE!LDMAR!IMM!<<RTHRS+1>&377>>
                        (1)  021566  001522
                              MEMINC  IBUS,IOBA1
                        (1)  021566  016400
1818  021570                MICPC=MICPC+1
                        (1)  021570  001523
                              <MOVE!WRMEM!INCMAR!IBUS!<IOBA1>>
                        (1)  021570  062620
1819  021572                MEMINC  IBUS,IOBA2
                        (1)  021572  001524
                              MICPC=MICPC+1
                        (1)  021572  002212
1820  021574                <MOVE!WRMEM!INCMAR!IBUS!<IOBA2>>
                        (1)  021574  001525
                              BRWRTE  IMM,20
                        (1)  021574  003002
1821  021576                MICPC=MICPC+1
                        (1)  021576  001526
                              <MOVE!WRTEBR!IMM!<20>>
                        (1)  021576  003001
1822  021600                MEMINC  IMM,0
                        (1)  021600  001527
                              MICPC=MICPC+1
                        (1)  021600  114666
1823  021602                <MOVE!WRMEM!INCMAR!IMM!<0>>
                        (1)  021602  001530
                              MEM     BR,SELB
                        (1)  021602  040757
1824  021604                MICPC=MICPC+1
                        (1)  021604  001531
                              <MOVE!WRMEM!BR!<SELB>>
                        (1)  021604  107562
1825  021606                OUTPUT  IMM,<200!ORCVCO>                ;FLUSH INPUT SILO
                        (1)  021606  001532
                              MICPC=MICPC+1
                        (1)  021606  070200
1826  021610                <MOVE!WROUT!IMM!<200!ORCVCO>>
                              SP      IMM,SP2,2                       ;INHIBIT FURTHER TRANSMISSIONS
                              MICPC=MICPC+1
                              <MOVE!SPX!IMM!SP2!2>
                              SP      IMM,1,SP1       ;SET INIT MODE IN PORT STATUS WORD
                              MICPC=MICPC+1
                              <MOVE!SPX!IMM!1!SP1>
                              ALWAYS  NTRS1
                              MICPC=MICPC+1
                              <JUMP!ALCOND!<NTRS1-INIT&3000*4>!<NTRS1-INIT&777/2>>
TDON3:                        BRWRTE  MEMX,SUB!SP17                   ;COMPARE RESPONSE TO MSG NO
                              MICPC=MICPC+1
                              <MOVE!WRTEBR!MEMX!<SUB!SP17>>
                              BR7     RH3                             ;IF NEGATIVE EXIT
                              MICPC=MICPC+1
                              <JUMP!BR7CON!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
TDON2:                        LDMA    BR,SELA!SP0                     ;ADDRESS THE TRANSMITLINK
                              MICPC=MICPC+1
                              <MOVE!LDMAR!BR!<SELA!SP0>>
                              MEM     IMM,0                           ;TURN OF ASSIGNEDAND TMTED BITS IN FLAG
```

```
  (1)           001533
  (1)  021610   002400
1827   021612
  (1)           001534
  (1)  021612   010067
1828   021614
  (1)           001535
  (1)  021614   002471
1829   021616
  (1)           001536
  (1)  021616   000543
1830   021620
  (1)           001537
  (1)  021620   060360
1831   021622
  (1)           001540
  (1)  021622   115543
1832   021624
  (1)           001541
  (1)  021624   000406
1833   021626
  (1)           001542
  (1)  021626   062400
1834   021630
  (1)           001543
  (1)  021630   010241
1835   021632
  (1)           001544
  (1)  021632   053223
1836
1837   021634
  (1)           001545
  (1)  021634   016600
1838   021636
  (1)           001546
  (1)  021636   062460
1839   021640
  (1)           001547
  (1)  021640   010241
1840   021642
  (1)           001550
  (1)  021642   002642
1841   021644
  (1)           001551
  (1)  021644   000776
1842   021646
  (1)           001552
  (1)  021646   060363
1843   021650
  (1)           001553
  (1)  021650   115556
1844   021652
  (1)           001554
  (1)  021652   000402
1845   021654
  (1)           001555
```

```
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<0>>
        LDMA    IMM,STC
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        MEM     IMM,TML1                        ;ASSUME WRAPAROUND
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<TML1>>
        BRWRTE  IMM,TML8                        ;WRAPAROUND?
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TML8>>
        CMP     BR,SP0
        MICPC=MICPC+1
        <SUBTC!BR!SP0>
        Z       TDON4                           ;YES
        MICPC=MICPC+1
        <JUMP!ZCOND!<TDON4-INIT&3000*4>!<TDON4-INIT&777/2>>
        BRWRTE  IMM,6                           ;OFFSET FOR NEXT TMT LINK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
        MEM     BR,ADD!SP0                      ;UPDATE THE POINTER
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<ADD!SP0>>
TDON4:  LDMA    IMM,NXTSP                       ;ADDRESS DONE LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NXTSP&377>>
        LDMA    MEMX,SELB!SPX!SP3               ;ADDRESS THE LINK,COPYING
        MICPC=MICPC+1
        <MOVE!LDMAR!MEMX!<SELB!SPX!SP3>>
                                                ;ITS ADDRESS TO SP0
        MEMINC  IMM,200                         ;WRITE THE INTERRUPT TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<200>>
        MEM     BR,INCA!SP0                     ;COPY ACTUAL LINK ADDRESS
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<INCA!SP0>>
        LDMA    IMM,NXTSP                       ;ADDRESS PTR INT STACK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NXTSP&377>>
        MEM     IMM,INTSTK                      ;ASSUME WRAP AROUND
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<INTSTK>>
        BRWRTE  IMM,<<MMEND-2>>                 ;ADDRESS ENDOFINT STACK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<<MMEND-2>>>
        CMP     BR,SP3                          ;WRAPAROUND?
        MICPC=MICPC+1
        <SUBTC!BR!SP3>
        Z       TDON40                          ;YES---BRANCH
        MICPC=MICPC+1
        <JUMP!ZCOND!<TDON40-INIT&3000*4>!<TDON40-INIT&777/2>>
        BRWRTE  IMM,2                           ;OFFSET TO NEXT PAIR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        MEM     BR,ADD!SP3                      ;UPDATE POINTER
        MICPC=MICPC+1
```

```
  (1)  021654  062403                    <MOVE!WRMEM!BR!<ADD!SP3>>
1846   021656                  TDON40: BRWRTE  IMM,20                  ;WRITE INTERUPT PENDING
  (1)          001556                  MICPC=MICPC+1
  (1)  021656  000420                  <MOVE!WRTEBR!IMM!<20>>
1847   021660                          SP      BR,AORB,SP1             ;IN PORT STATUS WORD
  (1)          001557                  MICPC=MICPC+1
  (1)  021660  063301                  <MOVE!SPX!BR!AORB!SP1>
1848   021662                          LDMA    IMM,ETC                 ;ADDRESS NEXT EMPTY PTR
  (1)          001560                  MICPC=MICPC+1
  (1)  021662  010070                  <MOVE!LDMAR!IMM!<ETC&377>>
1849   021664                          SP      MEMX,SELB,SPO           ;COPY IT TO SPO
  (1)          001561                  MICPC=MICPC+1
  (1)  021664  043220                  <MOVE!SPX!MEMX!SELB!SPO>
1850   021666                          LDMA    IMM,STC                 ;GET NEXT DONE PTR
  (1)          001562                  MICPC=MICPC+1
  (1)  021666  010067                  <MOVE!LDMAR!IMM!<STC&377>>
1851   021670                          CMP     MEMX,SPO                ;IDENTICAL?
  (1)          001563                  MICPC=MICPC+1
  (1)  021670  040360                  <SUBTC!MEMX!SPO>
1852   021672                          Z       RH3                     ;FINISH PROCESSING HEADER
  (1)          001564                  MICPC=MICPC+1
  (1)  021672  105562                  <JUMP!ZCOND!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
1853
1854   021674                  TDON1:  LDMA    IMM,ISP17               ;GET LAST ACKED
  (1)          001565                  MICPC=MICPC+1
  (1)  021674  010153                  <MOVE!LDMAR!IMM!<ISP17&377>>
1855   021676                          SP      MEMX,SELB,SP17          ;STORE IT IN SP17
  (1)          001566                  MICPC=MICPC+1
  (1)  021676  043237                  <MOVE!SPX!MEMX!SELB!SP17>
1856   021700                          LDMA    IMM,STC                 ;GET START OF TMT CHAIN
  (1)          001567                  MICPC=MICPC+1
  (1)  021700  010067                  <MOVE!LDMAR!IMM!<STC&377>>
1857   021702                          LDMA    MEMX,SELB!SPBRX!SPO     ;ADDRESS THE LINK
  (1)          001570                  MICPC=MICPC+1
  (1)  021702  053620                  <MOVE!LDMAR!MEMX!<SELB!SPBRX!SPO>>
1858   021704                          BRWRTE  MEMX!INCMAR,SELB        ;GET THE FLAGS
  (1)          001571                  MICPC=MICPC+1
  (1)  021704  054620                  <MOVE!WRTEBR!MEMX!INCMAR!<SELB>>
1859   021706                          BR1     TDON3                   ;IF BUFFER ASSIGNED PROCEED
  (1)          001572                  MICPC=MICPC+1
  (1)  021706  116530                  <JUMP!BR1CON!<TDON3-INIT&3000*4>!<TDON3-INIT&777/2>>
1860   021710                          ALWAYS  RH3                     ;ELSE---EXIT
  (1)          001573                  MICPC=MICPC+1
  (1)  021710  104562                  <JUMP!ALCOND!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
1861
1862   021712                  OVRRUN: BRWRTE  IMM,4
  (1)          001574                  MICPC=MICPC+1
  (1)  021712  000404                  <MOVE!WRTEBR!IMM!<4>>
1863   021714                          ALWAYS  NTRSO
  (1)          001575                  MICPC=MICPC+1
  (1)  021714  114663                  <JUMP!ALCOND!<NTRSO-INIT&3000*4>!<NTRSO-INIT&777/2>>
1864
1865                                  ; INPUTS:
1866                                  ;       SPO = RECEIVE CHARACTER
1867
1868   021716                  PASWRD: SP      IBUS,LNOSW,SP13         ;READ PASSWD SWITCH
```

```
                001576
 (1)            023333
 (1)  021716
1869  021720
 (1)            001577
 (1)  021720    115603
1870  021722
 (1)            001600
 (1)  021722    000406
1871  021724
 (1)            001601
 (1)  021724    060360
1872  021726
 (1)            001602
 (1)  021726    115611
1873  021730
 (1)            001603
 (1)  021730    060601
1874  021732
 (1)            001604
 (1)  021732    001620
1875  021734
 (1)            001605
 (1)  021734    106740
1876  021736
 (1)            001606
 (1)  021736    001620
1877  021740
 (1)            001607
 (1)  021740    106732
1878  021742
 (1)            001610
 (1)  021742    104641
1879  021744
 (1)            001611
 (1)  021744    063164
1880  021746
 (1)            001612
 (1)  021746    000735
1881  021750
 (1)            001613
 (1)  021750    100450
1882
1883
1884
1885  021752
 (1)            001614
 (1)  021752    010001
1886  021754
 (1)            001615
 (1)  021754    002401
1887  021756
 (1)            001616
 (1)  021756    010167
1888  021760
 (1)            001617
 (1)  021760    057224
```

```
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!LNOSW!SP13>
        Z       10$                     ;IF ALL ONES NO RLD ENABLED
        MICPC=MICPC+1
        <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
        BRWRTE  IMM,6                   ;CHECK FOR ENTER MOP MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
        CMP     BR,SPO
        MICPC=MICPC+1
        <SUBTC!BR!SPO>
        Z       20$                     ;IF EQUAL   ENTER MOP
        MICPC=MICPC+1
        <JUMP!ZCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
10$:    BRWRTE  BR,SELA!SP1             ;READ STATUS BYTE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BRSHFT                          ;SHIFT IT RIGHT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR1     RHX                     ;MESSAGE WITH NO BUFFER ASSIGNED
        MICPC=MICPC+1
        <JUMP!BR1CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
        BRSHFT                          ;SHIFT RIGHT AGAIN
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR1     RCVM0                   ;DLE RECEIVED IN NORMAL MODE
        MICPC=MICPC+1
        <JUMP!BR1CON!<RCVM0-INIT&3000*4>!<RCVM0-INIT&777/2>>
        ALWAYS  RK3                     ;HANDLE MAINT MODE MESSAGE
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
20$:    SP      BR,DECA,SP4             ;COUNT FOR NUMB OF COMPARES
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP4>
        STATE   EM2
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<EM2-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
;
        .ENABL  LSB
;
RCVM1:  LDMA    IMM,NAKST               ;RESET NAKS SENT
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NAKST&377>>
        MEM     IMM,1                   ;..
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<1>>
        LDMA    IMM,BC                  ;ADDRESS ORIGINAL RECV BYTE COUNT
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<BC&377>>
        SP      MEMX!INCMAR,SELB,SP4    ;MOVE BYTE COUNT TO SP4
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!INCMAR!SELB!SP4>
```

```
1889  021762
 (1)            001620
 (1)  021762    057225          SP      MEMX!INCMAR,SELB,SP5    ;AND SP5
1890  021764                    MICPC=MICPC+1
 (1)            001621          <MOVE!SPX!MEMX!INCMAR!SELB!SP5>
 (1)  021764    062571          MEM     BR,DECA!SP11                    ;COPY SP11 FROM MEMORY
1891  021766                    MICPC=MICPC+1
 (1)            001622          <MOVE!WRMEM!BR!<DECA!SP11>>
 (1)  021766    010241          LDMA    IMM,NXTSP
1892  021770                    MICPC=MICPC+1
 (1)            001623          <MOVE!LDMAR!IMM!<NXTSP&377>>
 (1)  021770    053223          SP      MEMX!LDMAR,SELB,SP3             ;COPY TO SP3
1893  021772                    MICPC=MICPC+1
 (1)            001624          <MOVE!SPX!MEMX!LDMAR!SELB!SP3>
 (1)  021772    016604          MEMINC  IMM,204                 ;RECEIVE DONE IMAGE
1894  021774                    MICPC=MICPC+1
 (1)            001625          <MOVE!WRMEM!INCMAR!IMM!<204>>
 (1)  021774    072614          MEM     BR!LDMAR,SELA!SP14              ;COPY LINK ADDRESS TO NEXT INTER
1895  021776                    MICPC=MICPC+1
 (1)            001626          <MOVE!WRMEM!BR!LDMAR!<SELA!SP14>>
 (1)  021776    016400          MEMINC  IMM,0                   ;ZERO THE FLAGS
1896  022000                    MICPC=MICPC+1
 (1)            001627          <MOVE!WRMEM!INCMAR!IMM!<0>>
 (1)  022000    017300          SP      IMM!INCMAR,SP0,300              ;WRITE A 300 TO SP0
1897  022002                    MICPC=MICPC+1
 (1)            001630          <MOVE!SPX!IMM!INCMAR!SP0!300>
 (1)  022002    014402          BRWRTE  IMM!INCMAR,2                    ;PREPARE TO ADDRESS NEXT
1898                            MICPC=MICPC+1
1899                            <MOVE!WRTEBR!IMM!INCMAR!<2>>
1900  022004                                                            ;INTERRUPT STACK AND INCREMENT
 (1)            001631                                                  ;THE MAR
 (1)  022004    042660          MEM     MEMX,AANDB!SP0                  ;MASK OFF ORIGINAL HIGH BYTE
1901                            MICPC=MICPC+1
1902  022006                    <MOVE!WRMEM!MEMX!<AANDB!SP0>>
 (1)            001632                                                  ;OF COUNT SAVING EXTENDED MEM BITS
 (1)  022006    056705          MEMINC  MEMX,AORB!SP5                   ;COPY TO MEMORY LINK
1903  022010                    MICPC=MICPC+1
 (1)            001633          <MOVE!WRMEM!INCMAR!MEMX!<AORB!SP5>>
 (1)  022010    076604          MEMINC  BR,SELA!SP4
1904  022012                    MICPC=MICPC+1
 (1)            001634          <MOVE!WRMEM!INCMAR!BR!<SELA!SP4>>
 (1)  022012    010241          LDMA    IMM,NXTSP                       ;ADDRESS NEXT INT STACK
1905  022014                    MICPC=MICPC+1
 (1)            001635          <MOVE!LDMAR!IMM!<NXTSP&377>>
 (1)  022014    062403          MEM     BR,ADD!SP3
1906  022016                    MICPC=MICPC+1
 (1)            001636          <MOVE!WRMEM!BR!<ADD!SP3>>
 (1)  022016    000776          BRWRTE  IMM,<<MMEND-2>>                 ;ADDRESSEND OF INT STACK
1907  022020                    MICPC=MICPC+1
 (1)            001637          <MOVE!WRTEBR!IMM!<<MMEND-2>>>
 (1)  022020    060363          CMP     BR,SP3                          ;WRAP AROUND
1908  022022                    MICPC=MICPC+1
 (1)            001640          <SUBTC!BR!SP3>
 (1)  022022    115651          Z       40$                             ;IF YES-- BRANCH
1909  022024                    MICPC=MICPC+1
1910  022024            20$:    <JUMP!ZCOND!<40$-INIT&3000*4>!<40$-INIT&777/2>>
                                BRWRTE  IMM,5                           ;INDEX TO NEXT BUFFER
```

```
                        MICPC=MICPC+1
                        <MOVE!WRTEBR!IMM!<5>>
                        SP      BR,ADD,SP14                     ;UPDATE COPY OF POINTER
                        MICPC=MICPC+1
                        <MOVE!SPX!BR!ADD!SP14>
                        BRWRTE  IMM,STC                         ;ADDRESS OF WRAP AROUND POINT
                        MICPC=MICPC+1
                        <MOVE!WRTEBR!IMM!<STC>>
                        CMP     BR,SP14                         ;WRAPAROUND?
                        MICPC=MICPC+1
                        <SUBTC!BR!SP14>
                        Z       50$                             ;IF YES---BRANCH
                        MICPC=MICPC+1
                        <JUMP!ZCOND!<50$-INIT&3000*4>!<50$-INIT&777/2>>
30$:                    BRWRTE  IMM,20                          ;MASK FOR INTERUPT PENDING
                        MICPC=MICPC+1
                        <MOVE!WRTEBR!IMM!<20>>
                        SP      DP,AORB,SP1                     ;UPDATE PORT STATUS WORD
                        MICPC=MICPC+1
                        <MOVE!SPX!DP!AORB!SP1>
                        ALWAYS  FLUSH
                        MICPC=MICPC+1
                        <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
40$:                    MEM     IMM,INTSTK                      ;POINT TO START OF INTERRUPT STACK
                        MICPC=MICPC+1
                        <MOVE!WRMEM!IMM!<INTSTK>>
                        ALWAYS  20$
                        MICPC=MICPC+1
                        <JUMP!ALCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
50$:                    BRWRTE  IMM,RCL1                        ;POINT TO START OF RECEIVE QUEUE
                        MICPC=MICPC+1
                        <MOVE!WRTEBR!IMM!<RCL1>>
                        SP      BR,SELB,SP14
                        MICPC=MICPC+1
                        <MOVE!SPX!BR!SELB!SP14>
                        ALWAYS  30$
                        MICPC=MICPC+1
                        <JUMP!ALCOND!<30$-INIT&3000*4>!<30$-INIT&777/2>>
                        .DSABL  LSB
NTHRES:                 LDMA    IMM,ST
                        MICPC=MICPC+1
                        <MOVE!LDMAR!IMM!<ST&377>>
                        SPBR    MEMX,SELB,SPO
                        MICPC=MICPC+1
                        <MOVE!SPBRX!MEMX!SELB!SPO>
                        BRWRTE  BR,ADD!SPO                              ;SHIFT LEFT
                        MICPC=MICPC+1
                        <MOVE!WRTEBR!BR!<ADD!SPO>>
                        BR4     OVRRUN
                        MICPC=MICPC+1
                        <JUMP!BR4CON!<OVRRUN-INIT&3000*4>!<OVRRUN-INIT&777/2>>
                        BRWRTE  IMM,1
                        MICPC=MICPC+1
                        <MOVE!WRTEBR!IMM!<1>>
ERRXX:
NTRSO:                  LDMA    IMM,<<RTHRS+3>>
```

```
  (1)           001663
  (1)  022070   010177          MICPC=MICPC+1
  (1)  022071   000000          <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
1939   022072                   MEMINC  IMM,0
  (1)           001664          MICPC=MICPC+1
  (1)  022072   016400          <MOVE!WRMEM!INCMAR!IMM!<0>>
1940   022074                   MEM     BR,SELB
  (1)           001665          MICPC=MICPC+1
  (1)  022074   062620          <MOVE!WRMEM!BR!<SELB>>
1941   022076           NTRS1:  LDMA    IMM,NXTSP
  (1)           001666          MICPC=MICPC+1
  (1)  022076   010241          <MOVE!LDMAR!IMM!<NXTSP&377>>
1942   022100                   LDMA    MEMX,SELB!SPX!SP0
  (1)           001667          MICPC=MICPC+1
  (1)  022100   053220          <MOVE!LDMAR!MEMX!<SELB!SPX!SP0>>
1943   022102                   MEMINC  IMM,201
  (1)           001670          MICPC=MICPC+1
  (1)  022102   016601          <MOVE!WRMEM!INCMAR!IMM!<201>>
1944   022104                   MEM     IMM,<<RTHRS>>
  (1)           001671          MICPC=MICPC+1
  (1)  022104   002574          <MOVE!WRMEM!IMM!<<RTHRS>>>
1945   022106                   LDMA    IMM,NXTSP
  (1)           001672          MICPC=MICPC+1
  (1)  022106   010241          <MOVE!LDMAR!IMM!<NXTSP&377>>
1946   022110                   MEM     IMM,INTSTK              ;ASSUME QUEUE WRAP AROUND
  (1)           001673          MICPC=MICPC+1
  (1)  022110   002642          <MOVE!WRMEM!IMM!<INTSTK>>
1947   022112                   BRWRTE  IMM,<<MMEND-2>>
  (1)           001674          MICPC=MICPC+1
  (1)  022112   000776          <MOVE!WRTEBR!IMM!<<MMEND-2>>>
1948   022114                   CMP     BR,SP0
  (1)           001675          MICPC=MICPC+1
  (1)  022114   060360          <SUBTC!BR!SP0>
1949   022116                   Z       NTRS2                   ;IT DID WRAP AROUND
  (1)           001676          MICPC=MICPC+1
  (1)  022116   115701          <JUMP!ZCOND!<NTRS2-INIT&3000*4>!<NTRS2-INIT&777/2>>
1950   022120                   BRWRTE  IMM,2                   ;OFFSET TO NEXT PAIR
  (1)           001677          MICPC=MICPC+1
  (1)  022120   000402          <MOVE!WRTEBR!IMM!<2>>
1951   022122                   MEM     BR,ADD!SP0              ;UPDATE QUEUE POINTER
  (1)           001700          MICPC=MICPC+1
  (1)  022122   062400          <MOVE!WRMEM!BR!<ADD!SP0>>
1952   022124           NTRS2:  BRWRTE  IMM,20
  (1)           001701          MICPC=MICPC+1
  (1)  022124   000420          <MOVE!WRTEBR!IMM!<20>>
1953   022126                   SPBR    BR,AORB,SP1
  (1)           001702          MICPC=MICPC+1
  (1)  022126   063701          <MOVE!SPBRX!BR!AORB!SP1>
1954   022130                   BRO     TAB1                    ;FLAGGED BY ERROR TYPE
  (1)           001703          MICPC=MICPC+1
  (1)  022130   116334          <JUMP!BROCON!<TAB1-INIT&3000*4>!<TAB1-INIT&777/2>>
1955   022132           SNAK:   LDMA    IMM,ISP11
  (1)           001704          MICPC=MICPC+1
  (1)  022132   010171          <MOVE!LDMAR!IMM!<ISP11&377>>
1956   022134                   SP      MEMX,SELB,SP11
  (1)           001705          MICPC=MICPC+1
  (1)  022134   043231          <MOVE!SPX!MEMX!SELB!SP11>
```

```
1957  022136
 (1)            001706
 (1)  022136    063071          SP      BR,INCA,SP11                    ;INCREMENT MSG EXPECTED
                                MICPC=MICPC+1
                                <MOVE!SPX!BR!INCA!SP11>
1958  022140                    BRWRTE  IMM,1                           ;UNNUMB PENING BIT TO BR
 (1)            001707         MICPC=MICPC+1
 (1)  022140    000401          <MOVE!WRTEBR!IMM!<1>>
1959  022142                    SP      BR,AORB,SP10                    ;UPDATE LINE STATUS WORD
 (1)            001710         MICPC=MICPC+1
 (1)  022142    063310          <MOVE!SPX!BR!AORB!SP10>
1960  022144                    ALWAYS  FLUSH
 (1)            001711         MICPC=MICPC+1
 (1)  022144    104415          <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
1961                            ;
1962  022146                    EMTRIG: BRWRTE  IMM,24
 (1)            001712         MICPC=MICPC+1
 (1)  022146    000424          <MOVE!WRTEBR!IMM!<24>>
1963  022150                    OUTPUT  BR,<SELB!OBA1>
 (1)            001713         MICPC=MICPC+1
 (1)  022150    062226          <MOVE!WROUT!BR!<SELB!OBA1>>
1964  022152                    BRWRTE  IMM,0
 (1)            001714         MICPC=MICPC+1
 (1)  022152    000400          <MOVE!WRTEBR!IMM!<0>>
1965  022154                    OUTPUT  BR,<SELB!OBA2>
 (1)            001715         MICPC=MICPC+1
 (1)  022154    062227          <MOVE!WROUT!BR!<SELB!OBA2>>
1966  022156                    SPBR    IBUS,BM873,SPO          ;READ BM873 ADDRESS---
 (1)            001716         MICPC=MICPC+1
 (1)  022156    023740          <MOVE!SPBRX!IBUS!BM873!SPO>
1967  022160                    OUTPUT  BR,SELB!OUTDA1          ;SET UP LOW BYTE OF ADDRESS
 (1)            001717         MICPC=MICPC+1
 (1)  022160    062222          <MOVE!WROUT!BR!<SELB!OUTDA1>>
1968  022162                    BRWRTE  IMM,366                 ;HIGH BYTE BASE FOR ROM BOOT
 (1)            001720         MICPC=MICPC+1
 (1)  022162    000766          <MOVE!WRTEBR!IMM!<366>>
1969  022164                    OUTPUT  BR,SELB!OUTDA2          ;
 (1)            001721         MICPC=MICPC+1
 (1)  022164    062223          <MOVE!WROUT!BR!<SELB!OUTDA2>>
1970  022166                    EM6:    BRWRTE  IMM,21          ;MASK FOR TIMER AND ALSO TO START NPR
 (1)            001722         MICPC=MICPC+1
 (1)  022166    000421          <MOVE!WRTEBR!IMM!<21>>
1971  022170                    OUT     BR,<SELB!OBR>
 (1)            001723         MICPC=MICPC+1
 (1)  022170    061231          <MOVE!WROUTX!BR!<SELB!OBR>>
1972  022172                    OUT     BR,<SELB!ONPR>
 (1)            001724         MICPC=MICPC+1
 (1)  022172    061230          <MOVE!WROUTX!BR!<SELB!ONPR>>
1973  022174                    EM1:    BRWRTE  IBUS,NPR                ;READ NPR CONTROL
 (1)            001725         MICPC=MICPC+1
 (1)  022174    120600          <MOVE!WRTEBR!IBUS!<NPR>>
1974  022176                    BRO     CKTIME
 (1)            001726         MICPC=MICPC+1
 (1)  022176    102371          <JUMP!BROCON!<CKTIME-INIT&3000*4>!<CKTIME-INIT&777/2>>
1975  022200                    MEMADR  RM1                             ;IF NPR DONE
 (1)            001727         MICPC=MICPC+1
 (1)  022200    002420          <MOVE!WRMEM!<RM1-INIT&777/2>>
1976  022202                    ALWAYS  ACLOW
```

```
  (1)           001730
  (1)  022202   100764          MICPC=MICPC+1
                               <JUMP!ALCOND!<ACLOW-INIT&3000*4>!<ACLOW-INIT&777/2>>
1977   022204                  TABUPD: SPBR    IBUS,RCVCON,SPO         ;READ RECEIVER CONTROL REG
  (1)           001731          MICPC=MICPC+1
  (1)  022204   023640          <MOVE!SPBRX!IBUS!RCVCON!SPO>
1978   022206                  BRWRTE  BR,ADD!SPO              ;SHIFT LEFT
  (1)           001732          MICPC=MICPC+1
  (1)  022206   060400          <MOVE!WRTEBR!BR!<ADD!SPO>>
1979   022210                  BR7     IDLE                    ;RECEIVE ACTIVE--IDLE
  (1)           001733          MICPC=MICPC+1
  (1)  022210   103451          <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1980   022212                  TAB1:   LDMA    IMM,IMG10
  (1)           001734          MICPC=MICPC+1
  (1)  022212   010154          <MOVE!LDMAR!IMM!<IMG10&377>>
1981   022214                  BRWRTE  IMM,2
  (1)           001735          MICPC=MICPC+1
  (1)  022214   000402          <MOVE!WRTEBR!IMM!<2>>
1982   022216                  MEMINC  BR,AANDB!SP10           ;SAVE BIT 1 OF SP10
  (1)           001736          MICPC=MICPC+1
  (1)  022216   076670          <MOVE!WRMEM!INCMAR!BR!<AANDB!SP10>>
1983   022220                  MEMINC  BR,SELA!SP11            ;SAVE SP11
  (1)           001737          MICPC=MICPC+1
  (1)  022220   076611          <MOVE!WRMEM!INCMAR!BR!<SELA!SP11>>
1984   022222                  MEMINC  BR,SELA!SP12            ;SAVE SP12
  (1)           001740          MICPC=MICPC+1
  (1)  022222   076612          <MOVE!WRMEM!INCMAR!BR!<SELA!SP12>>
1985   022224                  MEMINC  BR,SELA!SP14            ;SAVE SP14
  (1)           001741          MICPC=MICPC+1
  (1)  022224   076614          <MOVE!WRMEM!INCMAR!BR!<SELA!SP14>>
1986   022226                  MEMINC  BR,SELA!SP16            ;SAVE SP16
  (1)           001742          MICPC=MICPC+1
  (1)  022226   076616          <MOVE!WRMEM!INCMAR!BR!<SELA!SP16>>
1987   022230                  MEMINC  BR,SELA!SP17            ;SAVE SP17
  (1)           001743          MICPC=MICPC+1
  (1)  022230   076617          <MOVE!WRMEM!INCMAR!BR!<SELA!SP17>>
1988                                                          ;MAR NOW POINTS TO BASE
1989   022232                  STATE   RB2             ;DO NOT CHANGE BR UNTIL RB0
  (1)           001744          MICPC=MICPC+1
  (1)  022232   000460          <MOVE!WRTEBR!IMM!<RB2-INIT&777/2>>
1990   022234                  LDMA    IMM,TABST       ;POINT TO TABLE UPDATE STATE
  (1)           001745          MICPC=MICPC+1
  (1)  022234   010210          <MOVE!LDMAR!IMM!<TABST&377>>
1991   022236                  PSTATE  TBU1                    ;NEW PORT STATE ADDRESS
  (1)  022236                   MEM     IMM,<<TBU1-INIT&777/2>>
  (2)           001746          MICPC=MICPC+1
  (2)  022236   002774          <MOVE!WRMEM!IMM!<<TBU1-INIT&777/2>>>
1992   022240                  SP      IMM,4,SP4               ;INITIALIZE COUNT
  (1)           001747          MICPC=MICPC+1
  (1)  022240   003004          <MOVE!SPX!IMM!4!SP4>
1993                                                          ;NOTE: FIRST 6 RAM LOCATIONS ARE NOT WRITTEN
1994                                                          ;TO CORE TABLE.
1995   022242                  LDMA    IMM,BASE
  (1)           001750          MICPC=MICPC+1
  (1)  022242   010017          <MOVE!LDMAR!IMM!<BASE&377>>
1996   022244                  ALWAYS  RB0
  (1)           001751          MICPC=MICPC+1
```

```
  (1)  022244  104442                   <JUMP!ALCOND!<RB0-INIT&3000*4>!<RB0-INIT&777/2>>
1997   022246                EC2:       BRWRTE  IMM,2                          ;INCREMENT COUNT/TEST
  (1)          001752                   MICPC=MICPC+1
  (1)  022246  000402                   <MOVE!WRTEBR!IMM!<2>>
1998   022250                           SP      BR,ADD,SP4
  (1)          001753                   MICPC=MICPC+1
  (1)  022250  063004                   <MOVE!SPX!BR!ADD!SP4>
1999   022252                           SP      IBUS,IOBA1,SP0                 ;POINT TO NEXT ADDRESS
  (1)          001754                   MICPC=MICPC+1
  (1)  022252  023140                   <MOVE!SPX!IBUS!IOBA1!SP0>
2000   022254                           OUTPUT  BR,ADD!OBA1
  (1)          001755                   MICPC=MICPC+1
  (1)  022254  062006                   <MOVE!WROUT!BR!<ADD!OBA1>>
2001   022256                           SP      IBUS,IOBA2,SP0
  (1)          001756                   MICPC=MICPC+1
  (1)  022256  023160                   <MOVE!SPX!IBUS!IOBA2!SP0>
2002   022260                           OUTPUT  BR,AC!OBA2
  (1)          001757                   MICPC=MICPC+1
  (1)  022260  062107                   <MOVE!WROUT!BR!<AC!OBA2>>
2003   022262                           C       TABMXT
  (1)          001760                   MICPC=MICPC+1
  (1)  022262  105366                   <JUMP!CCOND!<TABMXT-INIT&3000*4>!<TABMXT-INIT&777/2>>
2004   022264                ECX:       BRWRTE  BR,SELA!SP1                    ;READ PORT STATUS
  (1)          001761                   MICPC=MICPC+1
  (1)  022264  060601                   <MOVE!WRTEBR!BR!<SELA!SP1>>
2005   022266                           BR0     30$                            ;INIT MODE, WRITE OUT 200 BYTES
  (1)          001762                   MICPC=MICPC+1
  (1)  022266  116374                   <JUMP!BR0CON!<30$-INIT&3000*4>!<30$-INIT&777/2>>
2006                                                                          ;OTHERWISE ONLY WRITE OUT ERROR COUNTERS
2007   022270                           BRWRTE  BR!LDMAR,SELA!SP4              ;READ COUNTER
  (1)          001763                   MICPC=MICPC+1
  (1)  022270  070604                   <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
2008   022272                           BR4     20$                            ;ALL DONE
  (1)          001764                   MICPC=MICPC+1
  (1)  022272  117371                   <JUMP!BR4CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
2009   022274                10$:       OUTPUT  MEMX!INCMAR,SELB!OUTDA1        ;STORE COUNTS OF ERRORS
  (1)          001765                   MICPC=MICPC+1
  (1)  022274  056222                   <MOVE!WROUT!MEMX!INCMAR!<SELB!OUTDA1>>
2010   022276                           OUTPUT  MEMX!INCMAR,SELB!OUTDA2
  (1)          001766                   MICPC=MICPC+1
  (1)  022276  056223                   <MOVE!WROUT!MEMX!INCMAR!<SELB!OUTDA2>>
2012   022300                           SP      IBUS,NPR,SP0
  (1)          001767                   MICPC=MICPC+1
  (1)  022300  123200                   <MOVE!SPX!IBUS!NPR!SP0>
2014   022302                           ALWAYS  RK8
  (1)          001770                   MICPC=MICPC+1
  (1)  022302  104622                   <JUMP!ALCOND!<RK8-INIT&3000*4>!<RK8-INIT&777/2>>
2015   022304                20$:       LDMA    IMM,TABST
  (1)          001771                   MICPC=MICPC+1
  (1)  022304  010210                   <MOVE!LDMAR!IMM!<TABST&377>>
2016   022306                           PSTATE  I3
  (1)  022306                           MEM     IMM,<<I3-INIT&777/2>>
  (2)          001772                   MICPC=MICPC+1
  (2)  022306  002460                   <MOVE!WRMEM!IMM!<<I3-INIT&777/2>>>
2017   022310                           ALWAYS  RM1
  (1)          001773                   MICPC=MICPC+1
```

```
  (1)  022310  104420
2018   022312
  (1)          001774
  (1)  022312  070604
2019   022314
  (1)          001775
  (1)  022314  117771
2020   022316
  (1)          001776
  (1)  022316  114765
2021   022320
  (1)          001777
  (1)  022320  000000
2022
2023           000001

. ABS.  022322     000

 ERRORS DETECTED:  0

 ,DDCMP/CRF/DS:CRF←DMCHGH,HILOW,DDCHGH
 RUN-TIME: 5 8 0 SECONDS
 RUN-TIME RATIO: 68/13=4.9
 CORE USED:  6K  (11 PAGES)

 1671                          03000
```

```
                ‹JUMP!ALCOND!‹RM1-INIT&3000*4›!‹RM1-INIT&777/2››
30$:    BRWRTE  BR!LDMAR,SELA!SP4       ;READ COUNTER
        MICPC=MICPC+1
        ‹MOVE!WRTEBR!BR!LDMAR!‹SELA!SP4››
        BR7     20$                     ;ALL DONE
        MICPC=MICPC+1
        ‹JUMP!BR7CON!‹20$-INIT&3000*4›!‹20$-INIT&777/2››
        ALWAYS  10$                     ;KEEP GOING
        MICPC=MICPC+1
        ‹JUMP!ALCOND!‹10$-INIT&3000*4›!‹10$-INIT&777/2››
        $ZERO
        MICPC=MICPC+1
        000000
        ;
        .END
```

```
                                02800

                                ;*************************** TEST 1 ****************************
                                ;*THIS IS A SPECIAL TEST WHICH IS ONLY EXECUTED  ONE TIME.
                                ;*THE FIRST PASS AFTER THE DIAGNOSTIC IS LOADED. IT TYPES ON
                                ;*THE CONSOLE THE PART NUMBERS OF THE CROMS WHICH THIS
                                ;*REVISION SUPPORTS. TO FORCE A TYPE OUT PATCH LOCATION
                                ;*ROMNUM: TO A ZERO.
                                ;:***************************************************************

                                ;  TEST 1
                                ;--------------
022322  012737  000001  001226  TST1:   MOV     #1,TSTNO
022330  012737  022760  001216          MOV     #TST2,NEXT
                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
022336  005737  022360                  TST     ROMNUM          ;FIRST TIME HERE?
022342  001002                          BNE     1$              ;SKIP IF NOT
022344  104402  022362                  TYPE,   ROM1            ;TYPE PART NUMBERS
022350  012737  177777  022360  1$:     MOV     #-1,ROMNUM      ;SET FLAG TO ONLY TYPE ONCE
022356  104400                          SCOPE                   ;
022360  000000                  ROMNUM: 0
022362  005377  055104  046504  ROM1:   .ASCII  <377><12>/DZDMG-C SUPPORTS THE FOLLOWING CROM PART NUMBERS:/
(1)     022445     377  042012  041515          .ASCII  <377><12>/DMC11-AR  (M8200-YA)/
(1)     022473     377  031462  032055          .ASCII  <377>/23-414A9/<15><12>/23-415A9/<15><12>/23-416A9/
(1)     022530  005015  031462  032055          .ASCII  <15><12>/23-417A9/<15><12>/23-418A9/<15><12>/23-419A9/
(1)     022566  005015  031462  032055          .ASCII  <15><12>/23-420A9/<15><12>/23-421A9/
(1)     022612  005377  046504  030503          .ASCII  <377><12>/DMC11-AL  (M8200-YB)/
(1)     022640  031377  026463  034463          .ASCII  <377>/23-392A9/<15><12>/23-393A9/<15><12>/23-394A9/
(1)     022675     015  031012  026463          .ASCII  <15><12>/23-395A9/<15><12>/23-396A9/<15><12>/23-397A9/
(1)     022733     015  031012  026463          .ASCII  <15><12>/23-398A9/
(1)     022745     015  031012  026463          .ASCIZ  <15><12>/23-399A9/
(1)                             .EVEN
(3)

                                ;*************************** TEST 2 ****************************
                                ;*THIS IS A SPECIAL TEST WHICH WILL RUN ON A KMC (DMC WITH
                                ;*WRITTABLE CONTROL STORE) TO LOAD THE CRAM WITH THE DDCMP
                                ;*MICRO-CODE. FIRST BE SURE BIT1 OF STAT3 IS SET UP AS FOLLOWS
                                ;*1=LOCAL HIGH SPEED CODE,  0=REMOTE LOW SPEED CODE  THE STATUS
                                ;*OF STAT3 BIT1 DETERMINES WHICH MICRO-CODE WILL
                                ;*BE LOADED IN THE KMC. LOOP ON THIS TEST FOR A FEW SECONDS
                                ;*TO LOAD THE KMC.
                                ;:***************************************************************

                                ;  TEST 2
                                ;--------------
022760  012737  000002  001226  TST2:   MOV     #2,TSTNO
022766  012737  023052  001216          MOV     #TST3,NEXT
                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
022774  004737  027012                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
023000  032737  100000  001366          BIT     #BIT15,STAT1    ;BE SURE DMC HAS CRAM
023006  001420                          BEQ     2$              ;SKIP IF NO CRAM
023010  005000                          CLR     R0              ;R0=CRAM ADDRESS
023012  013702  012320                  MOV     ROMMAP,R2       ;R2 POINTS TO ROMMAP
023016  012711  002000          1$:     MOV     #BIT10,(R1)     ;SET ROM0
```

```
1717  023022  010061  000004                    MOV     R0,4(R1)        ;LOAD CRAM ADDRESS
1718  023026  012261  000006                    MOV     (R2)+,6(R1)     ;LOAD WORD TO BE WRITTEN
1719  023032  052711  020000                    BIS     #BIT13,(R1)     ;WRITE IT!
1720  023036  005200                            INC     R0              ;NEXT ADDRESS
1721  023040  022700  002000                    CMP     #2000,R0        ;DONE YET?
1722  023044  001364                            BNE     1$              ;BR IF NO
1723  023046  005011                            CLR     (R1)            ;CLEAR SELO
1724  023050  104400                    2$:     SCOPE                   ;SCOPE THIS TEST
1725
1726
1727                                            ;*************************** TEST 3 ****************************
1728                                            ;*TEST OF BR RIGHT SHIFT
1729                                            ;*VERIFY THAT A DEST OF BR RSH (011) OF A MICRO-INSTRUCTION
1730                                            ;*SHIFTS THE RESULTING BR DATA RIGHT ONCE.
1731                                            ;:***********************************************************
1732
1733                                            ;  TEST 3
1734                                            ;--------------
1735  023052  012737  000003  001226    TST3:   MOV     #3,TSTNO
1736  023060  012737  023166  001216            MOV     #TST4,NEXT
1737                                                                    ;R1 CONTAINS BASE DMC11 ADDRESS
1738  023066  104412                            MSTCLR                  ;MASTER CLEAR DMC11
1739  023070  104412                            MSTCLR                  ;MASTER CLEAR DMC11
1740  023072  013701  001404                    MOV     DMCSR,R1        ;R1 = DMC BASE ADDRESS
1741  023076  005011                            CLR     (R1)            ;CLEAR SELO
1742  023100  012705  052525                    MOV     #52525,R5       ;START WITH 125
1743  023104  010561  000004                    MOV     R5,4(R1)        ;PORT4←125
1744  023110  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1745  023112  120500                            120500                  ;BR ← PORT4
1746  023114  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1747  023116  061620                            061620                  ;BR RSH←BR, SHIFT BR RIGHT
1748  023120  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1749  023122  061225                            061225                  ;PORT5←BR
1750  023124  006005                            ROR     R5              ;R5 = "EXPECTED"
1751  023126  116104  000005                    MOVB    5(R1),R4        ;R4 = "FOUND"
1752  023132  120504                            CMPB    R5,R4           ;DID BR SHIFT RIGHT ONCE?
1753  023134  001401                            BEQ     1$              ;BR IF YES
1754  023136  104012                            HLT     12              ;BR RIGHT SHIFT ERROR
1755  023140                             1$:
1756  023140  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1757  023142  061620                            061620                  ;BR RSH←BR, SHFT BR RIGHT AGAIN
1758  023144  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1759  023146  061225                            061225                  ;PORT5←BR
1760  023150  006005                            ROR     R5              ;R5 = "EXPECTED"
1761  023152  116104  000005                    MOVB    5(R1),R4        ;R4 = "FOUND"
1762  023156  120504                            CMPB    R5,R4           ;DID BR SHIFT RIGHT?
1763  023160  001401                            BEQ     2$              ;BR IF YES
1764  023162  104012                            HLT     12              ;BR RIGHT SHIFT ERROR
1765  023164  104400                    2$:     SCOPE                   ;SCOPE THIS TEST
1766
1767
1768                                            ;*************************** TEST 4 ****************************
1769                                            ;*CROM READ TEST
1770                                            ;*THIS TEST READS EACH ROM LOCATION AND COMPARES
1771                                            ;*IT TO A SOFTWARE DUPLICATE OF THE CROM. THIS TEST
1772                                            ;*ALSO TESTS THE JUMP(I) MICRO-PROCESSOR INSTRUCTION.
```

```
1773                                    ;*IF THIS TEST FAILS CHECK YOUR CROM PART NUMBERS.
1774                                    ;*DZDMG-C SUPPORTS THE FOLLOWING PART NUMBERS:
1775                                    ;*
1776                                    ;*DMC11-AR  (M8200-YA)
1777                                    ;*      23-414A9
1778                                    ;*      23-415A9
1779                                    ;*      23-416A9
1780                                    ;*      23-417A9
1781                                    ;*      23-418A9
1782                                    ;*      23-419A9
1783                                    ;*      23-420A9
1784                                    ;*      23-421A9
1785                                    ;*
1786                                    ;*DMC11-AL  (M8200-YB)
1787                                    ;*      23-392A9
1788                                    ;*      23-393A9
1789                                    ;*      23-394A9
1790                                    ;*      23-395A9
1791                                    ;*      23-396A9
1792                                    ;*      23-397A9
1793                                    ;*      23-398A9
1794                                    ;*      23-399A9
1795                                    ;:*****************************************************************
1796
1797                                    ;  TEST 4
1798                                    ;--------------
1799  023166  012737  000004  001226    TST4:   MOV     #4,TSTNO
1800  023174  012737  023362  001216            MOV     #TST5,NEXT
1801  023202  012737  023240  001220            MOV     #1$,LOCK
1802                                                                    ;R1 CONTAINS BASE DMC11 ADDRESS
1803  023210  104412                           MSTCLR                  ;MASTER CLEAR DMC11
1804  023212  032737  100000  001366            BIT     #BIT15,STAT1    ;IS IT RAM OR ROM
1805  023220  001057                           BNE     4$              ;SKIP TEST IF CRAM
1806  023222  004737  027012                   JSR     PC,MAPCK        ;CHECK FOR HI OR LO
1807  023226  005011                           CLR     (R1)            ;CLEAR RUN
1808  023230  013700  012320                   MOV     ROMMAP,R0       ;R0 POINTS TO SOFTWARE ROM MAP
1809  023234  005002                           CLR     R2              ;R2 CONTAINS ROM ADDRESS BITS 0-7
1810  023236  005003                           CLR     R3              ;R3 CONTAINS ROM ADDRESS BITS 8&9 IN BITS 11&12
1811  023240  042737  014377  023260    1$:     BIC     #14377,2$       ;CLEAR ADDRESS FIELDS OF INSTRUCTION
1812  023246  050237  023260                   BIS     R2,2$           ;ADD BITS 0-7 TO INSTRUCTION
1813  023252  050337  023260                   BIS     R3,2$           ;ADD BITS 11&12 TO INSTRUCTION
1814  023256  104414                           ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1815  023260  100400                    2$:     100400                  ;JUMP(I) TO ROM ADDRESS IN R2 & R3
1816  023262  012711  002000                   MOV     #BIT10,(R1)     ;SET ROM0
1817  023266  011005                           MOV     (R0),R5         ;PUT "EXPECTED" IN R5
1818  023270  016104  000006                   MOV     6(R1),R4        ;PUT "FOUND" IN R4
1819  023274  020504                           CMP     R5,R4           ;COMPARE ROM CONTENTS TO SOFT DUP
1820  023276  001414                           BEQ     3$              ;BR IF OK
1821  023300  010337  001252                   MOV     R3,TEMP3        ;PUT ROM ADDRESS IN TEMP3
1822  023304  000241                           CLC                     ;FOR ERROR TYPEOUT
1823  023306  006037  001252                   ROR     TEMP3
1824  023312  006037  001252                   ROR     TEMP3
1825  023316  006037  001252                   ROR     TEMP3
1826  023322  050237  001252                   BIS     R2,TEMP3        ;TEMP3 NOW CONTAINS CORRECT ADDRESS
1827  023326  104004                           HLT     4               ;ROM READ ERROR
1828  023330  104401                    3$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
```

```
1829  023332  005720                          TST     (R0)+           ;BUMP SOFT POINTER
1830  023334  005202                          INC     R2              ;BUMP ROM ADDRESS
1831  023336  022702  000400                  CMP     #400,R2         ;IS R2 TO MAX YET?
1832  023342  001336                          BNE     1$              ;BR IF NO
1833  023344  005002                          CLR     R2              ;YES, RESET R2 TO 0
1834  023346  062703  004000                  ADD     #4000,R3        ;INC  TO NEXT PAGE OF ROM
1835  023352  022703  020000                  CMP     #20000,R3       ;DONE YET?
1836  023356  001330                          BNE     1$              ;BR IF NO
1837  023360  104400                  4$:     SCOPE                   ;SCOPE THIS TEST
1838
1839
1840                                          ;*************************** TEST 5 ****************************
1841                                          ;*CROM TEST OF JUMP(I) NEVER MICRO-PROCESSOR INSTRUCTION.
1842                                          ;*PERFORM THE JUMP INSTRUCTION
1843                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
1844                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
1845                                          ;:***************************************************************
1846
1847                                          ;  TEST 5
1848                                          ;--------------
1849  023362  012737  000005  001226  TST5:   MOV     #5,TSTNO
1850  023370  012737  023556  001216          MOV     #TST6,NEXT
1851  023376  012737  023422  001220          MOV     #1$,LOCK
1852                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
1853  023404  104412                          MSTCLR                  ;MASTER CLEAR DMC11
1854  023406  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
1855  023414  001057                          BNE     6$+2            ;SKIP TEST IF YES
1856  023416  004737  027012                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
1857  023422                          1$:
1858  023422  004737  026656                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
1859  023426  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1860  023430  100400                          100400                  ;START AT ROM PC=0
1861  023432  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1862  023434  114377                          114377!<400*0>          ;JUMP TO ROM PC OF 1777
1863  023436  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1864  023442  000002                          2                       ;INDEX
1865  023444  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1866  023446  001401                          BEQ     2$              ;BR IF YES
1867  023450  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
1868  023452  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
1869  023454  012737  023462  001220          MOV     #3$,LOCK        ;NEW SCOP1
1870  023462                          3$:
1871  023462  004737  026656                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
1872  023466  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1873  023470  100403                          100403                  ;START AT ROM PC=3
1874  023472  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1875  023474  100000                          100000!<400*0>  ;JUMP TO ROM PC OF 0
1876  023476  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1877  023502  000010                          10                      ;INDEX
1878  023504  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1879  023506  001401                          BEQ     4$              ;BR IF YES
1880  023510  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
1881  023512  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
1882  023514  012737  023522  001220          MOV     #5$,LOCK        ;NEW SCOP1
1883  023522                          5$:
1884  023522  004737  026656                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
```

```
1885  023526  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1886  023530  100406                          100406                  ;START AT ROM PC=6
1887  023532  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1888  023534  104125                          104125!<400*0>          ;JUMP TO ROM PC OF 525
1889  023536  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1890  023542  000016                          16                      ;INDEX
1891  023544  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
1892  023546  001401                          BEQ     6$              ;BR IF YES
1893  023550  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
1894  023552  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
1895  023554  104400                          SCOPE                   ;SCOPE THIS TEST
1896
1897
1898                                          ;*************************** TEST 6 ****************************
1899                                          ;*CROM TEST OF JUMP(I) ALWAYS MICRO-PROCESSOR INSTRUCTION.
1900                                          ;*PERFORM THE JUMP INSTRUCTION
1901                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
1902                                          ;:***********************************************************
1903
1904                                          ;  TEST 6
1905                                          ;--------------
1906  023556  012737  000006  001226  TST6:   MOV     #6,TSTNO
1907  023564  012737  023736  001216          MOV     #TST7,NEXT
1908  023572  012737  023616  001220          MOV     #1$,LOCK
1909                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
1910  023600  104412                          MSTCLR                  ;MASTER CLEAR DMC11
1911  023602  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
1912  023610  001051                          BNE     6$+2            ;SKIP TEST IF YES
1913  023612  004737  027012                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
1914  023616                          1$:
1915  023616  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1916  023620  100400                          100400                  ;START AT ROM PC=0
1917  023622  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1918  023624  114777                          114377!<400*1>          ;JUMP TO ROM PC OF 1777
1919  023626  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1920  023632  003776                          3776                    ;INDEX
1921  023634  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1922  023636  001401                          BEQ     2$              ;BR IF YES
1923  023640  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
1924  023642  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
1925  023644  012737  023652  001220          MOV     #3$,LOCK        ;NEW SCOP1
1926  023652                          3$:
1927  023652  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1928  023654  100403                          100403                  ;START AT ROM PC=3
1929  023656  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1930  023660  100400                          100000!<400*1>  ;JUMP TO ROM PC OF 0
1931  023662  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1932  023666  000000                          0                       ;INDEX
1933  023670  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1934  023672  001401                          BEQ     4$              ;BR IF YES
1935  023674  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
1936  023676  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
1937  023700  012737  023706  001220          MOV     #5$,LOCK        ;NEW SCOP1
1938  023706                          5$:
1939  023706  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1940  023710  100406                          100406                  ;START AT ROM PC=6
```

```
 1941  023712  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1942  023714  104525                          104125!<400*1>          ;JUMP TO ROM PC OF 525
 1943  023716  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
 1944  023722  001252                          1252                    ;INDEX
 1945  023724  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
 1946  023726  001401                          BEQ     6$              ;BR IF YES
 1947  023730  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
 1948  023732  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
 1949  023734  104400                          SCOPE                   ;SCOPE THIS TEST
 1950
 1951
 1952                                          ;*************************** TEST 7 ****************************
 1953                                          ;*CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
 1954                                          ;*SET THE C BIT, PERFORM THE JUMP INSTRUCTION,
 1955                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
 1956                                          ;;***********************************************************************
 1957
 1958                                          ;  TEST 7
 1959                                          ;--------------
 1960  023736  012737  000007  001226  TST7:   MOV     #7,TSTNO
 1961  023744  012737  024132  001216          MOV     #TST10,NEXT
 1962  023752  012737  023776  001220          MOV     #1$,LOCK
 1963                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
 1964  023760  104412                          MSTCLR                  ;MASTER CLEAR DMC11
 1965  023762  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
 1966  023770  001057                          BNE     6$+2            ;SKIP TEST IF YES
 1967  023772  004737  027012                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
 1968  023776                          1$:
 1969  023776  004737  026724                  JSR     PC,SETC :SET THE C BIT'
 1970  024002  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1971  024004  100400                          100400                  ;START AT ROM PC=0
 1972  024006  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1973  024010  115377                          114377!<400*2>          ;JUMP TO ROM PC OF 1777
 1974  024012  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
 1975  024016  003776                          3776                    ;INDEX
 1976  024020  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
 1977  024022  001401                          BEQ     2$              ;BR IF YES
 1978  024024  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
 1979  024026  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
 1980  024030  012737  024036  001220          MOV     #3$,LOCK        ;NEW SCOP1
 1981  024036                          3$:
 1982  024036  004737  026724                  JSR     PC,SETC :SET THE C BIT'
 1983  024042  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1984  024044  100403                          100403                  ;START AT ROM PC=3
 1985  024046  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1986  024050  101000                          100000!<400*2>  ;JUMP TO ROM PC OF 0
 1987  024052  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
 1988  024056  000000                          0                       ;INDEX
 1989  024060  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
 1990  024062  001401                          BEQ     4$              ;BR IF YES
 1991  024064  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
 1992  024066  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
 1993  024070  012737  024076  001220          MOV     #5$,LOCK        ;NEW SCOP1
 1994  024076                          5$:
 1995  024076  004737  026724                  JSR     PC,SETC :SET THE C BIT'
 1996  024102  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
1997  024104  100406                          100406                  ;START AT ROM PC=6
1998  024106  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1999  024110  105125                          104125!<400*2>          ;JUMP TO ROM PC OF 525
2000  024112  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2001  024116  001252                          1252                    ;INDEX
2002  024120  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2003  024122  001401                          BEQ     6$              ;BR IF YES
2004  024124  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2005  024126  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2006  024130  104400                          SCOPE                   ;SCOPE THIS TEST
2007
2008                                          ;************************** TEST 10 **************************
2009                                          ;*CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
2010                                          ;*SET THE Z BIT, PERFORM THE JUMP INSTRUCTION,
2011                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2012                                          ;:**********************************************************************
2013
2014                                          ;  TEST 10
2015                                          ;--------------
2016
2017  024132  012737  000010  001226  TST10:  MOV     #10,TSTNO
2018  024140  012737  024326  001216          MOV     #TST11,NEXT
2019  024146  012737  024172  001220          MOV     #1$,LOCK
2020                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
2021  024154  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2022  024156  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2023  024164  001057                          BNE     6$+2            ;SKIP TEST IF YES
2024  024166  004737  027012                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2025  024172                          1$:
2026  024172  004737  026742                  JSR     PC,SETZ ;SET THE Z BIT'
2027  024176  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2028  024200  100400                          100400                  ;START AT ROM PC=0
2029  024202  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2030  024204  115777                          114377!<400*3>          ;JUMP TO ROM PC OF 1777
2031  024206  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2032  024212  003776                          3776                    ;INDEX
2033  024214  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2034  024216  001401                          BEQ     2$              ;BR IF YES
2035  024220  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2036  024222  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2037  024224  012737  024232  001220          MOV     #3$,LOCK        ;NEW SCOP1
2038  024232                          3$:
2039  024232  004737  026742                  JSR     PC,SETZ ;SET THE Z BIT'
2040  024236  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2041  024240  100403                          100403                  ;START AT ROM PC=3
2042  024242  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2043  024244  101400                          100000!<400*3>  ;JUMP TO ROM PC OF 0
2044  024246  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2045  024252  000000                          0                       ;INDEX
2046  024254  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2047  024256  001401                          BEQ     4$              ;BR IF YES
2048  024260  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2049  024262  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2050  024264  012737  024272  001220          MOV     #5$,LOCK        ;NEW SCOP1
2051  024272                          5$:
2052  024272  004737  026742                  JSR     PC,SETZ ;SET THE Z BIT'
```

```
2053  024276  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2054  024300  100406                          100406                  ;START AT ROM PC=6
2055  024302  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2056  024304  105525                          104125!<400*3>          ;JUMP TO ROM PC OF 525
2057  024306  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2058  024312  001252                          1252                    ;INDEX
2059  024314  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2060  024316  001401                          BEQ     6$              ;BR IF YES
2061  024320  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2062  024322  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2063  024324  104400                          SCOPE                   ;SCOPE THIS TEST
2064
2065
2066                                          ;*************************** TEST 11 ****************************
2067                                          ;*CROM TEST OF JUMP(I) ON BRO SET MICRO-PROCESSOR INSTRUCTION.
2068                                          ;*SET THE BRO BIT, PERFORM THE JUMP INSTRUCTION,
2069                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2070                                          ;:************************************************************
2071
2072                                          ;  TEST 11
2073                                          ;--------------
2074  024326  012737  000011  001226  TST11:  MOV     #11,TSTNO
2075  024334  012737  024522  001216          MOV     #TST12,NEXT
2076  024342  012737  024366  001220          MOV     #1$,LOCK
2077                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
2078  024350  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2079  024352  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2080  024360  001057                          BNE     6$+2            ;SKIP TEST IF YES
2081  024362  004737  027012                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2082  024366                          1$:
2083  024366  004737  026674                  JSR     PC,SETBRO       ;SET THE BRO BIT'
2084  024372  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2085  024374  100400                          100400                  ;START AT ROM PC=0
2086  024376  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2087  024400  116377                          114377!<400*4>          ;JUMP TO ROM PC OF 1777
2088  024402  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2089  024406  003776                          3776                    ;INDEX
2090  024410  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2091  024412  001401                          BEQ     2$              ;BR IF YES
2092  024414  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2093  024416  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2094  024420  012737  024426  001220          MOV     #3$,LOCK        ;NEW SCOP1
2095  024426                          3$:
2096  024426  004737  026674                  JSR     PC,SETBRO       ;SET THE BRO BIT'
2097  024432  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2098  024434  100403                          100403                  ;START AT ROM PC=3
2099  024436  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2100  024440  102000                          100000!<400*4>  ;JUMP TO ROM PC OF 0
2101  024442  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2102  024446  000000                          0                       ;INDEX
2103  024450  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2104  024452  001401                          BEQ     4$              ;BR IF YES
2105  024454  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2106  024456  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2107  024460  012737  024466  001220          MOV     #5$,LOCK        ;NEW SCOP1
2108  024466                          5$:
```

```
2109  024466  004737  026674                        JSR     PC,SETBRO       ;SET THE BRO BIT'
2110  024472  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2111  024474  100406                                100406                  ;START AT ROM PC=6
2112  024476  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2113  024500  106125                                104125!<400*4>          ;JUMP TO ROM PC OF 525
2114  024502  004737  026750                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2115  024506  001252                                1252                    ;INDEX
2116  024510  020504                                CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2117  024512  001401                                BEQ     6$              ;BR IF YES
2118  024514  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
2119  024516  104401                        6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2120  024520  104400                                SCOPE                   ;SCOPE THIS TEST
2121
2122
2123                                                ;*************************** TEST 12 ****************************
2124                                                ;*CROM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION.
2125                                                ;*SET THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
2126                                                ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2127                                                ;:************************************************************
2128
2129                                                ;  TEST 12
2130                                                ;--------------
2131  024522  012737  000012  001226        TST12:  MOV     #12,TSTNO
2132  024530  012737  024716  001216                MOV     #TST13,NEXT
2133  024536  012737  024562  001220                MOV     #1$,LOCK
2134                                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
2135  024544  104412                                MSTCLR                  ;MASTER CLEAR DMC11
2136  024546  032737  100000  001366                BIT     #BIT15,STAT1    ;IS IT CRAM?
2137  024554  001057                                BNE     6$+2            ;SKIP TEST IF YES
2138  024556  004737  027012                        JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2139  024562                                1$:
2140  024562  004737  026702                        JSR     PC,SETBR1       ;SET THE BR1 BIT'
2141  024566  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2142  024570  100400                                100400                  ;START AT ROM PC=0
2143  024572  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2144  024574  116777                                114377!<400*5>          ;JUMP TO ROM PC OF 1777
2145  024576  004737  026750                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2146  024602  003776                                3776                    ;INDEX
2147  024604  020504                                CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2148  024606  001401                                BEQ     2$              ;BR IF YES
2149  024610  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
2150  024612  104401                        2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2151  024614  012737  024622  001220                MOV     #3$,LOCK        ;NEW SCOP1
2152  024622                                3$:
2153  024622  004737  026702                        JSR     PC,SETBR1       ;SET THE BR1 BIT'
2154  024626  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2155  024630  100403                                100403                  ;START AT ROM PC=3
2156  024632  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2157  024634  102400                                102000!<400*5>  ;JUMP TO ROM PC OF 0
2158  024636  004737  026750                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2159  024642  000000                                0                       ;INDEX
2160  024644  020504                                CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2161  024646  001401                                BEQ     4$              ;BR IF YES
2162  024650  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
2163  024652  104401                        4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2164  024654  012737  024662  001220                MOV     #5$,LOCK        ;NEW SCOP1
```

```
2165  024662
2166  024662  004737  026702          5$:     JSR     PC,SETBR1       ;SET THE BR1 BIT'
2167  024666  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2168  024670  100406                          100406                  ;START AT ROM PC=6
2169  024672  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2170  024674  106525                          104125!<400*5>          ;JUMP TO ROM PC OF 525
2171  024676  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2172  024702  001252                          1252                    ;INDEX
2173  024704  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2174  024706  001401                          BEQ     6$              ;BR IF YES
2175  024710  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2176  024712  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2177  024714  104400                          SCOPE                   ;SCOPE THIS TEST
2178
2179
2180                                          ;*************************** TEST 13 ****************************
2181                                          ;*CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION.
2182                                          ;*SET THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
2183                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2184                                          ;:************************************************************
2185
2186                                          ;  TEST 13
2187                                          ;--------------
2188  024716  012737  000013  001226  TST13:  MOV     #13,TSTNO
2189  024724  012737  025112  001216          MOV     #TST14,NEXT
2190  024732  012737  024756  001220          MOV     #1$,LOCK
2191                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
2192  024740  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2193  024742  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2194  024750  001057                          BNE     6$+2            ;SKIP TEST IF YES
2195  024752  004737  027012                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2196  024756                          1$:
2197  024756  004737  026710                  JSR     PC,SETBR4       ;SET THE BR4 BIT'
2198  024762  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2199  024764  100400                          100400                  ;START AT ROM PC=0
2200  024766  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2201  024770  117377                          114377!<400*6>          ;JUMP TO ROM PC OF 1777
2202  024772  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2203  024776  003776                          3776                    ;INDEX
2204  025000  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2205  025002  001401                          BEQ     2$              ;BR IF YES
2206  025004  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2207  025006  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2208  025010  012737  025016  001220          MOV     #3$,LOCK        ;NEW SCOP1
2209  025016                          3$:
2210  025016  004737  026710                  JSR     PC,SETBR4       ;SET THE BR4 BIT'
2211  025022  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2212  025024  100403                          100403                  ;START AT ROM PC=3
2213  025026  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2214  025030  103000                          100000!<400*6>  ;JUMP TO ROM PC OF 0
2215  025032  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2216  025036  000000                          0                       ;INDEX
2217  025040  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2218  025042  001401                          BEQ     4$              ;BR IF YES
2219  025044  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2220  025046  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
```

```
2221  025050  012737  025056  001220          MOV     #5$,LOCK        ;NEW SCOP1
2222  025056                          5$:
2223  025056  004737  026710                  JSR     PC,SETBR4       ;SET THE BR4 BIT'
2224  025062  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2225  025064  100406                          100406                  ;START AT ROM PC=6
2226  025066  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2227  025070  107125                          104125!<400*6>          ;JUMP TO ROM PC OF 525
2228  025072  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2229  025076  001252                          1252                    ;INDEX
2230  025100  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2231  025102  001401                          BEQ     6$              ;BR IF YES
2232  025104  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2233  025106  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW09=1
2234  025110  104400                          SCOPE                   ;SCOPE THIS TEST
2235
2236
2237                                          ;*************************** TEST 14 ****************************
2238                                          ;*CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION.
2239                                          ;*SET THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
2240                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2241                                          ;:************************************************************
2242
2243                                          ;  TEST 14
2244                                          ;--------------
2245  025112  012737  000014  001226  TST14:  MOV     #14,TSTNO
2246  025120  012737  025306  001216          MOV     #TST15,NEXT
2247  025126  012737  025152  001220          MOV     #1$,LOCK
2248                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
2249  025134  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2250  025136  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2251  025144  001057                          BNE     6$+2            ;SKIP TEST IF YES
2252  025146  004737  027012                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2253  025152                          1$:
2254  025152  004737  026716                  JSR     PC,SETBR7       ;SET THE BR7 BIT'
2255  025156  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2256  025160  100400                          100400                  ;START AT ROM PC=0
2257  025162  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2258  025164  117777                          114377!<400*7>          ;JUMP TO ROM PC OF 1777
2259  025166  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2260  025172  003776                          3776                    ;INDEX
2261  025174  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2262  025176  001401                          BEQ     2$              ;BR IF YES
2263  025200  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2264  025202  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2265  025204  012737  025212  001220          MOV     #3$,LOCK        ;NEW SCOP1
2266  025212                          3$:
2267  025212  004737  026716                  JSR     PC,SETBR7       ;SET THE BR7 BIT'
2268  025216  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2269  025220  100403                          100403                  ;START AT ROM PC=3
2270  025222  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2271  025224  103400                          100000!<400*7>  ;JUMP TO ROM PC OF 0
2272  025226  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2273  025232  000000                          0                       ;INDEX
2274  025234  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2275  025236  001401                          BEQ     4$              ;BR IF YES
2276  025240  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
```

```
2277  025242  104401                    4$:   SCOP1                ;LOOP TO 3$ IF SW09=1
2278  025244  012737  025252  001220          MOV     #5$,LOCK      ;NEW SCOPE
2279  025252                            5$:
2280  025252  004737  026716                  JSR     PC,SETBR7     ;SET THE BR7 BIT'
2281  025256  104414                          ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2282  025260  100406                          100406                ;START AT ROM PC=6
2283  025262  104414                          ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2284  025264  107525                          104125!<400*7>        ;JUMP TO ROM PC OF 525
2285  025266  004737  026750                  JSR     PC,ROMDAT     ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2286  025272  001252                          1252                  ;INDEX
2287  025274  020504                          CMP     R5,R4         ;ARE NEW ROM PC CONTENTS CORRECT?
2288  025276  001401                          BEQ     6$            ;BR IF YES
2289  025300  104006                          HLT     6             ;ERROR, CROM PC IS WRONG
2290  025302  104401                    6$:   SCOP1                 ;LOOP TO 5$ IF SW59=1
2291  025304  104400                          SCOPE                 ;SCOPE THIS TEST
2292
2293
2294                                          ;************************** TEST 15 **************************
2295                                          ;*CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
2296                                          ;*CLEAR THE C BIT, PERFORM THE JUMP INSTRUCTION,
2297                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2298                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2299                                          ;;*************************************************************
2300
2301                                          ;  TEST 15
2302                                          ;--------------
2303  025306  012737  000015  001226  TST15:  MOV     #15,TSTNO
2304  025314  012737  025502  001216          MOV     #TST16,NEXT
2305  025322  012737  025346  001220          MOV     #1$,LOCK
2306                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
2307  025330  104412                          MSTCLR                ;MASTER CLEAR DMC11
2308  025332  032737  100000  001366          BIT     #BIT15,STAT1  ;IS IT CRAM?
2309  025340  001057                          BNE     6$+2          ;SKIP TEST IF YES
2310  025342  004737  027012                  JSR     PC,MAPCK      ;CHECK FOR HI OR LO
2311  025346                            1$:
2312  025346  004737  026656                  JSR     PC,CLRALL     ;CLEAR ALL  CONDITIONS
2313  025352  104414                          ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2314  025354  100400                          100400                ;START AT ROM PC=0
2315  025356  104414                          ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2316  025360  115377                          114377!<400*2>        ;JUMP TO ROM PC OF 1777
2317  025362  004737  026750                  JSR     PC,ROMDAT     ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2318  025366  000002                          2                     ;INDEX
2319  025370  020504                          CMP     R5,R4         ;ARE NEW PC CONTENTS CORRECT?
2320  025372  001401                          BEQ     2$            ;BR IF YES
2321  025374  104006                          HLT     6             ;ERROR, CROM PC IS WRONG
2322  025376  104401                    2$:   SCOP1                 ;LOOP TO 1$ IF SW09=1
2323  025400  012737  025406  001220          MOV     #3$,LOCK      ;NEW SCOPE
2324  025406                            3$:
2325  025406  004737  026656                  JSR     PC,CLRALL     ;CLEAR ALL  CONDITIONS
2326  025412  104414                          ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2327  025414  100403                          100403                ;START AT ROM PC=3
2328  025416  104414                          ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2329  025420  101000                          100000!<400*2>  ;JUMP TO ROM PC OF 0
2330  025422  004737  026750                  JSR     PC,ROMDAT     ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2331  025426  000010                          10                    ;INDEX
2332  025430  020504                          CMP     R5,R4         ;ARE NEW PC CONTENTS CORRECT?
```

```
2333  025432  001401                          BEQ     4$              ;BR IF YES
2334  025434  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2335  025436  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2336  025440  012737  025446  001220          MOV     #5$,LOCK        ;NEW SCOP1
2337  025446                          5$:
2338  025446  004737  026656                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2339  025452  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2340  025454  100406                          100406                  ;START AT ROM PC=6
2341  025456  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2342  025460  105125                          104125!<400*2>          ;JUMP TO ROM PC OF 525
2343  025462  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2344  025466  000016                          16                      ;INDEX
2345  025470  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2346  025472  001401                          BEQ     6$              ;BR IF YES
2347  025474  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2348  025476  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2349  025500  104400                          SCOPE                   ;SCOPE THIS TEST
2350
2351
2352                                          ;*************************** TEST 16 ****************************
2353                                          ;*CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
2354                                          ;*CLEAR THE Z BIT, PERFORM THE JUMP INSTRUCTION,
2355                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2356                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2357                                          ;:************************************************************
2358
2359                                          ;  TEST 16
2360                                          ;---------------
2361  025502  012737  000016  001226  TST16:  MOV     #16,TSTNO
2362  025510  012737  025676  001216          MOV     #TST17,NEXT
2363  025516  012737  025542  001220          MOV     #1$,LOCK
2364                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
2365  025524  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2366  025526  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2367  025534  001057                          BNE     6$+2            ;SKIP TEST IF YES
2368  025536  004737  027012                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2369  025542                          1$:
2370  025542  004737  026656                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2371  025546  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2372  025550  100400                          100400                  ;START AT ROM PC=0
2373  025552  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2374  025554  115777                          114377!<400*3>          ;JUMP TO ROM PC OF 1777
2375  025556  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2376  025562  000002                          2                       ;INDEX
2377  025564  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2378  025566  001401                          BEQ     2$              ;BR IF YES
2379  025570  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2380  025572  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2381  025574  012737  025602  001220          MOV     #3$,LOCK        ;NEW SCOP1
2382  025602                          3$:
2383  025602  004737  026656                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2384  025606  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2385  025610  100403                          100403                  ;START AT ROM PC=3
2386  025612  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2387  025614  101400                          100000!<400*3>  ;JUMP TO ROM PC OF 0
2388  025616  004737  026750                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
```

```
2389  025622  000010                                    10                      ;INDEX
2390  025624  020504                                    CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2391  025626  001401                                    BEQ     4$              ;BR IF YES
2392  025630  104006                                    HLT     6               ;ERROR, CROM PC IS WRONG
2393  025632  104401                              4$:   SCOP1                   ;LOOP TO 3$ IF SW09=1
2394  025634  012737  025642  001220                    MOV     #5$,LOCK        ;NEW SCOP1
2395  025642                                      5$:
2396  025642  004737  026656                            JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2397  025646  104414                                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2398  025650  100406                                    100406                  ;START AT ROM PC=6
2399  025652  104414                                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2400  025654  105525                                    104125!<400*3>          ;JUMP TO ROM PC OF 525
2401  025656  004737  026750                            JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2402  025662  000016                                    16                      ;INDEX
2403  025664  020504                                    CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2404  025666  001401                                    BEQ     6$              ;BR IF YES
2405  025670  104006                                    HLT     6               ;ERROR, CROM PC IS WRONG
2406  025672  104401                              6$:   SCOP1                   ;LOOP TO 5$ IF SW59=1
2407  025674  104400                                    SCOPE                   ;SCOPE THIS TEST
2408
2409
2410                                                    ;************************** TEST 17 **************************
2411                                                    ;*CROM TEST OF JUMP(I) ON BRO SET MICRO-PROCESSOR INSTRUCTION.
2412                                                    ;*CLEAR THE BRO BIT, PERFORM THE JUMP INSTRUCTION,
2413                                                    ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2414                                                    ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2415                                                    ;;**********************************************************
2416
2417                                                    ;  TEST 17
2418                                                    ;--------------
2419  025676  012737  000017  001226            TST17:  MOV     #17,TSTNO
2420  025704  012737  026072  001216                    MOV     #TST20,NEXT
2421  025712  012737  025736  001220                    MOV     #1$,LOCK
2422                                                                            ;R1 CONTAINS BASE DMC11 ADDRESS
2423  025720  104412                                    MSTCLR                  ;MASTER CLEAR DMC11
2424  025722  032737  100000  001366                    BIT     #BIT15,STAT1    ;IS IT CRAM?
2425  025730  001057                                    BNE     6$+2            ;SKIP TEST IF YES
2426  025732  004737  027012                            JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2427  025736                                      1$:
2428  025736  004737  026656                            JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2429  025742  104414                                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2430  025744  100400                                    100400                  ;START AT ROM PC=0
2431  025746  104414                                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2432  025750  116377                                    114377!<400*4>          ;JUMP TO ROM PC OF 1777
2433  025752  004737  026750                            JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2434  025756  000002                                    2                       ;INDEX
2435  025760  020504                                    CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2436  025762  001401                                    BEQ     2$              ;BR IF YES
2437  025764  104006                                    HLT     6               ;ERROR, CROM PC IS WRONG
2438  025766  104401                              2$:   SCOP1                   ;LOOP TO 1$ IF SW09=1
2439  025770  012737  025776  001220                    MOV     #3$,LOCK        ;NEW SCOP1
2440  025776                                      3$:
2441  025776  004737  026656                            JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2442  026002  104414                                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2443  026004  100403                                    100403                  ;START AT ROM PC=3
2444  026006  104414                                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
2445  026010  102000                        100000!<400*4>  ;JUMP TO ROM PC OF 0
2446  026012  004737  026750                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2447  026016  000010                        10                      ;INDEX
2448  026020  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2449  026022  001401                        BEQ     4$              ;BR IF YES
2450  026024  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2451  026026  104401                4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2452  026030  012737  026036  001220        MOV     #5$,LOCK        ;NEW SCOP1
2453  026036                        5$:
2454  026036  004737  026656                JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2455  026042  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2456  026044  100406                        100406                  ;START AT ROM PC=6
2457  026046  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2458  026050  106125                        104125!<400*4>          ;JUMP TO ROM PC OF 525
2459  026052  004737  026750                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2460  026056  000016                        16                      ;INDEX
2461  026060  020504                        CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2462  026062  001401                        BEQ     6$              ;BR IF YES
2463  026064  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2464  026066  104401                6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2465  026070  104400                        SCOPE                   ;SCOPE THIS TEST
2466
2467
2468                                        ;************************** TEST 20 **************************
2469                                        ;*CROM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION.
2470                                        ;*CLEAR THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
2471                                        ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2472                                        ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2473                                        ;:**********************************************************
2474
2475                                        ;  TEST 20
2476                                        ;--------------
2477  026072  012737  000020  001226  TST20: MOV     #20,TSTNO
2478  026100  012737  026266  001216        MOV     #TST21,NEXT
2479  026106  012737  026132  001220        MOV     #1$,LOCK
2480                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
2481  026114  104412                        MSTCLR                  ;MASTER CLEAR DMC11
2482  026116  032737  100000  001366        BIT     #BIT15,STAT1    ;IS IT CRAM?
2483  026124  001057                        BNE     6$+2            ;SKIP TEST IF YES
2484  026126  004737  027012                JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2485  026132                        1$:
2486  026132  004737  026656                JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2487  026136  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2488  026140  100400                        100400                  ;START AT ROM PC=0
2489  026142  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2490  026144  116777                        114377!<400*5>          ;JUMP TO ROM PC OF 1777
2491  026146  004737  026750                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2492  026152  000002                        2                       ;INDEX
2493  026154  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2494  026156  001401                        BEQ     2$              ;BR IF YES
2495  026160  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2496  026162  104401                2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2497  026164  012737  026172  001220        MOV     #3$,LOCK        ;NEW SCOP1
2498  026172                        3$:
2499  026172  004737  026656                JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2500  026176  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
2501  026200  100403                                  100403                  ;START AT ROM PC=3
2502  026202  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2503  026204  102400                                  100000!<400*5>  ;JUMP TO ROM PC OF 0
2504  026206  004737  026750                         JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2505  026212  000010                                  10                      ;INDEX
2506  026214  020504                                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2507  026216  001401                                  BEQ     4$              ;BR IF YES
2508  026220  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
2509  026222  104401                          4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2510  026224  012737  026232  001220                  MOV     #5$,LOCK        ;NEW SCOP1
2511  026232                                  5$:
2512  026232  004737  026656                         JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2513  026236  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2514  026240  100406                                  100406                  ;START AT ROM PC=6
2515  026242  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2516  026244  106525                                  104125!<400*5>          ;JUMP TO ROM PC OF 525
2517  026246  004737  026750                         JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2518  026252  000016                                  16                      ;INDEX
2519  026254  020504                                  CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2520  026256  001401                                  BEQ     6$              ;BR IF YES
2521  026260  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
2522  026262  104401                          6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2523  026264  104400                                  SCOPE                   ;SCOPE THIS TEST
2524
2525
2526                                                  ;************************** TEST 21 **************************
2527                                                  ;*CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION.
2528                                                  ;*CLEAR THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
2529                                                  ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2530                                                  ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2531                                                  ;:************************************************************
2532
2533                                                  ;  TEST 21
2534                                                  ;--------------
2535  026266  012737  000021  001226          TST21:  MOV     #21,TSTNO
2536  026274  012737  026462  001216                  MOV     #TST22,NEXT
2537  026302  012737  026326  001220                  MOV     #1$,LOCK
2538                                                                          ;R1 CONTAINS BASE DMC11 ADDRESS
2539  026310  104412                                  MSTCLR                  ;MASTER CLEAR DMC11
2540  026312  032737  100000  001366                  BIT     #BIT15,STAT1    ;IS IT CRAM?
2541  026320  001057                                  BNE     63+2            ;SKIP TEST IF YES
2542  026322  004737  027012                         JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2543  026326                                  1$:
2544  026326  004737  026656                         JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2545  026332  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2546  026334  100400                                  100400                  ;START AT ROM PC=0
2547  026336  104414                                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2548  026340  117377                                  114377!<400*6>          ;JUMP TO ROM PC OF 1777
2549  026342  004737  026750                         JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2550  026346  000002                                  2                       ;INDEX
2551  026350  020504                                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2552  026352  001401                                  BEQ     2$              ;BR IF YES
2553  026354  104006                                  HLT     6               ;ERROR, CROM PC IS WRONG
2554  026356  104401                          2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2555  026360  012737  026366  001220                  MOV     #3$,LOCK        ;NEW SCOP1
2556  026366                                  3$:
```

```
2557  026366  004737  026656                        JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2558  026372  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2559  026374  100403                                100403                  ;START AT ROM PC=3
2560  026376  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2561  026400  103000                                100000!<400*6>  ;JUMP TO ROM PC OF 0
2562  026402  004737  026750                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2563  026406  000010                                10                      ;INDEX
2564  026410  020504                                CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2565  026412  001401                                BEQ     4$              ;BR IF YES
2566  026414  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
2567  026416  104401                        4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2568  026420  012737  026426  001220                MOV     #5$,LOCK        ;NEW SCOP1
2569  026426                                5$:
2570  026426  004737  026656                        JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2571  026432  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2572  026434  100406                                100406                  ;START AT ROM PC=6
2573  026436  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2574  026440  107125                                104125!<400*6>          ;JUMP TO ROM PC OF 525
2575  026442  004737  026750                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2576  026446  000016                                16                      ;INDEX
2577  026450  020504                                CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2578  026452  001401                                BEQ     6$              ;BR IF YES
2579  026454  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
2580  026456  104401                        6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2581  026460  104400                                SCOPE                   ;SCOPE THIS TEST
2582
2583
2584                                                ;************************** TEST 22 **************************
2585                                                ;*CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION.
2586                                                ;*CLEAR THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
2587                                                ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2588                                                ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2589                                                ;:************************************************************
2590
2591                                                ;  TEST 22
2592                                                ;--------------
2593  026462  012737  000022  001226        TST22:  MOV     #22,TSTNO
2594  026470  012737  003364  001216                MOV     #.EOP,NEXT
2595  026476  012737  026522  001220                MOV     #1$,LOCK
2596                                                                        ;R1 CONTAINS BASE DMC11 ADDRESS
2597  026504  104412                                MSTCLR                  ;MASTER CLEAR DMC11
2598  026506  032737  100000  001366                BIT     #BIT15,STAT1    ;IS IT CRAM?
2599  026514  001057                                BNE     6$+2            ;SKIP TEST IF YES
2600  026516  004737  027012                        JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2601  026522                                1$:
2602  026522  004737  026656                        JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2603  026526  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2604  026530  100400                                100400                  ;START AT ROM PC=0
2605  026532  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2606  026534  117777                                114377!<400*7>          ;JUMP TO ROM PC OF 1777
2607  026536  004737  026750                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2608  026542  000002                                2                       ;INDEX
2609  026544  020504                                CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2610  026546  001401                                BEQ     2$              ;BR IF YES
2611  026550  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
2612  026552  104401                        2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
```

```
2613  026554  012737  026562  001220                    MOV     #3$,LOCK        ;NEW SCOP1
2614  026562                                    3$:
2615  026562  004737  026656                            JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2616  026566  104414                                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2617  026570  100403                                    100403                  ;START AT ROM PC=3
2618  026572  104414                                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2619  026574  103400                                    100000!<400*7>  ;JUMP TO ROM PC OF 0
2620  026576  004737  026750                            JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2621  026602  000010                                    10                      ;INDEX
2622  026604  020504                                    CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2623  026606  001401                                    BEQ     4$              ;BR IF YES
2624  026610  104006                                    HLT     6               ;ERROR, CROM PC IS WRONG
2625  026612  104401                            4$:     SCOP1                   ;LOOP TO 3$ IF SW05=1
2626  026614  012737  026622  001220                    MOV     #5$,LOCK        ;NEW SCOP1
2627  026622                                    5$:
2628  026622  004737  026656                            JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2629  026626  104414                                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2630  026630  100406                                    100406                  ;START AT ROM PC=6
2631  026632  104414                                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2632  026634  107525                                    104125!<400*7>          ;JUMP TO ROM PC OF 525
2633  026636  004737  026750                            JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2634  026642  000016                                    16                      ;INDEX
2635  026644  020504                                    CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2636  026646  001401                                    BEQ     6$              ;BR IF YES
2637  026650  104006                                    HLT     6               ;ERROR, CROM PC IS WRONG
2638  026652  104401                            6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2639  026654  104400                                    SCOPE                   ;SCOPE THIS TEST
2640                                            00300
2641                                            00400
2642                                            00500   ;SUBROUTINES
2643                                            00600   ;-----------
2644                                            00700
2645  026656                                    00800   CLRALL:
2646                                            00900           ;THIS SUBROUTINE CLEARS THE C&Z BITS AND THE BR
2647                                            01000
2648  026656  104414                                           ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2649  026660  000400                            01200           000400                  ;BR<-0
2650  026662  104414                                           ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2651  026664  063220                            01400           063220                  ;SP(0)<-BR
2652  026666  104414                                           ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2653  026670  060400                            01600           060400                  ;BR<-SP(0)+BR
2654  026672  000207                            01700           RTS     PC
2655                                            01800
2656                                            01900
2657  026674                                    02000   SETBR0:
2658                                            02100           ;THIS SUBROUTINE SETS BR0 BIT
2659                                            02200
2660  026674  104414                                           ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2661  026676  000401                            02400           000401                  ;BR<-001
2662  026700  000207                            02500           RTS     PC
2663                                            02600
2664                                            02700
2665  026702                                    02800   SETBR1:
2666                                            02900           ;THIS SUBROUTINE SETS BR1 BIT
2667                                            03000
2668  026702  104414                                           ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
2669  026704  000402                  03200                   000402                  ;BR+002
2670  026706  000207                  03300                   RTS     PC
2671                                  03400
2672                                  03500
2673  026710                          03600   SETBR4:
2674                                  03700           ;THIS SUBROUTINE SETS BR4 BIT
2675                                  03800
2676  026710  104414                                 ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2677  026712  000420                  04000           000420                  ;BR+020
2678  026714  000207                  04100           RTS     PC
2679                                  04200
2680                                  04300
2681  026716                          04400   SETBR7:
2682                                  04500           ;THIS SUBROUTINE SETS BR7 BIT
2683                                  04600
2684  026716  104414                                 ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2685  026720  000600                  04800           000600                  ;BR+200
2686  026722  000207                  04900           RTS     PC
2687                                  05000
2688                                  05100
2689  026724                          05200   SETC:
2690                                  05300           ;THIS SUBROUTINE SETS THE C BIT
2691                                  05400
2692  026724  104414                                 ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2693  026726  000777                  05600           000777                  ;BR+377
2694  026730  104414                                 ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2695  026732  063220                  05800           063220                  ;SP(0)+BR
2696  026734  104414                                 ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2697  026736  060400                  06000           060400                  ;BR+SP(0)+BR
2698  026740  000207                  06100           RTS     PC
2699                                  06200
2700                                  06300
2701  026742                          06400   SETZ:
2702                                  06500           ;THIS SUBROUTINE SETS THE Z BIT
2703                                  06600
2704  026742  104414                                 ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2705  026744  000777                  06800           000777                  ;BR+377
2706  026746  000207                  06900           RTS     PC
2707                                  07000
2708                                  07100
2709  026750                          07200   ROMDAT:
2710                                  07300           ;THIS SUBROUTINE LOADS R5 WITH EXPECTED ROM CONTENTS
2711                                  07400           ;AND LOADS R4 WITH ACTUAL ROM CONTENTS
2712                                  07500
2713  026750  017600  000000          07600           MOV     @(SP),R0        ;INDEX FOR COMPARE
2714  026754  062716  000002          07700           ADD     #2,(SP)         ;ADJUST STACK
2715  026760  012711  002000          07800           MOV     #BIT10,(R1)     ;SET ROM0
2716  026764  016005  012322          07900           MOV     LOMAP(R0),R5    ;PUT EXPECTED IN R5 (LOSPEED)
2717  026770  032737  000002  001372  08000           BIT     #BIT1,STAT3     ;LOW OR HIGH SPEED?
2718  026776  001402                  08100           BEQ     1$              ;BR IF LOW SPEED
2719  027000  016005  016322          08200           MOV     HIMAP(R0),R5    ;PUT EXPECTED IN R5 (HISPEED)
2720  027004  016104  000006          08300   1$:     MOV     6(R1),R4        ;PUT "FOUND" IN R4
2721  027010  000207                  08400           RTS     PC              ;RETURN
2722                                  08500
2723  027012                          08600   MAPCK:
2724                                  08700           ;THIS SUBROUTINE CHECKS THE STATUS TABLE AND LOADS
```

```
2725                                    08800           ;THE ROMMAP POINTER TO POINT TO EITHER THE HIGH OR
2726                                    08900           ;LOW SPEED MICRO-CODE.
2727                                    09000
2728  027012  012737  012322  012320    09100           MOV     #LOMAP,ROMMAP   ;LOAD POINTER TO LOW SPEED
2729  027020  032737  000002  001372    09200           BIT     #BIT1,STAT3     ;CHECK STATUS TABLE
2730  027026  001403                    09300           BEQ     1$              ;BR IF LOW SPEED
2731  027030  012737  016322  012320    09400           MOV     #HIMAP,ROMMAP   ;LOAD POINTER TO HIGH SPEED
2732  027036  000207                    09500   1$:     RTS     PC              ;RETURN
2733                                    09600
2734                                    00300
      027040  041777  040522  020115    00400   EM1:    .ASCIZ  <377>/CRAM DATA ERROR/
      027061     377  051103  046501    00500   EM2:    .ASCIZ  <377>/CRAM DUAL ADDRESSING ERROR/
      027115     377  051103  046517    00600   EM3:    .ASCIZ  <377>/CROM DATA ERROR/
      027136  045377  046525  020120    00700   EM4:    .ASCIZ  <377>/JUMP ERROR/
      027152  047777  052104  042440    00800   EM5:    .ASCIZ  <377>/ODT ERROR IN IBUS* REG10/
      027204  044777  050117  046440    00900   EM7:    .ASCIZ  <377>/IOP MAR TEST/
      027222  041377  020122  044522    01000   EM10:   .ASCIZ  <377>/BR RIGHT SHIFT TEST/
      027247     377  042522  042503    01100   EM11:   .ASCIZ  <377>/RECEIVE DATA ERROR/
      027273     377  051106  042505    01200   EM12:   .ASCIZ  <377>/FREE RUNNING ERROR/
      027317     377  047503  052116    01300   EM13:   .ASCIZ  <377>/CONTROL OUT ERROR/
                                        01400
      027342  042777  050130  041505    01500   DH1:    .ASCIZ  <377>/EXPECTED  FOUND  ADDRESS/
      027374  042777  050130  041505    01600   DH2:    .ASCIZ  <377>/EXPECTED  FOUND/
      027415     377  051440  046105    01700   DH3:    .ASCIZ  <377>/ SEL4      SEL6/
                                        01800           .EVEN
                                        01900
027436  000003                          02000   DT1:    3
027440     006  004                     02100           .BYTE   6,4
027442  001264                          02200           SAVR2
027444     006  004                     02300           .BYTE   6,4
027446  001270                          02400           SAVR4
027450     004  002                     02500           .BYTE   4,2
027452  001260                          02600           SAVR0
027454  000003                          02700   DT2:    3
027456     006  004                     02800           .BYTE   6,4
027460  001272                          02900           SAVR5
027462     006  004                     03000           .BYTE   6,4
027464  001270                          03100           SAVR4
027466     004  002                     03200           .BYTE   4,2
027470  001264                          03300           SAVR2
027472  000003                          03400   DT3:    3
027474     006  004                     03500           .BYTE   6,4
027476  001272                          03600           SAVR5
027500     006  004                     03700           .BYTE   6,4
027502  001270                          03800           SAVR4
027504     004  002                     03900           .BYTE   4,2
027506  001252                          04000           TEMP3
027510  000002                          04100   DT4:    2
027512     003  007                     04200           .BYTE   3,7
027514  001272                          04300           SAVR5
027516     003  002                     04400           .BYTE   3,2
027520  001270                          04500           SAVR4
027522  000002                          04600   DT5:    2
027524     006  004                     04700           .BYTE   6,4
027526  001272                          04800           SAVR5
027530     006  002                     04900           .BYTE   6,2
```

```
027532  001270                  05000           SAVR4
027534  000003                  05100   DT7:    3
027536      003     010         05200           .BYTE   3,10
027540  001272                  05300           SAVR5
027542      003     004         05400           .BYTE   3,4
027544  001270                  05500           SAVR4
027546      004     002         05600           .BYTE   4,2
027550  001264                  05700           SAVR2
027552  000003                  05800   DT10:   3
027554      003     007         05900           .BYTE   3,7
027556  001272                  06000           SAVR5
027560      003     004         06100           .BYTE   3,4
027562  001270                  06200           SAVR4
027564      006     002         06300           .BYTE   6,2
027566  001252                  06400           TEMP3
027570  000002                  06500   DT11:   2
027572      006     004         06600           .BYTE   6,4
027574  001252                  06700           TEMP3
027576      006     002         06800           .BYTE   6,2
027600  001254                  06900           TEMP4
                                07000
027602                          07100   .ERRTAB:
027602  000000                  07200           0
027604  000000                  07300           0
027606  000000                  07400           0
027610  027040                  07500   EM1
027612  027342                  07600   DH1     ;HLT    1
027614  027436                  07700   DT1
027616  027061                  07800   EM2
027620  027342                  07900   DH1     ;HLT    2
027622  027436                  08000   DT1
027624  027040                  08100   EM1
027626  027342                  08200   DH1     ;HLT    3
027630  027454                  08300   DT2
027632  027115                  08400   EM3
027634  027342                  08500   DH1     ;HLT    4
027636  027472                  08600   DT3
027640  027136                  08700   EM4
027642  027374                  08800   DH2     ;HLT    5
027644  027510                  08900   DT4
027646  027136                  09000   EM4
027650  027374                  09100   DH2     ;HLT    6
027652  027522                  09200   DT5
027654  027152                  09300   EM5
027656  027374                  09400   DH2     ;HLT    7
027660  027510                  09500   DT4
027662  000000                  09600   0
027664  000000                  09700   0
027666  000000                  09800   0
027670  027204                  09900   EM7
027672  027342                  10000   DH1     ;HLT    11
027674  027534                  10100   DT7
027676  027222                  10200   EM10
027700  027374                  10300   DH2     ;HLT    12
027702  027510                  10400   DT4
027704  027247                  10500   EM11
```

```
027706  027342
027710  027552
027712  027273
027714  000000
027716  000000
027720  027273
027722  027374
027724  027522
027726  027317
027730  027415
027732  027570

027734
        000001

10600   DH1     ;HLT    13
10700   DT10
10800   EM12
10900   0       ;HLT    14
11000   0
11100   EM12
11200   DH2     ;HLT    15
11300   DT5
11400   EM13
11500   DH3     ;HLT    16
11600   DT11
11700
11800
11900   CORMAX:
12400   .END
```

```
ADRCNT= 004373   879*   915*   924#
AUDONE  003024   569    608    613    659#
AUSTRT  002446   568#   663
AUTO.S  010512   526    1368#
BINWRD  004714   965*   968*   969    1006#
BIT0   = 000001   95#   1155   1156
BIT1   = 000002   94#   531    1149   1155   1156   1449   1532   1545   2717   2729
BIT10  = 002000   85#   1513   1524   1716   1816   2715
BIT11  = 004000   84#
BIT12  = 010000   83#   1464   1543
BIT13  = 020000   82#   1467   1516   1547   1719
BIT14  = 040000   81#   781    1478   1480   1547   1558
BIT15  = 100000   80#   485    572    575    596    599    1451   1519   1712   1804   1854   1911   1965
                 2022   2079   2136   2193   2250   2308   2366   2424   2482   2540   2598
BIT2   = 000004   93#   531    712    1156
BIT3   = 000010   92#   1549   1556
BIT4   = 000020   91#   1139   1162   1164
BIT5   = 000040   90#   1661
BIT6   = 000100   89#   1144   1145   1551
BIT7   = 000200   88#   1145   1265   1661
BIT8   = 000400   87#   1523   1538   1540   1553   1555   1561   1564   1622
BIT9   = 001000   86#   1521   1523   1535   1561   1564   1620   1622
BM      007054  1187#   1492
BRLVL   012252  1617    1627   1635   1647#
BRW     003730   718     804#
BRX     003732   719     805#
CHRCNT  004712   963*    966    970    986*   1004#  1005
CKSWR   007606   512     779    811    1026   1212#
CKSWR1  007666  1225#   1237
CKSWR2  007700  1228#
CKSWR3  007704  1230#
CKSWR4  007710  1231#   1239   1246
CKSWR5  010014  1213    1220   1255#
CLKX    001242   171#
CLRALL  026656  1858   1871   1884   2312   2325   2338   2370   2383   2396   2428   2441   2454   2486
                2499   2512   2544   2557   2570   2602   2615   2628   2645#
CNERR   007277   626    1187#
CNT.MA  001702   196     364#   484    486    488    1293
CNVRT  = 104411   233#   623    738    740    742    744    1060   1062   1122   1228
CONERR  007223   621    1187#
CONN    007114  1187#   1469
CONTAB  002776   630     643#
CONVRT= 104410   231#   543    629    1076   1391
CORMAX  027734  2734#
CRAM    006606  1187#   1430
CREAM   001320   195#   483*   1289*  1290   1292*  1296
CSR     006510  1187#   1395
CSRMAP  010514  1370#
CYCLE   010060   720     761    762    1280#
DATABP  005216  1049*   1052   1074   1077#
DATACL= 104415   241#
DATAHD  005204  1048*   1070   1073#
DELAY  = 104413   237#
DEVADR  004370   877*    912    922#
DEVTAB  003010   583     648#
DH1     027342  2734#
```

```
DM2     027374          2734#
DM3     027415          2734#
DISPLA  001200           142#   498*   504*   735*
DISPRE  000174           128#   504
DMACTV  001306           189#   521    676*   677    1280   1294   1376*  1577*  1583*  1584*  1588   1613
DMCM    007320           634   1187#
DMCR00  001500           279#
DMCR01  001510           284#
DMCR02  001520           289#
DMCR03  001530           294#
DMCR04  001540           299#
DMCR05  001550           304#
DMCR06  001560           309#
DMCR07  001570           314#
DMCR10  001600           319#
DMCR11  001610           324#
DMCR12  001620           329#
DMCR13  001630           334#
DMCR14  001640           339#
DMCR15  001650           344#
DMCR16  001660           349#
DMCR17  001670           354#
DMCSR   001404           262#   568*   602    607*   612*   647    765    802    1094   1117   1163   1298*  1307
                        1356   1740
DMCSRH  001406           263#  1144*  1145*  1149*  1155*  1156*  1162*  1164*  1307*  1308*  1309
DMCTL   001410           264#  1309*  1310*  1311
DMNUM   001310           190#   479    751   1374*  1389*  1568*  1569   1578   1580
DMP04   001412           265#  1133*  1139   1176   1181   1311*  1312*  1313
DMP06   001414           266#  1150*  1313*  1314*
DMRLVL  001376           259#  1316*  1317*  1318
DMRVEC  001374           258#   768   1299*  1300*  1316
DMS100  001502           280#
DMS101  001512           285#
DMS102  001522           290#
DMS103  001532           295#
DMS104  001542           300#
DMS105  001552           305#
DMS106  001562           310#
DMS107  001572           315#
DMS110  001602           320#
DMS111  001612           325#
DMS112  001622           330#
DMS113  001632           335#
DMS114  001642           340#
DMS115  001652           345#
DMS116  001662           350#
DMS117  001672           355#
DMS200  001504           281#
DMS201  001514           286#
DMS202  001524           291#
DMS203  001534           296#
DMS204  001544           301#
DMS205  001554           306#
DMS206  001564           311#
DMS207  001574           316#
DMS210  001604           321#
```

```
DMS211  001614          326#
DMS212  001624          331#
DMS213  001634          336#
DMS214  001644          341#
DMS215  001654          346#
DMS216  001664          351#
DMS217  001674          356#
DMS300  001506          282#
DMS301  001516          287#
DMS302  001526          292#
DMS303  001536          297#
DMS304  001546          302#
DMS305  001556          307#
DMS306  001566          312#
DMS307  001576          317#
DMS310  001606          322#
DMS311  001616          327#
DMS312  001626          332#
DMS313  001636          337#
DMS314  001646          342#
DMS315  001656          347#
DMS316  001666          352#
DMS317  001676          357#
DMTLVL  001402          261#   1320*   1321*
DMTVEC  001400          260#   1318*   1319*   1320
DM.END  001700          359#   1372
DM.MAP  001500          195     278#    483     536     546     662    1290    1292    1370    1375    1605
DONE    003734          784     786*    806#   1216*
DT1     027436         2734#
DT10    027552         2734#
DT11    027570         2734#
DT2     027454         2734#
DT3     027472         2734#
DT4     027510         2734#
DT5     027522         2734#
DT7     027534         2734#
EM1     027040         2734#
EM10    027222         2734#
EM11    027247         2734#
EM12    027273         2734#
EM13    027317         2734#
EM2     027061         2734#
EM3     027115         2734#
EM4     027136         2734#
EM5     027152         2734#
EM7     027204         2734#
ERCT00  001704          366#
ERCT01  001710          369#
ERCT02  001714          372#
ERCT03  001720          375#
ERCT04  001724          378#
ERCT05  001730          381#
ERCT06  001734          384#
ERCT07  001740          387#
ERCT10  001744          390#
ERCT11  001750          393#
```

```
ERCT12  001754  396#
ERCT13  001760  399#
ERCT14  001764  402#
ERCT15  001770  405#
ERCT16  001774  408#
ERCT17  002000  411#
ERR     002700  592  618#  622
ERRCNT  001232  163#  747  774  1087*  1305*
ERRFLG  001325  202#  481*  733*  795*  1037*  1050  1064*  1123*
ERRMSG  005172  1047*  1065  1068#
ERRPC   002770  624  640#
ERTABO  005322  1062  1097#
EXIT =  000205  96#
EXITER  005252  1082  1087#
FLOAT   002536  585#  591
FY      002566  594#  605  609  615
HALTS   005222  1033  1079#
HILIM   004366  876*  903  921#
HIMAP   016322  1670#  2719  2731
ICOUNT  001222  159#  793  798*
INBUF   007502  846  882  1202#
INCHAR  010020  1231  1259#
INIFLG  001324  201#  507  527  534*
INSTER= 104404  223#  897
INSTR = 104403  221#  1329  1381  1394  1403  1482  1491
INSTR2  004166  853  865#
INTTY   012266  1414  1431  1441  1454  1470  1655#
KMCM    007330  637  1187#
LIMITS  004314  892  903#
LINE    007016  1187#  1483
LOBITS  004372  878*  907  923#  924
LOCK    001220  158#  797*  814  816  1056  1801*  1851*  1869*  1882*  1908*  1925*  1937*  1962*
                1980*  1993*  2019*  2037*  2050*  2076*  2094*  2107*  2133*  2151*  2164*  2190*  2208*
                2221*  2247*  2265*  2278*  2305*  2323*  2336*  2363*  2381*  2394*  2421*  2439*  2452*
                2479*  2497*  2510*  2537*  2555*  2568*  2595*  2613*  2626*
LOKFLG  001326  203#
LOLIM   004364  875*  905  920#
LOMAP   012322  1667#  2716  2728
LPCNT   001224  160#  792*  793  796*
LSTERR  001234  164#  490*  732*  1034  1036*  1124*
MAPCK   027012  1711  1806  1856  1913  1967  2024  2081  2138  2195  2252  2310  2368  2426
                2484  2542  2600  2723#
MASKX   001244  172#
MASTEK  006142  1058  1187#
MCRLF   005672  831  954  1054  1055  1063  1187#  1328  1390
MCSRX   006072  737  1187#
MDATA   007544  984  994  1204#
MEMLIM  001304  188#  700*
MEPASS  005733  736  1187#
MERRPC  006217  1061  1187#
MERRX   006117  743  1187#
MERR2   005760  1187#  1590
MERR3   006005  673  1187#
MILK    001322  196#  484*  745  1288*  1293*  1297
MLOCK   006043  714  1187#
MNEW    006144  668  1187#
```

```
MODU    006704          1187*   1453
MPASSX  006106           741    1187*
MPFAIL  005675          1121    1187*
MQM     005666           861    1187*  1352   1427   1437   1447   1462   1476
MR      005755           723    1187*  1346
MRESET= 004000            96*
MSTCLR= 104412           235*   1126   1738   1739   1803   1853   1910   1964   2021   2078   2135   2192   2249
                        2307    2365   2423   2481   2539   2597
MTITLE  001000           136*    511
MTSTN   006130          1059    1187*  1330
MTSTPC  006031          1187*
MVECX   006100           739    1187*
NEXT    001216           157*    799   1092   1686*  1709*  1736*  1800*  1850*  1907*  1961*  2018*  2075*  2132*
                        2189*   2246*  2304*  2362*  2420*  2478*  2536*  2594*
NOACT   007154           523    1187*  1282
NODEV   002674           577     616*
NUM     006450          1187*   1382
OK      002646           603     610*   639
ONE     001302           187*
PACT00  001702           365*
PACT01  001706           368*
PACT02  001712           371*
PACT03  001716           374*
PACT04  001722           377*
PACT05  001726           380*
PACT06  001732           383*
PACT07  001736           386*
PACT10  001742           389*
PACT11  001746           392*
PACT12  001752           395*
PACT13  001756           398*
PACT14  001762           401*
PACT15  001766           404*
PACT16  001772           407*
PACT17  001776           410*
PARAM = 104405           225*   1331   1383   1396   1405   1484   1493
PARAM1  004234           881*    898
PARBIT= 040000            96*
PARERR  004310           884     886    888    897*   904    906    908
PASCNT  001230           162*    734*   735    746    771   1304*
PERFOR= 004537            96*
PFTAB   005430          1122    1128*
POPRO = 012600            72*   1086
POP1SP= 005726            70*
POP2SP= 022626            74*    801
PRIO    006547          1187*   1413
PS      = 177776          63*    476*   711*   1617*  1627*
PUSHRO= 010046            71*   1083
PUSH1S= 005746            69*
PUSH2S= 024646            73*
QV.FLG  001327           204*    482*   750*   790
RESREG  005220          1075    1078*
RESTAR  005350          1108    1114*
RESTRT  003534           749     753    761*
RESOS = 104407           229*   1078
RETURN  001214           156*    492*   720*   724    761*   799*   803   1092*  1095   1127   1345*  1355*  1357
```

```
ROMCLK= 104414          239#  1134  1137  1174  1179  1744  1746  1748  1756  1758  1814  1859  1861
                       1672  1874  1885  1887  1915  1917  1927  1929  1939  1941  1970  1972  1983
                       1985  1996  1998  2027  2029  2040  2042  2053  2055  2084  2086  2097  2099
                       2110  2112  2141  2143  2154  2156  2167  2169  2198  2200  2211  2213  2224
                       2226  2255  2257  2268  2270  2281  2283  2313  2315  2326  2328  2339  2341
                       2371  2373  2384  2386  2397  2399  2429  2431  2442  2444  2455  2457  2487
                       2489  2500  2502  2513  2515  2545  2547  2558  2560  2571  2573  2603  2605
                       2616  2618  2629  2631  2648  2650  2652  2660  2668  2676  2684  2692  2694
                       2696  2704
ROMDAT  026750         1863  1876  1889  1919  1931  1943  1974  1987  2000  2031  2044  2057  2088
                       2101  2114  2145  2158  2171  2202  2215  2228  2259  2272  2285  2317  2330
                       2343  2375  2388  2401  2433  2446  2459  2491  2504  2517  2549  2562  2575
                       2607  2620  2633  2709#
ROMMAP  012320         1665#  1715  1808  2728*  2731*
ROMNUM  022360         1688  1691*  1693#
ROM1    022362         1690  1694#
RUN     001316          193#   485*  1286*  1287*  1294
SAVACT  001312          191#   671  1588*
SAVNUM  001314          192#   479*   748*   751*  1581*
SAVPC   001276          185#   622*   642   929*  1099
SAVR0   001260          178#   938*   943  2734
SAVR1   001262          179#   628*   937*   944
SAVR2   001264          180#   936*   945  2734
SAVR3   001266          181#   935*   946
SAVR4   001270          182#   934*   947  2734
SAVR5   001272          183#   933*   948  2734
SAVSP   001274          184#
SAVOS = 104406          227#  1038
SCOPE = 104400          215#  1692  1724  1765  1837  1895  1949  2006  2063  2120  2177  2234  2291
                       2349  2407  2465  2523  2581  2639
SCOP1 = 104401          217#  1828  1868  1881  1894  1924  1936  1948  1979  1992  2005  2036  2049
                       2062  2093  2106  2119  2150  2163  2176  2207  2220  2233  2264  2277  2290
                       2322  2335  2348  2380  2393  2406  2438  2451  2464  2496  2509  2522  2554
                       2567  2580  2612  2625  2638
SETBR0  026674         2083  2096  2109  2657#
SETBR1  026702         2140  2153  2166  2665#
SETBR4  026710         2197  2210  2223  2673#
SETBR7  026716         2254  2267  2280  2681#
SETC    026724         1969  1982  1995  2689#
SETZ    026742         2026  2039  2052  2701#
SKIP    002632          573   576   597   600   606#
SOFTSW  010052         1229  1268#
SPACNT= 004713          964*   988   991*  1005#
SPEED   007340         1187#  1440
STACK = 001200           64#   477   690  1093  1116
STAT    001240          170#
STAT1   001366          251#  1301*  1712  1804  1854  1911  1965  2022  2079  2136  2193  2250  2308
                       2366  2424  2482  2540  2598
STAT2   001370          252#  1302*
STAT3   001372          253#  1303*  2717  2729
STRTSW  001236          169#   513*   516*   517   519   529   531   666   712   721  1323  1347*  1377
                       1601
SVOS    004402          933#
SWFLG   010016          480*   825  1224*  1251*  1257#
SWMES   007205         1187#  1227
SWMES1  007215         1187#  1230
```

```
SWR     001202          143*   497*   499    503*   513    671    676    781    788    812    827    1027   1032
                       1081    1088   1090   1152   1212   1250*
SWREG   000176          129*   503    1212   1270
SW00  = 000001           45*   517    1377   1601
SW01  = 000002           44*   721    1323   1347
SW02  = 000004           43*
SW03  = 000010           42*   666
SW04  = 000020           41*
SW05  = 000040           40*
SW06  = 000100           39*   1152
SW07  = 000200           38*
SW08  = 000400           37*   1088
SW09  = 001000           36*   812
SW10  = 002000           35*   1090
SW11  = 004000           34*   788
SW12  = 010000           33*   827    1027
SW13  = 020000           32*   1032
SW14  = 040000           31*
SW15  = 100000           30*
TEMP    001416          271*   971    1118*  1119*  1160*  1165*  1171*  1183*
TEMP1   001246          173*   537*   1189   1613*  1614*
TEMP2   001250          174*   538*   1191
TEMP3   001252          175*   540*   566*   618    627*   1193   1386   1389   1501*  1821*  1823*  1824*  1825*
                       1826*   2734
TEMP4   001254          176*   541*   1195   1399   1402   1408   1411   1487   1490   1496   1499   2734
TEMP5   001256          177*   542*   1197   1380*  1393*  1599
TIMER = 104416          243*
TKCSR   001204          148*   848    1214   1259   1655
TKDBR   001206          149*   850    856    1217   1219   1261   1657
TLAST = 026462         1350    2640*
TPCSR   001210          150*   832    854    1029   1262   1658
TPDBR   001212          151*   834*   856*   1031*  1264*  1660*
TRPOK   004730         1017*
TSTNO   001226          161*   491*   1102   1130   1334   1341   1343   1685*  1708*  1735*  1799*  1849*  1906*
                       1960*   2017*  2074*  2131*  2188*  2245*  2303*  2361*  2419*  2477*  2535*  2593*
TST1    022322         1337    1355   1685*
TST10   024132         1961    2017*
TST11   024326         2018    2074*
TST12   024522         2075    2131*
TST13   024716         2132    2188*
TST14   025112         2189    2245*
TST15   025306         2246    2303*
TST16   025502         2304    2361*
TST17   025676         2362    2419*
TST2    022760         1686    1708*
TST20   026072         2420    2477*
TST21   026266         2478    2535*
TST22   026462         2536    2593*  2640
TST23 = ****** U       2594
TST3    023052         1709    1735*
TST4    023166         1736    1799*
TST5    023362         1800    1849*
TST6    023556         1850    1906*
TST7    023736         1907    1960*
TTST    003612          715*   716*   718*   719*   782*
TWOSYN= 010000           96*
```

```
TYPDAT  005206          1053    1071    1074#
TYPE  = 104402           219#    511     523     535     620     625     633     636     668     673     714     723     736
                         737     739     741     743     831     844     861     954     994    1054    1055    1058    1059
                        1061    1063    1067    1072    1121    1227    1230    1282    1328    1346    1352    1390    1412
                        1426    1429    1436    1439    1446    1452    1461    1468    1475    1590    1690
TYPMSG  005106          1051    1054#
VEC     006526          1187#   1404
VECMAP  012010          1589    1601#
WHICH   012002          1392    1597#
WRDCNT  004710           962#    995#   1003#
WRKO.F  005174          1066    1069#
XBX     005000          1028    1030    1032#
XCSR    003546           738     763#
XERR    003570           744     772#
XHEAD   006224           535    1187#
XLOC    003022           593#    611#    614     645     658#
XPASS   003562           742     769#
XSTATQ  007454           544    1187#
XTSTN   005330          1060    1100#
XVEC    003554           740     766#
ZERO    001300           186#
$COD  = ****** U           1
$CRAP = 177777             1#    1673#   1676    1681#   1694#   1697    1704#   1725#   1728    1731#   1766#   1769    1795#
                        1838#   1841    1845#   1896#   1899    1902#   1950#   1953    1956#   2007#   2010    2013#   2064#
                        2067    2070#   2121#   2124    2127#   2178#   2181    2184#   2235#   2238    2241#   2292#   2295
                        2299#   2350#   2353    2357#   2408#   2411    2415#   2466#   2469    2473#   2524#   2527    2531#
                        2582#   2585    2589#
$ENDAD  003522           123     509     755#   1079
$N    = 000022             1#    1673    1681    1683    1688#   1694    1704    1706    1711#   1725    1731    1733    1738
                        1739#   1766    1795    1797    1803    1804#   1838    1845    1847    1853    1854#   1896    1902
                        1904    1910    1911#   1950    1956    1958    1964    1965#   2007    2013    2015    2021    2022#
                        2064    2070    2072    2078    2079#   2121    2127    2129    2135    2136#   2178    2184    2186
                        2192    2193#   2235    2241    2243    2249    2250#   2292    2299    2301    2307    2308#   2350
                        2357    2359    2365    2366#   2408    2415    2417    2423    2424#   2466    2473    2475    2481
                        2482#   2524    2531    2533    2539    2540#   2582    2589    2591    2597    2598#   2640#
$S    = 000024             1#    1686    1688#   1709    1711#   1736    1739#   1800    1804#   1850    1854#   1907    1911#
                        1961    1965#   2018    2022#   2075    2079#   2132    2136#   2189    2193#   2246    2250#   2304
                        2308#   2362    2366#   2420    2424#   2478    2482#   2536    2540#   2594    2598#
$Y    = 000017             1#     207#    215     217#    219#    221#    223#    225#    227#    229#    231#    233#    235#
                         237#    239#    241#    243#    245#
.     = 027734           108#     109     112#    119#    124#    127#    131#    135#    137#    189#    190#    191#    192#
                         272#    277#    279#    280#    281#    282#    284#    285#    286#    287#    289#    290#    291#
                         292#    294#    295#    296#    297#    299#    300#    301#    302#    304#    305#    306#    307#
                         309#    310#    311#    312#    314#    315#    316#    317#    319#    320#    321#    322#    324#
                         325#    326#    327#    329#    330#    331#    332#    334#    335#    336#    337#    339#    340#
                         341#    342#    344#    345#    346#    347#    349#    350#    351#    352#    354#    355#    356#
                         357#    515     525     657#    675    1110    1120    1168#   1203#   1205#   1260    1263    1284
                        1378    1422    1542    1546    1593    1624    1656    1659
.BEGIN  003152           690#
.CNVRT  004472           234     955#
.CONVR  004466           232     954#
.DATAC  005552           242    1159#
.DELAY  005436           238    1132#
.EOP    003364           731#   2594
.ERRTA  027602          1046    2734#
.HLT    004750           115    1026#
```