LPA/DMC-11

DIAGNOSTIC TEST 2
MD-11-DRLPM-A

EP-DRLPM-A-DL

FICHE 1 OF 1

MAR 1978
digital
MADE IN USA

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DRLPM-A-D
PRODUCT NAME:	LPA/DMC-11 DIAGNOSTIC TEST 2
DATE:	JAN. 1978
MAINTAINER:	DIAGNOSTICS

1. ABSTRACT

This diagnostic is one of a series of diagnostics aimed at the the lpa-11x system. please reference section 8.7 for a complete list.

The function of the m8200-yc 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 m8200-yc are correct in its environment.

this diagnostic requires the user to recable the system, that is, the lpa-11x i/o bus must join the unibus.

Parameters must be set up to alert the diagnostics to the m8200-yc 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.

It performs jump tests on the micro-processor and verifies the control ROM of the M8200-yc. This diagnostic will not run on a KMC (M8204), however it is possible to load the KMC CRAM with the m8200-yc micro-code. See test 2 for details.

Currently there are two 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 two diagnostics are:

1. DRLPL [REV] Basic W/R and Micro-processor tests
2. DRLPM [REV] Jump and CROM tests

2. REQUIREMENTS

2.1 EQUIPMENT

Any PDP11 family CPU (except an LSI-11) with minimum 8k memory
ASR 33 (or equilivalent)

SEQ 0003

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 m8200-yc 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 M8200-YC STATUS

PC	CSR	STAT1	STAT2	STAT3
001500	160010	145310	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 an M8200-YC. 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 M8200-YC'S TO BE TESTED?1

01
CSR ADDRESS?160010
VECTOR ADDRESS?310
BR PRIORITY LEVEL?  (4,5,6,7)?5
```

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 m8200-yc'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 M8200-YC'S DESIRED ACTIVE. Please note that a message is typed out for setting the switch register equal to m8200-yc's active. this means if the system has four m8200-ycs; 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 m8200-ycs 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 m8200-yc's 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 m8200-yc desired active.
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 enabled; 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 m8200-yc 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 m8200-yc'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 m8200-yc 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 m8200-yc 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 m8200-yc 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 m8200-yc diagnostic because the STATUS TABLE is overlayed. The current parameters in the STATUS TABLE are used when SW07=1 on start up.

m8200-yc must be on the unibus.

7.3 HARDWARE CONFIGURATION RESTRICTIONS

M8200-YC - 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 m8200-yc 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 m8200-yc'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 m8200-yc 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 M8200-YC 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 m8200-yc currently being tested. EXAMPLE: (RUN) 1302/0000000001000000 Means that m8200-yc no.06 is the m8200-yc now running.

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

These locations contain the information needed to test up to 16 (decimal) m8200-ycs sequentialy. they contain the CSR,VECTOR and STATUS concerning the configuration of each m8200-yc.

DMACTV (1306) Each bit set in this location indicates that the associated m8200-yc will be tested in turn. EXAMPLE: (DMACTV) 1276/0000000000011111 means that m8200-yc no. 00,01,02,03,04 will be tested. EXAMPLE: (DMACTV) 1276/0000000000010001 Means that m8200-yc no. 00,04 will be tested.

DMCSR (1404) Contains the CSR of the current m8200-yc 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 M8200-YC'S. the table can contain up to 16 M8200-YC'S. Following the map is a description of the bits for each map entry

MAP OF M8200-YC 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 M8200-YC. The PC shows where in core memory the first of the 4 words is. In the example above, the first m8200-yc's status is in locations, 1500, 1502, 1504, and 1506. The second m8200-yc status is located at 1510, 1512, 1514, and 1516. The information contained in each 4 word entry is defined as follows:

CSR: Contains M8200-YC CSR address

STAT1: BITS 00-08 IS M8200-YC 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
BITS 09-11 IS M8200-YC 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
m8200-yc/lpa micro code version 3

8.5 METHOD OF AUTO SIZING

8.5.1 FINDING THE CONTROL STATUS REGISTER.

The auto-sizing routine finds a m8200-yc as follows: It starts at address 170440 and tests all address in increments of 10 up to and including address 170500. 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 456 or 16520 a m8200-yc or KMC11 has been found, if not, the address is updated by 10 and the search continues. a 125252 indicates a KMC11 with CRAM. a 456 indicates a M8200-YC. 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 m8200-yc's in the system will be found by the auto-sizer. If it does not find a m8200-yc 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 m8200-yc) 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 m8200-yc 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.

8.7 lpa-11 (system) diagnostic summary

diagnostics for the lpa-11 are written at three levels: (1) total pdp-11 system, (2) lpa-11 system; and, (3) lpa-11 options.

level 1, is designed to isolate a failure to the lpa-11 system. all options on the pdp-11 are exercised.

level 2 diagnostics isolate a failure to the individual option within the lpa-11. the level 2 diagnostic is md-11-drlpa. when the user runs drlpa he can generally tell which option diagnostic (level 3) to run next. m8254 and m8200-yc errors may "look" alike and drlpa may not be able to distinguish between them. arbitration errors will not be detected by this diagnostic.

level three diagnostics aid in determining if the error was in fact on the option the drlpa specified. the user may "loop" on the error. within level three, there are two groups of diagnostics. the first group requires no "extra" work by the user in order to run. group "a" diagnostics do not check arbitration, and require extra time for execution. the second group (group "b") requires that the user reconfigure the pdp-11 system. this reconfiguration involves cabling the unibus to the

!pa's i/o bus.

the diagnostic for the m8254 falls into the group "b" catagory.

the lpa-11xx diagnostic kit will include:

option	group	diag. #	diag. title
lpa-11xx	level 2	md-11-drlpa	lpa-11 system diag.
m8254	"b"	md-11-drm8a	m8254 (jpbm) diag.
aa11k	a	md-11-drlpb	aa11k diag.
	b	md-11-dzaac	aa11-k diag.
ar11k	a	md-11-drlpc	lpa/ar11 diag. #1
	a	md-11-drlpd	lpa/ar11 diag. #2
	a	md-11-drlpe	lpa/ar11 diag. #3
	b	md-11-dzara	ar11 diag. #1
	b	md-11-dzarb	ar11 diag. #2
	b	md-11-dzarc	ar11 diag. #3
dr11k	a	md-11-drlpf	lpa/dr11k diag.
	b	md-11-dzdrg	dr11k diag.
kw11k	a	md-11-drlpg	lpa/kw11k diag.
	b	md-11-dzkwk	kw11k diag.
lps-11	a	md-11-drlph	lpa/lps-11 diag. #1
	a	md-11-drlpi	lpa/lps-11 diag. #2
	a	md-11-drlpj	lpa/lps-11 diag. #3
	b	md-11-dzlpc	lps-11 diag. #1
	b	md-11-dzlpd	lps-11 diag. #2
	b	md-11-dzlpi	lps-11 diag. #3
ad11k	a	md-11-drlpk	lpa/ad11k diag.
	b	md-11-dzadl	ad11k diag.
m8200-yc	b	md-11-dzlpl	lpa/m8200-yc basic micro-cpu r/w test
	b jmp+rom	md-11-dzlpm	lpa/m8200-yc

```
read test
```

```
                                ;*MAINDEC-11-DRLPM-A   LPA-M8200-YC CROM AND JUMP TESTS
                                ;*COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
                                ;*-------------------------------------------------------------------

                                ;STARTING PROCEDURE
                                ;LOAD PROGRAM
                                ;LOAD ADDRESS 000200
                                ;SWR=0  AUTOSIZE M8200-YC
                                ;SW07=1  USE CURRENT M8200-YC PARAMETERS
                                ;SW00=1  INPUT NEW M8200-YC PARAMETERS
                                ;PRESS START
                                ;PROGRAM WILL TYPE "MAINDEC-11-DRLPM-A   LPA-M8200-YC 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 M8200-YC PARAMETERS, =0,AUTOSIZE M8200-YC
000100                          SW06=100                ;=1, HALT BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION
000040                          SW05=40
000020                          SW04=20
000010                          SW03=10                 ;RESELECT M8200-YC'S TO BE TESTED (ACTIVE)
000004                          SW02=4                  ;LOCK ON TEST SELECT
000002                          SW01=2                  ;RESTART PROGRAM AT SELECTED TEST
000001                          SW00=1                  ;INPUT M8200-YC PARAMETERS
```

```
                        ;REGISTER DEFINITIONS
                        ;--------------------

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

                        ;LOCATION EQUIVALENCIES
                        ;----------------------

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

                        ;INSTRUCTION DEFINITIONS
                        ;-----------------------

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

                        ;BIT DEFINITIONS
                        ;---------------

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

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

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

            000024      .=24
000024  005346                  .PFAIL                  ;POWER FAIL HANDLER
000026  000340                  340                     ;SERVICE AT LEVEL 7
000030  004760                  .HLT                    ;ERROR HANDLER
000032  000340                  340                     ;SERVICE AT LEVEL 7
000034  004726                  .TRPSRV                 ;GENERAL HANDLER DISPATCH SERVICE
000036  000340                  340                     ;SERVICE AT LEVEL 7
            000040      .=40
000040  000000                  0                       ;SAVE FOR ACT-11 OR XXDP
000042  000000                  0                       ;RETURN ADDRESS IF UNDER ACT-11 OR XXDP
000044  000000                  0                       ;SAVE FOR ACT-11 OR XXDP
000046  003532                  $ENDAD                  ;FOR USE WITH ACT-11 OR XXDP
            000052      .=52
000052  000000                  0                       ;ACT-11 PROGRAM CHARACTERISTICS

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

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


            001000      .=1000
001000  005377  040515  047111  MTITLE: .ASCII  <377><12>/MAINDEC-11-DRLPM-A/<377>
(2) 001025      114  040520  046455          .ASCIZ  /LPA-M8200-YC CROM AND JUMP TESTS/<377>
(2)
            001200      .=1200

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

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

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

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

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

001214  000000                  RETURN: 0               ;SCOPE ADDRESS FOR LOOP ON TEST
001216  000000                  NEXT:   0               ;ADDRESS OF NEXT TEST TO BE EXECUTED
001220  000000                  LOCK:   0               ;ADDRESS FOR LOCK ON CURRENT DATA
001222  000003                  ICOUNT: 3               ;NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE EXECUTED
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         ;M8200-YC'S SELECTED ACTIVE.
001310  000001                  DMNUM:  .BLKW 1         ;OCTAL NUMBER OF M8200-YC'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 M8200-YC ITERATIONS WILL BE SUPPR
                                .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  003606                          .SCOPE
        104401                  SCOP1=TRAP+1            ;CALL TO LOOP ON CURRENT DATA HANDLER
001332  003746                          .SCOP1
        104402                  TYPE=TRAP+2             ;CALL TO TELETYPE OUTPUT ROUTINE
001334  003776                          .TYPE
        104403                  INSTR=TRAP+3            ;CALL TO ASCII STRING INPUT ROUTINE
001336  004060                          .INSTR
        104404                  INSTER=TRAP+4           ;CALL TO INPUT ERROR HANDLER
001340  004164                          .INSTER
        104405                  PARAM=TRAP+5            ;CALL TO NUMERICAL DATA INPUT ROUTINE
001342  004204                          .PARAM
        104406                  SAV05=TRAP+6            ;CALL TO REGISTER SAVE ROUTINE
001344  004404                          .SAV05
        104407                  RES05=TRAP+7            ;CALL TO REGISTER RESTORE ROUTINE
001346  004444                          .RES05
        104410                  CONVRT=TRAP+10          ;CALL TO DATA OUTPUT ROUTINE
001350  004476                          .CONVRT
        104411                  CNVRT=TRAP+11           ;CALL TO DATA OUTPUT ROUNTINE WITHOUT CR/LF.
001352  004502                          .CNVRT
        104412                  MSTCLR=TRAP+12          ;CALL TO ISUE A MASTER CLEAR
001354  005476                          .MSTCLR
        104413                  DELAY=TRAP+13           ;CALL TO DELAY
001356  005446                          .DELAY
        104414                  ROMCLK=TRAP+14          ;CALL TO CLOCK ROM ONCE
001360  005514                          .ROMCLK
        104415                  DATACLK=TRAP+15         ;CALL TO CLK DATA
001362  005562                          .DATACLK
        104416                  TIMER=TRAP+16           ;CALL TO dELAY A CLOCK TICK
001364  005626                          .TIMER

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

```
                                ;M8200-YC CONTROL INDICATORS FOR CURRENT M8200-YC UNDER TEST
                                ;-----------------------------------------------------
001366  000000                  STAT1:  0
001370  000000                  STAT2:  0
001372  000000                  STAT3:  0

                                ;M8200-YC VECTOR AND REGISTER INDIRECT POINTERS
                                ;---------------------------------------------
001374  000000                  DMRVEC: 0               ;POINTER TO M8200-YC RECEIVER INTERRUPT VECTOR
001376  000000                  DMRLVL: 0               ;POINTER TO M8200-YC RECEIVER INTERRUPT SERVICE PS
001400  000000                  DMTvEC: 0               ;POINTER TO M8200-YC TRANSMITTER INTERRUPT VECTOR
001402  000000                  DMTLVL: 0               ;POINTER TO M8200-YC TRANSMITTER INTERRUPT SERVICE PS
001404  000000                  DMCSR:  0               ;POINTER TO M8200-YC CONTROL STATUS REGISTER
001406  000000                  DMCSRH: 0               ;POINTER TO M8200-YC CONTROL STATUS REGISTER HIGH BYTE.
001410  000000                  DMCTL:  0               ;POINTER TO M8200-YC CONTOL OUT REGISTER
001412  000000                  DMP04:  0               ;POINTER TO M8200-YC PORT REGISTER(SEL 4)
001414  000000                  DMP06:  0               ;POINTER TO M8200-YC PORT REGISTER(SEL 6)

                                ;TEMP STORAGE
                                ;------------
001416  000000                  TEMP:   0
        001460                  .=.+40

                                ;M8200-YC STATUS TABLE AND ADDRESS ASSIGNMENTS
                                ;--------------------------------------------
        001500                  .=1500
001500                          DM.mAP:
001500  000001                  DMCR00: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 00
001502  000001                  DMS100: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 00
001504  000001                  DMS200: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 00
001506  000001                  DMS300: .BLKW   1               ;3RD STATUS WORD

001510  000001                  DMCR01: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 01
001512  000001                  DMS101: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 01
001514  000001                  DMS201: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 01
001516  000001                  DMS301: .BLKW   1               ;3RD STATUS WORD

001520  000001                  DMCr02: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 02
001522  000001                  DMS102: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 02
001524  000001                  DMS202: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 02
001526  000001                  DMS302: .BLKW   1               ;3RD STATUS WORD

001530  000001                  DMCR03: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 03
001532  000001                  DMS103: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 03
001534  000001                  DMS203: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 03
001536  000001                  DMS303: .BLKW   1               ;3RD STATUS WORD

001540  000001                  DMCR04: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 04
001542  000001                  DMS104: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 04
001544  000001                  DMS204: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 04
```

```
                        DMS304: .BLKW   1               ;3RD STATUS WORD

                        DMCr05: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 05
                        DMS105: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 05
                        DMS205: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 05
                        DMS305: .BLKW   1               ;3RD STATUS WORD

                        DMCR06: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 06
                        DMS106: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 06
                        DMS206: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 06
                        DMS306: .BLKW   1               ;3RD STATUS WORD

                        DMCR07: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 07
                        DMS107: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 07
                        DMS207: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 07
                        DMS307: .BLKW   1               ;3RD STATUS WORD

                        DMCR10: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 10
                        DMS110: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 10
                        DMS210: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 10
                        DMS310: .BLKW   1               ;3RD STATUS WORD

                        DMCR11: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 11
                        DMS111: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 11
                        DMS211: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 11
                        DMS311: .BLKW   1               ;3RD STATUS WORD

                        DMCr12: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 12
                        DMS112: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 12
                        DMS212: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 12
                        DMS312: .BLKW   1               ;3RD STATUS WORD

                        DMCR13: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 13
                        DMS113: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 13
                        DMS213: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 13
                        DMS313: .BLKW   1               ;3RD STATUS WORD

                        DMCR14: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 14
                        DMS114: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 14
                        DMS214: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 14
                        DMS314: .BLKW   1               ;3RD STATUS WORD

                        DMCR15: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 15
                        DMS115: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 15
                        DMS215: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 15
                        DMS315: .BLKW   1               ;3RD STATUS WORD

                        DMCR16: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 16
                        DMS116: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 16
                        DMS216: .BLKW   1               ;DDCMP LINE# FOR M8200-YC NUMBER 16
                        DMS316: .BLKW   1               ;3RD STATUS WORD

                        DMCr17: .BLKW   1               ;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 17
                        DMS117: .BLKW   1               ;VECTOR FOR M8200-YC NUMBER 17
```

```
356  001674  000001                DMS217: .BLKW   1          ;DDCMP LINE# FOR M8200-YC NUMBER 17
357  001676  000001                DMS317: .BLKW   1          ;3RD STATUS WORD
358
359  001700  000000                DM.END: 000000
```

```
                                ;M8200-YC PASS COUNT AND ERROR COUNT TABLE
                                ;-----------------------------------------

001702                          CNT.MAP:
001702  000000                  PACT00: 0               ;PASS COUNT FOR M8200-YC NUMBER 00
001704  000000                  ERCT00: 0               ;ERROR COUNT FOR M8200-YC NUMBER 00

001706  000000                  PACT01: 0               ;PASS COUNT FOR M8200-YC NUMBER 01
001710  000000                  ERCT01: 0               ;ERROR COUNT FOR M8200-YC NUMBER 01

001712  000000                  PACT02: 0               ;PASS COUNT FOR M8200-YC NUMBER 02
001714  000000                  ERCT02: 0               ;ERROR COUNT FOR M8200-YC NUMBER 02

001716  000000                  PACT03: 0               ;PASS COUNT FOR M8200-YC NUMBER 03
001720  000000                  ERCT03: 0               ;ERROR COUNT FOR M8200-YC NUMBER 03

001722  000000                  PACT04: 0               ;PASS COUNT FOR M8200-YC NUMBER 04
001724  000000                  ERCT04: 0               ;ERROR COUNT FOR M8200-YC NUMBER 04

001726  000000                  PACT05: 0               ;PASS COUNT FOR M8200-YC NUMBER 05
001730  000000                  ERCT05: 0               ;ERROR COUNT FOR M8200-YC NUMBER 05

001732  000000                  PACT06: 0               ;PASS COUNT FOR M8200-YC NUMBER 06
001734  000000                  ERCT06: 0               ;ERROR COUNT FOR M8200-YC NUMBER 06

001736  000000                  PACT07: 0               ;PASS COUNT FOR M8200-YC NUMBER 07
001740  000000                  ERCT07: 0               ;ERROR COUNT FOR M8200-YC NUMBER 07

001742  000000                  PACT10: 0               ;PASS COUNT FOR M8200-YC NUMBER 10
001744  000000                  ERCT10: 0               ;ERROR COUNT FOR M8200-YC NUMBER 10

001746  000000                  PACT11: 0               ;PASS COUNT FOR M8200-YC NUMBER 11
001750  000000                  ERCT11: 0               ;ERROR COUNT FOR M8200-YC NUMBER 11

001752  000000                  PACT12: 0               ;PASS COUNT FOR M8200-YC NUMBER 12
001754  000000                  ERCT12: 0               ;ERROR COUNT FOR M8200-YC NUMBER 12

001756  000000                  PACT13: 0               ;PASS COUNT FOR M8200-YC NUMBER 13
001760  000000                  ERCT13: 0               ;ERROR COUNT FOR M8200-YC NUMBER 13

001762  000000                  PACT14: 0               ;PASS COUNT FOR M8200-YC NUMBER 14
001764  000000                  ERCT14: 0               ;ERROR COUNT FOR M8200-YC NUMBER 14

001766  000000                  PACT15: 0               ;PASS COUNT FOR M8200-YC NUMBER 15
001770  000000                  ERCT15: 0               ;ERROR COUNT FOR M8200-YC NUMBER 15

001772  000000                  PACT16: 0               ;PASS COUNT FOR M8200-YC NUMBER 16
001774  000000                  ERCT16: 0               ;ERROR COUNT FOR M8200-YC NUMBER 16

001776  000000                  PACT17: 0               ;PASS COUNT FOR M8200-YC NUMBER 17
002000  000000                  ERCT17: 0               ;ERROR COUNT FOR M8200-YC NUMBER 17
```

413

FORMAT OF STATUS TABLE

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	
C	O	N	T	R	O	L		R	E	G	I	S	T	E	R	CSR
*	*	*	*	*	*	*	*	*	V	E	C	T	O	R	*	STAT1
*	B	M		A	D	D	*	*	L	I	N	E		#	*	STAT2
														*	*	STAT3

DEFINITION OF FORMAT

CSR: CONTAINS M8200-YC CSR ADDRESS

STAT1: BITS 00-08 IS M8200-YC 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 M8200-YC 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 M8200-YC-AL LOCAL HIGH SPEED MICRO-CODE
BIT1=0 M8200-YC-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  005346  000024          MOV     #.PFAIL,@#24    ;SET UP POWER FAIL VECTOR
002022  013737  001310  001314          MOV     DMNUM,SAVNUM    ;SAVE NUMBER OF DEVICES IN SYSTEM.
002030  005037  010056                  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  001012                          BNE     20$             ;BR IF YES
002222  022737  003532  000042          CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
002230  001406                          BEQ     20$             ;
002232  104402  001000                  TYPE    ,MTITLE         ;TYPE TITLE MESSAGE
002236  104402  023251                  TYPE    ,ROM1           ;TYPE VERSION MESSAGE
002242  104402  022454                  TYPE    ,MESWCH         ;TYPE SWITCH 7 MESSAGE
002246  004737  007646          20$:    JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
002252  017737  176724  001236          MOV     @SWR,STRTSW     ;STORE STARTING SWITCHES
002260  005737  000042                  TST     @#42            ;IS IT RUNNING IN AUTO MODE?
002264  001402                          BEQ     .+6             ;BR IF NO
002266  005037  001236                  CLR     STRTSW          ;IF YES, CLEAR SWITCHES
002272  032737  000001  001236          BIT     #SW00,STRTSW    ;IF SW00=1, QUESTIONS ARE ASKED.
002300  001012                          BNE     17$             ;BR IF SW00=1
002302  105737  001236                  TSTB    STRTSW          ;BIT7=1??
```

```
                                100007                  BPL     17$             ;BR IF SW07=0
                                005737  001306          TST     DMACTV          ;ARE ANY DEVICES SELECTED?
                                001006                  BNE     16$             ;BR IF YES
                                104402  007175          TYPE,   NOACT           ;NO DEVICES SELECTED.
                                000000                  HALT                    ;STOP THE SHOW
                                000776                  BR      .-2             ;DISQUALIFY CONTINUE SWITCH
                                004737  010552  17$:    JSR     PC,AUTO.SIZE    ;GO DO THE AUTO SIZE
                                105737  001324  16$:    TSTB    INIFLG          ;FIRST TIME?
                                001410                  BEQ     21$             ;BR IF YES
                                105737  001236          TSTB    STRTSW          ;IF USING SAME PARAMETERS DON't TYPE MAP
                                100431                  BMI     1$
                                032737  000006  001236  BIT     #BIT1!BIT2,STRTSW;IS TEST NO. OR LOCK SELECTED
                                001403                  BEQ     24$             ;IF NO THEN TYPE STATUS
                                000424                  BR      1$              ;IF YES DO NOT TYPE STATUS
                                005137  001324  21$:    COM     INIFLG          ;SET FLAG
                                104402  006237  24$:    TYPE    ,XHEAD          ;TYPE HEADER
                                012704  001500          MOV     #DM.MAP,R4      ;SET POINTER
                                010437  001246  5$:     MOV     R4,TEMP1        ;SET ADDRESS
                                012437  001250          MOV     (R4)+,TEMP2     ;SET CSR
                                001411                  BEQ     1$              ;ALL DONE IF ZERO
                                012437  001252          MOV     (R4)+,TEMP3     ;SET STAT1
                                012437  001254          MOV     (R4)+,TEMP4     ;SET STAT2
                                012437  001256          MOV     (R4)+,TEMP5     ;SET STAT3
                                104410                  CONVRT                  ;TYPE OUT STATUS MAP
                                007514                  XSTATQ                  ;
                                000762                  BR      5$
                                012700  001500  1$:     MOV     #DM.mAP,R0      ;R0 POINTS TO STATUS TABLE

                                ;;*************************************************************************
                                ;;*AUTO SIZE TEST
                                ;;*THIS TEST VERIFYS THAT THE MB200-YCS 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 MB200-YC, THE FIRST
                                ;;*MB200-YC 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).
                                ;;*************************************************************************

002434  013746  000004          MOV     @#4,-(SP)       ;SAVE LOC 4
002440  013746  000006          MOV     @#6,-(SP)       ;SAVE LOC 6
002444  005037  000006          CLR     @#6             ;CLEAR VEC+2
002450  005037  001252          CLR     TEMP3           ;CLEAR FLAG
002454  005005                  CLR     R5              ;R5=0=DMC, R5=-1=KMC
002456  011037  001404  AUSTRT: MOV     (R0),DMCSR      ;GET NEXT DMC CSR
002462  001564                  BEQ     AUDONE          ;BR IF DONE
002464  005705                  TST     R5              ;DMC OR KMC?
002466  001005                  BNE     1$              ;BR IF KMC
002470  032760  100000  000002  BIT     #BIT15,2(R0)    ;CHECK FOR DMC CSR
002476  001061                  BNE     SKIP            ;SKIP IF NOT DMC
```

```
002500  000404                          BR      2$              ;ITS A DMC SO CONTINUE
002502  032760  100000  000002  1$:     BIT     #BIT15,2(R0)    ;CHECK FOR KMC CSR
002510  001454                          BEQ     SKIP            ;SKIP IF NOT KMC
002512  012737  002704  000004  2$:     MOV     #NODEV,@#4      ;SET UP FOR TIMEOUT
002520  005705                          TST     R5              ;DMC OR KMC?
002522  001003                          BNE     3$              ;BR IF KMC
002524  012703  000006                  MOV     #6,R3           ;R3 IS COUNT OF DEVICES BEFORE DMC
002530  000402                          BR      4$              ;GO ON
002532  012703  000010          3$:     MOV     #10,R3          ;R3 IS COUNT OF DEVICES BEFORE KMC
002536  012702  003020          4$:     MOV     #DEVTAB,R2      ;R2 IS DEVICE TABLE PONTER
002542  012701  160010                  MOV     #160010,R1      ;START WITH ADDRESS 160010
002546  005711                  FLOAT:  TST     (R1)            ;CHECK ADDRESS IN R1
002550  111204                          MOVB    (R2),R4         ;IF NO TIMEOUT, GET NEXT ADDRESS
002552  060401                          ADD     R4,R1           ;IN R1
002554  005201                          INC     R1              ;
002556  040401                          BIC     R4,R1           ;
002560  005703                          TST     R3              ;ANY MORE DEVICES TO cHECK FOR?
002562  001371                          BNE     FLOAT           ;BR IF YES
002564  012737  002710  000004          MOV     #ERR,@#4        ;OK ONLY DMC'S ARE LEFT, SET UP FOR TIMEOUT
002572  010137  003032                  MOV     R1,XLOC         ;SAVE FIRST DMC/KMC ADDRESS
002576  005705                  FY:     TST     R5              ;DMC OR KMC?
002600  001005                          BNE     1$              ;BR IF KMC
002602  032760  100000  000002          BIT     #BIT15,2(R0)    ;CHECK FOR DMC CSR
002610  001014                          BNE     SKIP            ;SKIP IF NOT DMC
002612  000404                          BR      2$              ;ITS A DMC SO CONTINUE
002614  032760  100000  000002  1$:     BIT     #BIT15,2(R0)    ;CHECK FOR KMC CSR
002622  001407                          BEQ     SKIP            ;SKIP IF NOT KMC
002624  005711                  2$:     TST     (R1)            ;CHECK DMC ADDRESS
002626  020137  001404                  CMP     R1,DMCSR        ;DOEs IT MATCH
002632  001411                          BEQ     OK              ;BR IF YES
002634  062701  000010                  ADD     #10,R1          ;GET NEXT DMC ADDRESS
002640  000756                          BR      FY              ;DO IT AGAIN
002642  062700  000010          SKIP:   ADD     #10,R0          ;SKIP TO NEXT CSR IN TABLE
002646  011037  001404                  MOV     (R0),DMCSR      ;GET NEXT CSR
002652  001470                          BEQ     AUDONE          ;BR IF DONE
002654  000750                          BR      FY              ;ELSE CONTINUE
002656  062700  000010          OK:     ADD     #10,R0          ;SKIP TO NEXT DMC CSR
002662  062737  000010  003032          ADD     #10,XLOC        ;UPDATE EXPECTED DMC/KMC ADDRESS
002670  011037  001404                  MOV     (R0),DMCsR      ;GET NEXT DMC/KMC CSR
002674  001457                          BEQ     AUDONE          ;BR IF DONE
002676  013701  003032                  MOV     XLOC,R1         ;GET EXPECTED DMC/KMC ADDRESS
002702  000735                          BR      FY              ;CONTINUE
002704  122243                  NODEV:  CMPB    (R2)+,-(R3)     ;ON TIMEOUT, INC R2, DEC R3
002706  000002                          RTI                     ;RETURN
002710  005737  001252          ERR:    TST     TEMP3           ;CHECK FLAG IF = 0 TYPE HEADER
002714  001014                          BNE     1$              ;SKIP HEADER
002716  104402                          TYPE                    ;TYPEOUT HEADER MESSAGE
002720  007244                          CONERR                  ;CONFIGURATION ERROR!!!!
002722  012737  002710  001276          MOV     #ERR,SAVPC      ;SAVE PC FOR TYPEOUT
002730  104411                          CNVRT                   ;TYPE OUT ERROR PC
002732  003000                          ERRPC                   ;
002734  104402                          TYPE                    ;TYPE REST OF HEADER
002736  007323                          CNERR                   ;
002740  012737  177777  001252          MOV     #-1,TEMF3       ;SET FLAG SO IT ONLY GETS TYPED ONCE
```

```
                                                1$:     MOV     R1,SAVR1        ;SAVE R1 FOR TYPEOUT
                                                        CONVRT
                                                        CONTAB                  ;TYPE CSR VALUES
                                                        TST     R5              ;DMC OR KMC ?
                                                        BNE     3$              ;BR IF KMC
                                                        TYPE
                                                        DMCM
                                                        BR      4$              ;CONTINUE
                                                3$:     TYPE
                                                        KMCM
                                                4$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
                                                        BR      OK              ;BR TO GET OUT
                                                ERRPC:  1
                                    002                 .BYTE   6,2
                                                        SAVPC
                                                CONTAB: 2
                                    004                 .BYTE   6,4
                                                        XLOC
                                    002                 .BYTE   6,2
                                                        DMCSR
                                                DEVTAB: .BYTE   7               ;DJ
                                                        .BYTE   17              ;DH
                                                        .BYTE   7               ;DQ
                                                        .BYTE   7               ;DU
                                                        .BYTE   7               ;DUP
                                                        .BYTE   7               ;LK
                                                        .BYTE   7               ;DMC
                                                        .BYTE   7               ;DZ
                                                        .BYTE   7               ;KMC
                                                .EVEN
                                                XLOC:   0
                                                AUDONE: TST     R5              ;DMC?
                                                        BNE     1$              ;BR IF KMC AND ALL DONE
                                                        MOV     #-1,R5          ;SET R5 TO -1 (KMC)
                                                        MOV     #DM.MAP,R0      ;RESET R0 TO START OF TABLE
                                                        BR      AUSTRT          ;GO DO KMC'S
                                                1$:     MOV     (SP)+,@#6       ;RESTORE LOC 6
                                                        MOV     (SP)+,@#4       ;RESTORE LOC 4
                                    001236              BIT     #SW03,STRTSW    ;SELECT SPECIFIC DEVICES??
                                                        BEQ     3$              ;BR IF NO.
                                                        TYPE    ,MNEW           ;TYPE THE MESSAGE.
                                                        CLR     R0              ;ZERO DATA LIGHTS
                                                        HALT                    ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
                                    001312              CMP     @SWR,SAVACT     ;IS THE NUMBER VALID?
                                                        BLOS    2$              ;BR IF NUMBER IS OK.
                                                        TYPE    ,MERR3          ;TELL USER OF INVALID NUMBER.
                                                        HALT                    ;STOP EVERY THING.
                                                        BR      .-2             ;RESTART THE PROGRAM AGAIN.
                                    001306      2$:     MOV     @SWR,DMACTV     ;GET NEW DEVICE PATTERN
                                                        MOV     DMACTV,R0       ;SHOW THE USER WHAT HE SELECTED.
                                                        HALT                    ;CONTINUE DYNAMIC SWITCHES.
                                                3$:     MOV     #300,R0         ;PREPARE TO CLEAR THE FLOATING
                                                        MOV     #302,R1         ;VECTOR AREA. 300-776
                                                4$:     MOV     R1,(R0)+        ;START PUTTING "PC+2 - HALT"
```

```
                                        CLR     (R1)+           ;IN VECTOR AREA.
                                        CMP     (R0)+,(R1)+     ;POP POINTERS
                                        CMP     #1000,R0        ;ALL DONE??
                                        BNE     4$              ;BR IF NO.

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

                        .BEGIN: MOV     #STACK,SP       ;SET UP STACK
                                MOV     @#6,-(SP)       ;SAVE LOC 6
                                MOV     @#4,-(SP)       ;SAVE LOC 4
                                CLR     R0              ;START AT 0
                                MOV     #2$,@#4         ;SET UP FOR TIME OUT
                                CLR     @#6             ;TO AUTOSIZE MEMORY
                        6$:     TST     (R0)+           ;CHECK ADDRESS IN R0
                                CMP     #157776,R0      ;IS IT AT LEAST 28K
                                BNE     6$              ;BR IF NO
                                SUB     #7776,R0        ;SAVE 2K FOR MONITORS
                        7$:     MOV     R0,MEMLIM       ;STORE MEMORY LIMIT
                                MOV     (SP)+,@#4       ;RESTORE LOC 4
                                MOV     (SP)+,@#6       ;RESTORE LOC 6
                                BR      10$             ;CONTINUE
                        2$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
                                SUB     #4,R0           ;GET LAST GOOD ADDRESS
                                SUB     #7776,R0        ;SAVE 2K FOR MONITORS
                                CMP     #30000,R0       ;IS IT 8K?
                                BNE     7$              ;BR IF NO
                                MOV     #37400,R0       ;IF 8K DON'T SAVE 2K
                                BR      7$              ;
                10$:    MOV     #340,PS         ;LOCK OUT INTERRUPTS
                        BIT     #BIT2,STRTSW    ;CHECK FOR LOCK ON TEST
                        BEQ     1$              ;BR IF NO LOCK DESIRED.
                        TYPE    ,MLOCK          ;TYPE LOCK SELECTED.
                        MOV     #NOP,TTST       ;ADJUST SCOPE ROUTINE.
                        MOV     #NOP,TTST+2     ;SET UP TO LOCK
                        BR      3$              ;CONTINUE ALONG.
                1$:     MOV     #BRW,TTST       ;PREPARE NORMAL SCOPE ROUTINE
                        MOV     #BRX,TTST+2     ;LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
                3$:     MOV     #CYCLE,RETURN   ;START AT "CYCLE" FIND WHICH DEVICE TO TEST
                4$:     BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED?
                        BNE     5$              ;BR IF YES
                        TYPE    ,MR             ;TYPE R
                5$:     JMP     @RETURN         ;START TESTING
```

```
                                                ;END OF PASS
                                                ;TYPE NAME OF TEST
                                                ;UPDATE PASS COUNT
                                                ;CHECK FOR EXIT TO ACT-11
                                                ;RESTART TEST
                                
003374  000005                          .EOP:   RESET                   ;MAKE THE  WORLD CLEAN AGAIN.
003376  005037  001234                          CLR     LSTERR          ;CLEAR LAST ERROR PC
003402  105037  001325                          CLRB    ERRFLG          ;CLEAR ERROR FLAG
003406  005237  001230                          INC     PASCNT          ;UPDATE PASS COUNT
003412  013777  001230  175560                  MOV     PASCNT,@DISPLAY ;DISPLAY PASS COUNT
003420  104402  005743                          TYPE    ,MEPASS         ;TYPE END PASS
003424  104402  006102                          TYPE    ,MCSRX          ;TYPE CSR
003430  104411  003556                          CNVRT   ,XCSR           ;SHOW IT
003434  104402  006110                          TYPE    ,MVECX          ;TYPE VECTOR
003440  104411  003564                          CNVRT   ,XVEC           ;SHOW IT
003444  104402  006116                          TYPE    ,MPASSX         ;TYPE PASSES
003450  104411  003572                          CNVRT   ,XPASS          ;SHOW IT
003454  104402  006127                          TYPE    ,MERRX          ;TYPE ERRORS
003460  104411  003600                          CNVRT   ,XERR           ;SHOW IT
003464  013700  001322                          MOV     MILK,R0         ;GET POINTER TO PASS COUNT
003470  013720  001230                          MOV     PASCNT,(R0)+    ;STORE PASS COUNT FOR THIS M8200-YC
003474  013720  001232                          MOV     ERRCNT,(R0)+    ;STORE ERROR COUNT FOR THIS M8200-YC
003500  005337  001314                          DEC     SAVNUM          ;ARE ALL DEVICES TESTED?
003504  001017                              .   BNE     RESTRT          ;BR IF NO
003506  112737  000377  001327                  MOVB    #377,QV.FLG     ;SET THE QUICK VERIFY FLAG.
003514  013737  001310  001314                  MOV     DMNUM,SAVNUM    ;RESTORE THE COUNT
003522  013701  000042                          MOV     @#42,R1         ;CHECK FOR ACT-11 OR DDP
003526  001406                                  BEQ     RESTRT          ;IF NOT, CONTINUE TESTING
003530  000005                                  RESET                   ;STOP THE SHOW--CLEAR THE WORLD
003532                              $ENDAD:
003532  004711                                  JSR     PC,(R1)
003534  000240                                  NOP
003536  000240                                  NOP
003540  000240                                  NOP
003542  000240                                  NOP
003544  012737  010120  001214  RESTRT:  MOV     #CYCLE,RETURN
003552  000137  010120                          JMP     CYCLE
003556  000001                          XCSR:   1
003560     006      002                         .BYTE   6,2
003562  001404                                  DMCSR
003564  000001                          XVEC:   1
003566     004      002                         .BYTE   4,2
003570  001374                                  DMRVEC
003572  000001                          XPASS:  1
003574     006      002                         .BYTE   6,2
003576  001230                                  PASCNT
003600  000001                          XERR:   1
003602     006      002                         .BYTE   6,2
003604  001232                                  ERRCNT

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

```
                                .SCOPE: JSR     PC,CKSWR        ;CKECK FOR SOFT SWR
                                        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 M8200-YC ADDRESS
                                        JMP     @RETURN         ;GO DO THE TEST
                                BRW:    1407
                                BRX:    437
                                DONE:   0


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


                                .SCOP1: JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
                                        BIT     #SW09,@SWR      ;IS SW09=1(SET)?
                                        BEQ     1$              ;BR IF NOT SET.
                                        TST     LOCK
                                        BEQ     1$
                                        MOV     LOCK,(SP)       ;GOTO THE ADDRESS IN LOCK.
                                1$:     RTI                     ;GO BACK.


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


                                .TYPE:  MOV     R5,-(SP)        ;SAVE R5 ON THE STACK.
                                        MOV     @2(SP),R5       ;GET ADDRESS OF MESSAGE.
                                        ADD     #2,2(SP)        ;POP OVER ADDRESS.
                                4$:     TST     SWFLG           ;SOFT SWR MESSAGE?
                                        BNE     1$              ;IF YES TYPE IT OUT REGARDLESS OF SW12
                                        BIT     #SW12,@SWR      ;INHIBIT ALL PRINT OUT??
                                        BNE     3$              ;BR IF NO PRINT OUT WANTED (SW12=1)
                                1$:     TSTB    (R5)            ;IS NUMBER MINUS? (MSB=1(BIT7))
                                        BPL     2$              ;BR IF NUMBER IS PLUS
                                        TYPE    ,MCRLF          ;TYPE A CR/LF!
                                2$:     TSTB    @TPCSR          ;TTY READY?
```

```
                                        BPL     2$              ;BR IF NO.
                                        MOVB    (R5)+,@TPDBR    ;PRINT CURRENT CHAR.
                                        BNE     4$              ;IF NOT ZERO KEEP PRINTING!
                                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

004444  013700  001260        .RESO5: MOV     SAVR0,R0        ;RESTORE R0
004450  013701  001262                MOV     SAVR1,R1        ;RESTORE R1
004454  013702  001264                MOV     SAVR2,R2        ;RESTORE R2
004460  013703  001266                MOV     SAVR3,R3        ;RESTORE R3
004464  013704  001270                MOV     SAVR4,R4        ;RESTORE R4
004470  013705  001272                MOV     SAVR5,R5        ;RESTORE R5
004474  000002                      RTI                     ;LEAVE

                                        ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
                                        ;-----------------------------------------------------

004476  104402  005702        .CONVR: TYPE    ,MCRLF
004502  010046                .CNVRT: MOV     R0,-(SP)
004504  010146                        MOV     R1,-(SP)
004506  010346                        MOV     R3,-(SP)
004510  010446                        MOV     R4,-(SP)
004512  010546                        MOV     R5,-(SP)
004514  017601  000012                MOV     @12(SP),R1
004520  062766  000002  000012        ADD     #2,12(SP)
004526  012137  004720                MOV     (R1)+,WRDCNT
004532  112137  004722        1$:     MOVB    (R1)+,CHRCNT
004536  112137  004723                MOVB    (R1)+,SPACNT
004542  013137  004724                MOV     @(R1)+,BINWRD
004546  122737  000003  004722        CMPB    #3,CHRCNT
004554  001003                      BNE     2$
004556  042737  177400  004724      BIC     #177400,BINWRD
004564  013704  004724        2$:     MOV     BINWRD,R4
004570  113705  004722                MOVB    CHRCNT,R5
004574  012700  001416                MOV     #TEMP,R0
004600  010403                3$:     MOV     R4,R3
004602  042703  177770                BIC     #177770,R3
004606  062703  000060                ADD     #060,R3
004612  110320                      MOVB    R3,(R0)+
004614  000241                      CLC
004616  006004                      ROR     R4
004620  000241                      CLC
004622  006004                      ROR     R4
004624  000241                      CLC
004626  006004                      ROR     R4
004630  005305                      DEC     R5
004632  001362                      BNE     3$
004634  012703  007604              MOV     #MDATA,R3
004640  114023                4$:   MOVB    -(R0),(R3)+
004642  105337  004722              DECB    CHRCNT
004646  001374                      BNE     4$
004650  105737  004723              TSTB    SPACNT
004654  001405                      BEQ     6$
004656  112723  000040        5$:   MOVB    #040,(R3)+
004662  105337  004723              DECB    SPACNT
004666  001373                      BNE     5$
004670  105013                6$:   CLRB    (R3)
004672  104402  007604              TYPE    ,MDATA
```

```
 997  004676  005337  004720                    DEC     WRDCNT
 998  004702  001313                            BNE     1$
 999  004704  012605                            MOV     (SP)+,R5
1000  004706  012604                            MOV     (SP)+,R4
1001  004710  012603                            MOV     (SP)+,R3
1002  004712  012601                            MOV     (SP)+,R1
1003  004714  012600                            MOV     (SP)+,R0
1004  004716  000002                            RTI
1005  004720  000000                WRDCNT: 0
1006  004722  000000                CHRCNT: 0
1007          004723                SPACNT=CHRCNT+1
1008  004724  000000                BINWRD: 0
1009
1010
1011                                ;TRAP DISPATCH SERVICE
1012                                ;ARGUMENT OF TRAP IS EXTRACTED
1013                                ;AND USED AS OFFSET TO oBTAIN POINTER
1014                                ;TO SELECTED SUBROUTINE
1015
1016  004726  011646                .TRPSR: MOV     (SP),-(SP)      ;GET PC OF RETURN
1017  004730  162716  000002                SUB     #2,(SP)         ;=PC OF TRAP
1018  004734  017616  000000                MOV     @(SP),(SP)      ;GET TRP
1019  004740  006316                TRPOK:  ASL     (SP)            ;MULTIPLY TRAP ARG BY 2
1020  004742  042716  177001                BIC     #177001,(SP)    ;CLEAR UNWANTED BITS
1021  004746  062716  001330                ADD     #.TRPTAB,(SP)   ;POINTER TO SUBROUTINE ADDRESS
1022  004752  017616  000000                MOV     @(SP),(SP)      ;SUBROUTINE ADDRESS
1023  004756  000136                        JMP     @(SP)+          ;GO TO SUBROUTINE
1024
1025                                ;ERROR HANDLER
1026                                ;-------------
1027
1028  004760  004737  007646        .HLT:   JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
1029  004764  032777  010000 174210        BIT     #SW12,@SWR      ;BELL ON ERROR?
1030  004772  001406                        BEQ     XBX             ;BR IF NO BELL
1031  004774  105777  174210                TSTB    @TPCSR          ;TTY READY.
1032  005000  100003                        BPL     XBX             ;DON'T WAIT IF TTY NOT READY.
1033  005002  112777  000207 174202        MOVB    #207,@TPDBR     ;PUSH A BELL AT THE TTY.
1034  005010  032777  020000 174164 XBX:    BIT     #SW13,@SWR      ;DELETE ERROR PRINT OUT?
1035  005016  001105                        BNE     HALTS           ;BR IF NO PRINT OUT WANTED.
1036  005020  021637  001234                CMP     (SP),LSTERR     ;WAS THIS ERROR FOUND LAST TIME?
1037  005024  001404                        BEQ     1$              ;BR IF YES
1038  005026  011637  001234                MOV     (SP),LSTERR     ;RECORD BEING HERE
1039  005032  105037  001325                CLRB    ERRFLG          ;PREPARE HEADER
1040  005036  104406                1$:     SAVOS                   ;SAVE ALL PROC REGISTERS
1041  005040  011605                        MOV     (SP),R5         ;GET THE PC OF ERROR
1042  005042  162705  000002                SUB     #2,R5           ;GET ADDRESS OF TRAP CALL
1043  005046  011504                        MOV     (R5),R4         ;GET HLT INSTRUCTION
1044  005050  006304                        ASL     R4              ;MULT BY TWO
1045  005052  061504                        ADD     (R5),R4         ;DOUBLE IT
1046  005054  006304                        ASL     R4              ;MULT AGAIN
1047  005056  042704  177001                BIC     #177001,R4      ;CLEAR JUNK
1048  005062  062704  023544                ADD     #.ERRTAB,R4     ;GET POINTER
1049  005066  012437  005202                MOV     (R4)+,ERRMSG    ;GET ERROR MESSAGE
1050  005072  012437  005214                MOV     (R4)+,DATAHD    ;GET DATA HEADRER
```

```
1051  005076  011437  005226                MOV     (R4),DATABP     ;GET DATA TABLE
1052  005102  105737  001325                TSTB    ERRFLG          ;TYPE HEADREER
1053  005106  001403                       BEQ     TYPMSG          ;BR IF YES
1054  005110  005737  005226                TST     DATABP          ;DOES DATA TABLE EXIST?
1055  005114  001040                       BNE     TYPDAT          ;BR IF YES.
1056  005116  104402  005702        TYPMSG: TYPE    ,MCRLF
1057  005122  104402  005702                TYPE    ,MCRLF
1058  005126  005737  001220                TST     LOCK
1059  005132  001402                       BEQ     1$
1060  005134  104402  006152                TYPE    ,MASTEK
1061  005140  104402  006140        1$:     TYPE    ,MTSTN
1062  005144  104411  005340                CNVRT   ,XTSTN          ;SHOW IT
1063  005150  104402  006232                TYPE    ,MERRPC         ;TYPE PC.
1064  005154  104411  005332                CNVRT   ,ERTABO         ;SHOW IT
1065  005160  104402  005702                TYPE    ,MCRLF          ;GIVE A CR/LF
1066  005164  112737  177777  001325        MOVB    #-1,ERRFLG      ;NO MORE HEADER UNLESS NO DATA TABLE.
1067  005172  005737  005202                TST     ERRMSG          ;IS THERE AN ERROR MESSAGE?
1068  005176  001402                       BEQ     WRKO.FM         ;BR IF NO.
1069  005200  104402                       TYPE                    ;TYPE
1070  005202  000000        ERRMSG: 0                               ;    ERROR MESSAGE
1071  005204                WRKO.FM:
1072  005204  005737  005214                TST     DATAHD          ;DATA HEADER?
1073  005210  001402                       BEQ     TYPDAT          ;BR IF NO
1074  005212  104402                       TYPE                    ;TYPE
1075  005214  000000        DATAHD: 0                               ;    DATA HEADER
1076  005216  005737  005226        TYPDAT: TST     DATABP          ;DATA TABLE?
1077  005222  001402                       BEQ     RESREG          ;BR IF NO.
1078  005224  104410                       CONVRT                  ;SHOW
1079  005226  000000        DATABP: 0                               ;    DATA TABLE
1080  005230  104407        RESREG: RESO5                           ;RESTORE PROC REGISTERS
1081  005232  022737  003532  000042  HALTS:  CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, HALT!!
1082  005240  001403                       BEQ     1$
1083  005242  005777  173734                TST     @SWR            ;HALT ON ERROR?
1084  005246  100005                       BPL     EXITER          ;BR IF NO HALT ON ERROR
1085  005250  010046                1$:     PUSHRO                  ;SAVE RO
1086  005252  016600  000002                MOV     2(SP),RO        ;SHOW ERROR PC IN DATA LIGHTS
1087  005256  000000                       HALT                    ;HALT
1088  005260  012600                       POPRO                   ;GET RO
1089  005262  005237  001232        EXITER: INC     ERRCNT          ;UPDATE ERROR COUNT
1090  005266  032777  000400  173706        BIT     #SW08,@SWR      ;GOTO TOP OF TEST?
1091  005274  001007                       BNE     1$              ;BR IF YES
1092  005276  032777  002000  173676        BIT     #SW10,@SWR      ;GOTO NEXT TEST?
1093  005304  001411                       BEQ     2$              ;BR IF NO
1094  005306  013737  001216  001214        MOV     NEXT,RETURN     ;SET FOR NEXT TEST
1095  005314  012706  001200        1$:     MOV     #STACK,SP       ;RESET SP
1096  005320  013701  001404                MOV     DMCSR,R1        ;SET UP R1
1097  005324  000177  173664                JMP     @RETURN         ;GOTO SPECIFIED TEST
1098  005330  000002        2$:     RTI                             ;RETURN
1099  005332  000001        ERTABO: 1
1100  005334      006  002                .BYTE   6,2
1101  005336  001276                       SAVPC
1102  005340  000001        XTSTN:  1
1103  005342      003  002                .BYTE   3,2
1104  005344  001226                       TSTNO
```

```
                                        ;ENTER HERE ON POWER FAILURE
                                        ;--------------------------

                              .PFAIL:
005346 012737 005360 000024             MOV     #PESTART,24             ;SET UP FOR POWER UP TRAP
005354 000000                           HALT                            ;HALT ON POWER DOWn NORMAL
005356 000777                           BR      .

                                        ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED

                              RESTAR:
005360 012737 005346 000024             MOV     #.PFAIL,24      ;SET UP FOR POWER FAILURE
005366 012706 001200                    MOV     #STACK,SP       ;RESET THE STACK POINTER
005372 013701 001404                    MOV     DMCSR,R1        ;RESTORE R1
005376 005037 001416                    CLR     TEMp            ;READY FOR TIMMER
005402 005237 001416                    INC     TEMP            ;PLUS ONE TO tHE TIMER!
005406 001375                           BNE     .-4             ;BR IF MORE TO GO
005410 104402 005705                    TYPE    ,MPFAIL         ;TYPE THE MESSAGE
005414 104411 005440                    CNVRT   ,PFTAB          ;TELL WHAT TESt TO rETURN TO.
005420 105037 001325                    CLRB    ERRFLG          ;START CLEAN
005424 005037 001234                    CLR     LSTERR          ;**************
005430 005011                           CLR     (R1)            ;CLEAR MAINT BITS
005432 104412                           MSTCLR                  ;START CLEAN UP OF DEVICE
005434 000177 173554                    JMP     @RETURN         ;START DOING THAT TEST AGAIN.
005440 000001                 PFTAB:    1
005442    003    002                    .BYTE   3,2
005444 001226                           TSTNO

                              .DELAY:
005446 012777 000020 173736             MOV     #20,@DMP04
005454 104414                           ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
005456 121111                           121111                  ;POKE CLOCK DELAY BIT
005460                        1$:
005460 104414                           ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
005462 121224                           121224                  ;PORT4←IBUS*11
005464 032777 000020 173720             BIT     #BIT4,@DMP04    ;IS CLOCK BIT SET?
005472 001772                           BEQ     1$              ;BR IF NO
005474 000002                           RTI

005476                        .MSTCLR:
005476 152777 000100 173702             BISB    #BIT6,@DMCSRH   ;SET MASTER CLEAR
005504 142777 000300 173674             BICB    #BIT6!BIT7,@DMCSRH  ;CLEAR MASTER CLEAR AND RUN
005512 000002                           RTI                     ;RETURN

005514                        .ROMCLK:
005514 152777 000002 173664             BISB    #BIT1,@DMCSRH   ;SET ROMI
005522 013677 173666                    MOV     @(SP)+,@DMP06   ;LOAD INSTRUCTION IN SEL6
005526 062746 000002                    ADD     #2,-(SP)        ;ADJUST STACK
005532 032777 000100 173442             BIT     #SW06,@sWR      ;HALT IF SW06 =1
005540 001401                           BEQ     1$              ;BR IF SW06 =0
005542 000000                           HALT                    ;HALT BEFORE CLOCKING INSTRUCTION
005544 152777 000003 173634   1$:       BISB    #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
005552 142777 000007 173626             BICB    #BIT2!BIT1!BIT0,@DMCSRH  ;CLEAR ROMO, ROMI, STEP
```

```
 1159  005560  000002                          RTI
 1160
 1161  005562                        .DATACLK:
 1162  005562  013637  001416                  MOV     @(SP)+,TEMP     ;PUT TICK COUNT IN TEMP
 1163  005566  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
 1164  005572  152777  000020  173606  1$:     BISB    #BIT4,@DMCSRH   ;SET STEP LU
 1165  005600  027777  173600  173576          CMP     @DMCSR,@DMCSR   ;WASTE TIME
 1166  005606  142777  000020  173572          BICB    #BIT4,@DMCSRH   ;CLEAR STEP LU
 1167  005614  005337  001416                  DEC     TEMp            ;DEC TICK COUNT
 1168  005620  001364                          BNE     1$              ;BR IF NOT DONE
 1169  005622  000002                          RTI                     ;RETURN
 1170  005624  000001                3$:       .BLKW 1
 1171
 1172  005626                        .TIMER:
 1173  005626  013637  001416                  MOV     @(SP)+,TEMP     ;MOVE COUNT TO TEMP
 1174  005632  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
 1175  005636                        1$:
 1176  005636  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1177  005640  021364                          021364                  ;PORT4←IBUS# REG11
 1178  005642  032777  000002  173542          BIT     #2,@DMP04       ;IS PGM CLOCK BIT CLEAR?
 1179  005650  001772                          BEQ     1$              ;BR IF YES
 1180  005652                        2$:
 1181  005652  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1182  005654  021364                          021364                  ;PORT4←IBUS# REG11
 1183  005656  032777  000002  173526          BIT     #2,@DMP04       ;IS PGM CLOCK BIT SET?
 1184  005664  001372                          BNE     2$              ;BR IF YES
 1185  005666  005337  001416                  DEC     TEMp            ;DEC COUNT
 1186  005672  001361                          BNE     1$              ;BR IF NOT DONE
 1187  005674  000002                          RTI                     ;RETURN
 1188
 1189  005676  020040  000077        MQM:      .ASCIZ  /  ?/
  (2)  005702  005015     000        MCRLF:    .ASCIZ  <15><12>
  (2)  005705     377  053520  020122 MPFAIL:   .ASCIZ  <377>/PWR FAILED. RESTART AT TESt /
  (2)  005743     377  047105  020104 MEPASS:   .ASCIZ  <377>/END PASS DRLPM  /
  (2)  005765     377  000122        MR:       .ASCIZ  <377>/R/
  (2)  005770  047377  020117  042504 MERR2:    .ASCIZ  <377>/NO DEVICES PRESENT./
  (2)  006015     377  047111  052523 MERR3:    .ASCIZ  <377>/INSUFFICIENT DATA!/
  (2)  006041     377  042524  052123 MTSTPC:   .ASCIZ  <377>/TEST PC-/
  (2)  006053     377  047514  045503 MLOCK:    .ASCIZ  <377>/LOCK ON SELECTED TESt/
  (2)  006102  051503  035122  000040 MCSRX:    .ASCIZ  /CSR: /
  (2)  006110  042526  035103  000040 MVECX:    .ASCIZ  /VEC: /
  (2)  006116  040520  051523  051505 MPASSX:   .ASCIZ  /PASSES: /
  (2)  006127     105  051122  051117 MERRX:    .ASCIZ  /ERRORS: /
  (2)  006140  042524  052123  047040 MTSTN:    .ASCIZ  /TEST NO: /
  (2)  006152  000052                MASTEK:    .ASCIZ  /*/
  (2)  006154  051777  052105  051440 MNEW:     .ASCIZ  <377>/SET SWITCH REG TO M8200-YC'S DESiRED ACTIVE./
  (2)  006232  041520  020072     000 MERRPC:   .ASCIZ  /PC: /
  (2)  006237     212  020040  020040 XHEAD:    .ASCII  <212>/            MAP OF M8200-YC STATUS/
  (2)  006301     377  020040  020040          .ASCII  <377>/            --------------------/
  (2)  006340  020212  050040  020103          .ASCII  <212>/  PC      CSR     STAT1    STAT2    STAT3/
  (2)  006412  026777  026455  026455          .ASCIZ  <377>/------   ------   ------   ------   ------/
  (2)  006466  044377  053517  046440 NUM:     .ASCIZ  <377>/HOW MANY M8200-YC'S TO BE TESiED?/
  (2)  006531     377  051503  020122 CSR:     .ASCIZ  <377>/CSR ADDRESS?/
  (2)  006547     377  042526  052103 VEC:     .ASCIZ  <377>/VECTOR ADDRESS?/
```

```
   (2)  006570  041377  020122  051120  PRIO:    .ASCIZ  <377>/BR PRIORITY LEVEL? (4,5,6,7)?/
   (2)  006627     377  043111  042040  CRAM:    .ASCIZ  <377>/IF DMC HAS CRAM (M8204) TYPE "Y", IF CROM (M8200) TYPE "N" ?/
   (2)  006725     377  044127  041511  MODU:    .ASCIZ  <377>/WHICH LINE UNIT? IF NONE TYPE "N", IF M8201 TYPE "1", IF M8202 TYP
   (2)  007037     377  053523  052111  LINE:    .ASCIZ  <377>/SWITCH PAC#1 (DDCMP LINE #)?/
   (2)  007075     377  053523  052111  BM:      .ASCIZ  <377>/SWITCH PAC#2 (BM873 BOOT ADD)?/
   (2)  007135     377  051511  052040  CONN:    .ASCIZ  <377>/IS THE LOOP BACK CONNECTOR ON?/
   (2)  007175     377  047516  042040  NOACT:   .ASCIZ  <377>/NO DEVICES ARE SELECTED/
   (2)  007226  005377  053523  036522  SWMES:   .ASCIZ  <377><12>/SWR= /
   (2)  007236  042516  037527  000040  SWMES1:  .ASCIZ  /NEW? /
   (2)  007244  177777  034115  030062  CONERR:  .ASCIZ  <377><377>/M8200-YC FOUND AT NON-STANDARD ADDRESS  PC: /
   (2)  007323     377  054105  042520  CNERR:   .ASCIZ  <377>/EXPECTED  FOUND/
   (2)  007344  024040  034115  030062  DMCM:    .ASCIZ  / (M8200-YC) /
   (2)  007361     040  045450  041515  KMCM:    .ASCIZ  / (KMC) /
   (2)  007371     377  034115  030062  SPEED:   .ASCIZ  <377>/M8200-YC-AR(REMOTE,LOW SPEED) OR M8200-YC-AL(LOCAL,HIGH SPEED) TYP
   (2)          007514                .EVEN
   (2)  007514  000005                XSTATQ:  5
1190    007516     006     003                 .BYTE   6,3
1191    007520  001246                         TEMP1
1192    007522     006     003                 .BYTE   6,3
1193    007524  001250                         TEMP2
1194    007526     006     003                 .BYTE   6,3
1195    007530  001252                         TEMP3
1196    007532     006     003                 .BYTE   6,3
1197    007534  001254                         TEMP4
1198    007536     006     002                 .BYTE   6,2
1199    007540  001256                         TEMp5
1200                                .EVEN
1201
1202                                        ;BUFFERS FOR INPUT-OUTPUT
1203
1204    007542  000000                INBUF:  0
1205            007604                .=.+40
1206    007604  000000                MDATA:  0
1207            007646                .=.+40
1208
1209
1210                                        ;ROUTINE USED TO CHANGE SOFTWARE SWITCH
1211                                        ;REGISTER USING THE CONSOLE TERmINAL
1212                                        ;---------------------------------------
1213
1214    007646  022737  000176  001202  CKSWR:  CMP     #SWREG,SWR      ;IS THE SOFT SWR BEING USED?
1215    007654  001077                         BNE     CKSWR5          ;BR IF NO
1216    007656  105777  171322                 TSTB    @TKCSR          ;IS DONE SET?
1217    007662  100003                         BPL     2$              ;GO ON IF NOT SET
1218    007664  012737  177777  003744         MOV     #-1,DONE        ;IF DONE SET, SET FLAG
1219    007672  022777  000007  171306  2$:     CMP     #7,@TKDBR       ;WAS CTRL G TYPED? (7 BIT ASCII)
1220    007700  001404                         BEQ     1$              ;BR IF YES
1221    007702  022777  000207  171276         CMP     #207,@TKDBR     ;WAS CTRL G TYPED? (8 BIT ASCII)
1222    007710  001061                         BNE     CKSWR5          ;BR IF NO
1223    007712  010246                 1$:     MOV     R2,-(SP)        ;STORE R2
1224    007714  010346                         MOV     R3,-(SP)        ;STORE R3
1225    007716  010446                         MOV     R4,-(SP)        ;STORE R4
1226    007720  012737  177777  010056         MOV     #-1,SWFLG       ;SET SOFT TYPE OUT FLAG
1227    007726  005002                 CKSWR1: CLR     R2              ;CLEAR NEW SWR CONTENTS
```

```
1228  007730  012704  177777                    MOV     #-1,R4          ;SET FLAG TO ALL ONES
1229  007734  104402  007226                    TYPE    ,SWMES          ;TYPE "SWR= "
1230  007740  104411                  CKSWR2:   CNVRT                   ;TYPE OUT PRESENT CONTENTS
1231  007742  010112                            SOFTSW                  ;OF SOFT SWITCH REGISTER
1232  007744  104402  007236          CKSWR3:   TYPE    ,SWMES1         ;TYPE "NEW? "
1233  007750  004737  010060          CKSWR4:   JSR     PC,INCHAR       ;GET RESPONSE
1234  007754  022703  000015                    CMP     #15,R3          ;WAS IT A CR?
1235  007760  001424                            BEQ     5$              ;BR IF YES
1236  007762  022703  000012                    CMP     #12,R3          ;WAS IT A LF?
1237  007766  001416                            BEQ     4$              ;BR IF YES
1238  007770  022703  000025                    CMP     #25,R3          ;WAS IT CTRL U?
1239  007774  001754                            BEQ     CKSWR1          ;BR IF YES(START OVER)
1240  007776  022703  000007                    CMP     #7,R3           ;IF CNTL G GET NEXT CHAR
1241  010002  001762                            BEQ     CKSWR4
1242  010004  005004                            CLR     R4              ;IT MUST BE A DIGIT SO CLR FLAG
1243  010006  042703  177770                    BIC     #177770,R3      ;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
1244  010012  006302                            ASL     R2              ;SHIFT R2 3 TIMES
1245  010014  006302                            ASL     R2
1246  010016  006302                            ASL     R2
1247  010020  050302                            BIS     R3,R2           ;ADD LAST DIGIT
1248  010022  000752                            BR      CKSWR4          ;GET NEXT CHARACTER
1249  010024  012766  002002  000006  4$:       MOV     #.START,6(SP)   ;LF WAS TYPED SO GO TO START
1250  010032  005704          5$:       TST     R4              ;IS FLAG CLEAR?
1251  010034  001002                            BNE     6$              ;IF NOT DON'T CHANGE SOFT SWR
1252  010036  010277  171140                    MOV     R2,@SWR         ;IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
1253  010042  005037  010056          6$:       CLR     SWFLG           ;CLEAR TYPEOUT FLAG
1254  010046  012604                            MOV     (SP)+,R4        ;RESTORE R4
1255  010050  012603                            MOV     (SP)+,R3        ;RESTORE R3
1256  010052  012602                            MOV     (SP)+,R2        ;RESTORE R2
1257  010054  000207                  CKSWR5:   RTS     PC              ;RETURN
1258
1259  010056  000000                  SWFLG:    0
1260
1261  010060  105777  171120          INCHAR:   TSTB    @TKCSR
1262  010064  100375                            BPL     .-4
1263  010066  017703  171114                    MOV     @TKDBR,R3
1264  010072  105777  171112                    TSTB    @TPCSR
1265  010076  100375                            BPL     .-4
1266  010100  010377  171106                    MOV     R3,@TPDBR
1267  010104  042703  000200                    BIC     #BIT7,R3
1268  010110  000207                            RTS     PC
1269
1270  010112  000001                  SOFTSW:   1
1271  010114     006      002                   .BYTE   6,2
1272  010116  000176                            SWREG
```

```
1273
1274                                        ;
1275                                        ;ROUTINE USED TO "CYCLE" THROUGH UP TO 16 M8200-YC'S
1276                                        ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
1277                                        ;AND RUNS THE SPECIFIED M8200-YC'S.   THIS ROUTINE *MUST*
1278                                        ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
1279                                        ;SETUP NECESSARY.
1280                                        ;
1281
1282  010120  005737  001306        CYCLE:  TST     DMACTV          ;ARE ANY M8200-YC'S TO BE TESTED?
1283  010124  001004                        BNE     1$              ;BR IF OK.
1284  010126  104402  007175                TYPE    ,NOACT          ;NO M8200-YC'S SELECTED!!
1285  010132  000000                        HALT                    ;STOP THE SHOW.
1286  010134  000776                        BR      .-2             ;DISQUALIFY CONT. SW.
1287  010136  000241                1$:     CLC                     ;CLEAR PROC. CARRY BIT.
1288  010140  006137  001316                ROL     RUN             ;UPDATE POINTER
1289  010144  005537  001316                ADC     RUN             ;CATCH CARRY FROM RUN
1290  010150  062737  000004  001322        ADD     #4,MILK         ;UPDATE POINTER
1291  010156  062737  000010  001320        ADD     #10,CREAM       ;UPDATE ADDRESS POINTER.
1292  010164  022737  001700  001320        CMP     #DM.MAP+200,CREAM
1293  010172  001006                        BNE     2$              ;KEEP GOING; NOT ALL TESTED FOR.
1294  010174  012737  001500  001320        MOV     #DM.MAP,CREAM   ;RESET ADDRESS POINTER.
1295  010202  012737  001702  001322        MOV     #CNT.MAP,MILK   ;RESET PASS COUNT POINTER
1296  010210  033737  001316  001306  2$:    BIT     RUN,DMACTV      ;IS THIS ONE ACTIVE?
1297  010216  001747                        BEQ     1$              ;BR IF NO
1298  010220  013700  001320                MOV     CREAM,R0        ;GET ADDRESS POINTER
1299  010224  013702  001322                MOV     MILK,R2         ;GET PASS COUNT POINTER
1300  010230  012037  001404                MOV     (R0)+,DMCSR     ;LOAD SYSTEM CTRL. REG
1301  010234  011037  001374                MOV     (R0),DMRVEC     ;LOAD VECTOR
1302  010240  042737  177000  001374        BIC     #177000,DMRVEC  ;CLEAR UNWANTED BITS
1303  010246  012037  001366                MOV     (R0)+,STAT1     ;LOAD STAT1
1304  010252  012037  001370                MOV     (R0)+,STAT2     ;LOAD STAT2
1305  010256  012037  001372                MOV     (R0)+,STAT3     ;LOAD STAT3
1306  010262  012237  001230                MOV     (R2)+,PASCNT    ;LOAD PASS COUNT
1307  010266  012237  001232                MOV     (R2)+,ERRCNT    ;LOAD ERROR COUNT
1308  010272  012700  000002                MOV     #2,R0           ;SAVE CORE THIS WAY!
1309  010276  013737  001404  001406        MOV     DMCSR,DMCSRH
1310  010304  005237  001406                INC     DMCSRH
1311  010310  013737  001406  001410        MOV     DMCSRH,DMCTL
1312  010316  005237  001410                INC     DMCTL
1313  010322  013737  001410  001412        MOV     DMCTL,DMP04
1314  010330  060037  001412                ADD     R0,DMP04
1315  010334  013737  001412  001414        MOV     DMP04,DMP06
1316  010342  060037  001414                ADD     R0,DMP06
1317
1318  010346  013737  001374  001376        MOV     DMRVEC,DMR1VL   ;PTY LVL
1319  010354  060037  001376                ADD     R0,DMRLVL       ;
1320  010360  013737  001376  001400        MOV     DMRLVL,DMTVEC   ;TX VEC
1321  010366  060037  001400                ADD     R0,DMTVEC       ;
1322  010372  013737  001400  001402        MOV     DMTVEC,DMTLVL   ;TX LVL
1323  010400  060037  001402                ADD     R0,DMTLVL
1324
1325  010404  032737  000002  001236        BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED
1326  010412  001450                        BEQ     7$              ;BR IF NO
```

```
1327  010414                          4$:
1328  010414  005737  000042                  TST     @#42            ;RUNNING IN AUTO MODE?
1329  010420  001045                          BNE     7$              ;BR IF YES
1330  010422  104402  005702                  TYPE    ,MCRLF
1331  010426  104403                          INSTR
1332  010430  006140                          MTSTN                   ;GET TEST NO.
1333  010432  104405                          PARAM
1334  010434  000001                          1
1335  010436  001000                          1000
1336  010440  001226                          TSTNO
1337  010442     000                  .BYTE   0
1338  010443     001                  .BYTE   1
1339  010444  012700  016376                  MOV     #TST1,R0
1340  010450  022710                  5$:     CMP     (PC)+,(R0)      ;CMP FIRST WORD TO 12737
1341  010452  012737                          MOV     (PC)+,@(PC)+
1342  010454  001020                          BNE     6$              ;BR IF NOT SAME
1343  010456  023760  001226  000002          CMP     TSTNO,2(R0)     ;DOES TSTNO MATCH?
1344  010464  001014                          BNE     6$              ;BR IF NO
1345  010466  022760  001226  000004          CMP     #TSTNO,4(R0)    ;IS LAST WORD OK?
1346  010474  001010                          BNE     6$              ;BR IF NO
1347  010476  010037  001214                  MOV     R0,RETURN       ;IT IS A LEGAL TEST SO DO IT
1348  010502  104402  005765                  TYPE    ,MR
1349  010506  042737  000002  001236          BIC     #SW01,STRTSW
1350  010514  000412                          BR      8$
1351  010516  005720                  6$:     TST     (R0)+           ;POP R0
1352  010520  020027  022106                  CMP     R0,#TLAST+10    ;AT END YET?
1353  010524  001351                          BNE     5$              ;BR IF NO
1354  010526  104402  005676                  TYPE    ,MQM            ;YES ILLEGAL TEST NO.
1355  010532  000730                          BR      4$              ;TRY AGAIN
1356
1357  010534  012737  016376  001214  7$:     MOV     #TST1,RETURN    ;PREPARE RETURN ADDRESS
1358  010542  013701  001404          8$:     MOV     DMCSR,R1        ;R1 = BASE M8200-YC ADDRESS
1359  010546  000177  170442                  JMP     @RETURN         ;GO START TESTING.
1360
1361
1362                                          ;ROUTINE USED TO "AUTO SIZE" THE M8200-YC
1363                                          ;CSR AND VECTOR.
1364                                          ;NOTE:   THE CSR MAY BE ANY WHERE IN THE
1365                                          ;        ADDRESS RANGE (170440:170510)
1366                                          ;        AND THE VECTOR MAY BE ANY WHERE IN THE
1367                                          ;        FLOATING VECTOR RANGE (300:770)
1368                                          ;
1369
1370  010552                          AUTO.SIZE:
1371  010552  000005                          RESET                   ;INSURE A BUS INIT.
1372  010554  012702  001500          CSRMAP: MOV     #DM.MAP,R2      ;LOAD MAP POINTER.
1373  010560  005022                  1$:     CLR     (R2)+           ;ZERO ENTIRE MAP
1374  010562  022702  001700                  CMP     #DM.END,R2      ;ALL DONE?
1375  010566  001374                          BNE     1$              ;BR IF NO
1376  010570  005037  001310                  CLR     DMNUM           ;SET OCTAL NUMBER OF M8200-YC'S TO 0
1377  010574  012702  001500                  MOV     #DM.MAP,R2      ;R2 POINTS TO M8200-YC MAP
1378  010600  005037  001306                  CLR     DMACTV          ;CLEAR ACTIVE
1379  010604  032737  000001  001236          BIT     #SW00,STRTSW    ;QUESTIONS?
1380  010612  001002                          BNE     .+6             ;BR IF YES
```

```
1381  010614  000137  011322            JMP     7$              ;IF NO SKIP QUESTIONS
1382  010620  012737  000001  001256    MOV     #1,TEMP5        ;START WITH 1
1383  010626  104403                    INSTR
1384  010630  006466                    NUM
1385  010632  104405                    PARAM
1386  010634  000001                    1
1387  010636  000020                    16.
1388  010640  001252                    TEMP3
1389  010642     000                    .BYTE   0
1390  010643     001                    .BYTE   1
1391  010644  013737  001252  001310    MOV     TEMP3,DMNUM     ;DMNUM = HOW MANY
1392  010652  104402  005702        12$: TYPE    ,MCRLF
1393  010656  104410                    CONVRT                  ;TYPE WHICH DMC IS BEING DONE
1394  010660  012054                    WHICH                   ;TEMP5 IS WHICH DMC
1395  010662  005237  001256            INC     TEMP5
1396  010666  104403                    INSTR
1397  010670  006531                    CSR
1398  010672  104405                    PARAM
1399  010674  170440                    170440
1400  010676  170510                    170510
1401  010700  001254                    TEMP4
1402  010702     000                    .BYTE   0
1403  010703     001                    .BYTE   1
1404  010704  013722  001254            MOV     TEMP4,(R2)+     ;STORE CSR IN MAP
1405  010710  104403                    INSTR
1406  010712  006547                    VEC
1407  010714  104405                    PARAM
1408  010716  000000                    0
1409  010720  000776                    776
1410  010722  001254                    TEMP4
1411  010724     000                    .BYTE   0
1412  010725     001                    .BYTE   1
1413  010726  013712  001254            MOV     TEMP4,(R2)      ;STORE VECTOR IN MAP
1414  010732  104402              10$:  TYPE
1415  010734  006570                    PRIO                    ;ASK WHAT BR LEVEL
1416  010736  004737  012340            JSR     PC,INTTY        ;GET RESPONSE
1417  010742  022703  000024            CMP     #24,R3          ;
1418  010746  101014                    BHI     50$             ;BR IF LESS THAN 4
1419  010750  022703  000027            CMP     #27,R3          ;
1420  010754  103411                    BLO     50$             ;BR IF GREATER THAN 7
1421  010756  012704  000011            MOV     #11,R4          ;R4 = NUMBER OF SHIFTS
1422  010762  006303                    ASL     R3              ;SHIFT R3 LEFT
1423  010764  005304                    DEC     R4              ;DEC SHIFT COUNT
1424  010766  001375                    BNE     .-4             ;BR IF NOT DONE
1425  010770  042703  170777            BIC     #170777,R3      ;BIC UNWANTED BITS
1426  010774  050312                    BIS     R3,(R2)         ;PUT BR LEVEL IN STATUS MAP
1427  010776  000403                    BR      8$              ;CONTINUE
1428  011000  104402              50$:  TYPE
1429  011002  005676                    MQM                     ;RESPONSE IS OUT OF LIMITS
1430  011004  000752                    BR      10$             ;TRY AGAIN
1431  011006                       8$:
1432  011006  000137  011300            JMP     33$
1433  011012  104402                    TYPE
1434  011014  006627                    CRAM                    ;DOES DMC HAVE CRAM?
```

```
1435  011016  004737  012340                    JSR     PC,INTTY        ;GET REPLY
1436  011022  022703  000131                    CMP     #131,R3
1437  011026  001427                            BEQ     9$              ;YES
1438  011030  022703  000116                    CMP     #116,R3         ;NO
1439  011034  001403                            BEQ     40$             ;NOT A Y OR N
1440  011036  104402                            TYPE
1441  011040  005676                            MQM                     ;TYPE "?"
1442  011042  000761                            BR      8$              ;ASK AGAIN
1443  011044  104402                    40$:    TYPE
1444  011046  007371                            SPEED                   ;M8200-YC-AR OR M8200-YC-AL?
1445  011050  004737  012340                    JSR     PC,INTTY        ;GET RESPONSE
1446  011054  022703  000122                    CMP     #122,R3         ;IS IT R
1447  011060  001414                            BEQ     16$             ;BR IF REMOTE
1448  011062  022703  000114                    CMP     #114,R3         ;IS IT L
1449  011066  001403                            BEQ     41$             ;BR IF LOCAL
1450  011070  104402                            TYPE
1451  011072  005676                            MQM
1452  011074  000763                            BR      40$             ;TRY AGAIN
1453  011076  052762  000002  000004    41$:    BIS     #BIT1,4(R2)     ;SET BIT1 IN STAT3
1454  011104  000402                            BR      16$             ;CONTINUE
1455  011106  052712  100000            9$:     BIS     #BIT15,(R2)     ;SET BIT 15 IF CRAM
1456  011112  104402                    16$:    TYPE
1457  011114  006725                            MODU                    ;ASK WHICH LINE UNIT
1458  011116  004737  012340                    JSR     PC,INTTY        ;GET REPLY
1459  011122  022703  000021                    CMP     #21,R3          ;"1"
1460  011126  001417                            BEQ     30$
1461  011130  022703  000022                    CMP     #22,R3          ;"2"
1462  011134  001412                            BEQ     31$
1463  011136  022703  000116                    CMP     #116,R3         ;"N"
1464  011142  001403                            BEQ     32$
1465  011144  104402                            TYPE
1466  011146  005676                            MQM                     ;IF NOT A 1,2 OR N TYPE "?"
1467  011150  000760                            BR      16$             ;TRY AGIAN
1468  011152  052722  010000            32$:    BIS     #BIT12,(R2)+    ;SET BIT 12 IN STAT2 IF NO LU
1469  011156  022222                            CMP     (R2)+,(R2)+     ;POP OVER STAT2 AND STAT3
1470  011160  000447                            BR      33$
1471  011162  052712  020000            31$:    BIS     #BIT13,(R2)     ;SET BIT 13 IN STAT2 IF M8202
1472  011166  104402                    30$:    TYPE
1473  011170  007135                            CONN                    ;ASK IF LOOP-BACK IS ON
1474  011172  004737  012340                    JSR     PC,INTTY        ;GET REPLY
1475  011176  022703  000131                    CMP     #131,R3         ;Y
1476  011202  001406                            BEQ     17$
1477  011204  022703  000116                    CMP     #116,R3         ;N
1478  011210  001406                            BEQ     18$
1479  011212  104402                            TYPE
1480  011214  005676                            MQM                     ;IF NOT Y OR N TYPE "?"
1481  011216  000763                            BR      30$             ;TRY AGAIN
1482  011220  052722  040000            17$:    BIS     #BIT14,(R2)+    ;TURNAROUND IS CONNECTED
1483  011224  000402                            BR      19$
1484  011226  042722  040000            18$:    BIC     #BIT14,(R2)+    ;NO TURNAROUND
1485  011232                            19$:
1486  011232  104403                            INSTR
1487  011234  007037                            LINE
1488  011236  104405                            PARAM
```

```
 1489  011240  000000                          0
 1490  011242  000377                          377
 1491  011244  001254                          TEMP4
 1492  011246     000                          .BYTE   0
 1493  011247     001                          .BYTE   1
 1494  011250  113722  001254                  MOVB    TEMP4,(R2)+     ;STORE SWITCH PAC IN MAP
 1495  011254  104403                          INSTR
 1496  011256  007075                          BM
 1497  011260  104405                          PARAM
 1498  011262  000000                          0
 1499  011264  000377                          377
 1500  011266  001254                          TEMP4
 1501  011270     000                          .BYTE   0
 1502  011271     001                          .BYTE   1
 1503  011272  113722  001254                  MOVB    TEMP4,(R2)+     ;STORE SWITCH PAC IN MAP
 1504  011276  005722                          TST     (R2)+           ;POP OVER STAT3
 1505  011300                          33$:
 1506  011300  062702  000006                  ADD     #6,R2
 1507  011304  005337  001252                  DEC     TEMP3           ;DEC DMC COUNT
 1508  011310  001402                          BEQ     34$             ;BR IF DONe
 1509  011312  000137  010652                  JMP     12$             ;JUMP IF NOT
 1510  011316  000137  011754          34$:    JMP     13$             ;CONTINUE
 1511  011322  012701  170440          7$:     MOV     #170440,R1      ;SET FOR FIRST ADDRESS TO BE TESTED
 1512  011326  012737  012046  000004          MOV     #6$,@#4         ;SET FOR NON-EXISTANT DEVICE TIME OUT
 1513  011334  005011                  2$:     CLR     (R1)            ;CLEAR SEL0
 1514  011336  005711                          TST     (R1)            ;IF M8200-YC DMCSR S/B 0
 1515  011340  001173                          BNE     3$              ;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO M8200-YC
 1516  011342  005061  000006                  CLR     6(R1)           ;CLEAR SEL6
 1517  011346  000424                          BR      21$
 1518  011350  005761  000006                  TST     6(R1)           ;IF M8200-YC THEN DMRIC S/B =0!
 1519  011354  001165                          BNE     3$              ;BR IF NOT M8200-YC
 1520  011356  012711  002000                  MOV     #BIT10,(R1)     ;SET ROM0
 1521  011362  005061  000004                  CLR     4(R1)           ;CLEAR SEL4
 1522  011366  012761  125252  000006          MOV     #125252,6(R1)   ;WRITE THIS TO SEL6
 1523  011374  052711  020000                  BIS     #BIT13,(R1)     ;WRITE IT!
 1524  011400  022761  125252  000004          CMP     #125252,4(R1)   ;WAS IT WRITTEN?
 1525  011406  001004                          BNE     21$             ;IF NO IT IS NOT CRAM
 1526  011410  052762  100000  000002          BIS     #BIT15,2(R2)    ;SET BIT15 IF CRAM
 1527  011416  000431                          BR      22$
 1528  011420  012711  001000          21$:    MOV     #BIT9,(R1)      ;SET ROMI
 1529  011424  012761  100400  000006          MOV     #100400,6(R1)   ;PUT INSTRUCTION IN SEL6
 1530  011432  012711  001400                  MOV     #BIT9!BIT8,(R1) ;CLOCK INSTRUCTION (MICRO PROC PC TO 0)
 1531  011436  012711  002000                  MOV     #BIT10,(R1)     ;SET ROM0
 1532  011442  022761  000456  000006          CMP     #456,6(R1)      ;IS IT LOCAL CROM
 1533  011450  001411                          BEQ     23$             ;BR IF YES
 1534  011452  022761  016520  000006          CMP     #16520,6(R1)    ;IS IT REMOTE CROM?
 1535  011460  001410                          BEQ     22$             ;BR IF YES
 1536  011462  022761  177777  000006          CMP     #-1,6(R1)       ;NO CROM?
 1537  011470  001404                          BEQ     22$             ;BR IF YES
 1538  011472  000516                          BR      3$              ;NOT A DMC
 1539  011474  052762  000002  000006  23$:    BIS     #BIT1,6(R2)     ;SET BIT 1 IN STAT3
 1540                          ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A M8200-YC CSR ADDRESS.
 1541  011502  010122                  22$:    MOV     R1,(R2)+        ;STORE CSR IN CORE TABLE.
 1542  011504  012711  001000          15$:    MOV     #BIT9,(R1)      ;CLEAR LINE UNIT LOOP
```

```
1543  011510  005061  000004                CLR     4(R1)           ;CLEAR PORT4
1544  011514  012761  122113  000006        MOV     #122113,6(R1)   ;LOAD INSTRUCTION (CLR DTR)
1545  011522  052711  000400                BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
1546  011526  012761  021264  000006        MOV     #021264,6(R1)   ;LOAD INSTRUCTION
1547  011534  052711  000400                BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
1548  011540  122761  000377  000004        CMPB    #377,4(R1)      ;IS IT ALL ONES?
1549  011546  001003                        BNE     .+10            ;BR IF NO
1550  011550  052712  010000                BIS     #BIT12,(R2)     ;IF YES, NO LINE UNIT, SET STATUS BIT
1551  011554  000436                        BR      20$
1552  011556  032761  000002  000004        BIT     #BIT1,4(R1)     ;IS SWITCH A ONE?
1553  011564  001403                        BEQ     .+10            ;BR IF M8201
1554  011566  052712  060000                BIS     #BIT13!BIT14,(r2) ;M8202 ASSUME CONNECTOR
1555  011572  000427                        BR      20$             ;CONNECTOR ON)
1556  011574  032761  000010  000004        BIT     #BIT3,4(R1)     ;IS MRDY SET
1557  011602  001023                        BNE     20$             ;BR IF M8201 NO CONNECTOR (ON LINE)
1558  011604  012761  000100  000004        MOV     #BIT6,4(R1)     ;LOAD PORT4
1559  011612  012761  122113  000006        MOV     #122113,6(R1)   ;LOAD INSTRUCTION
1560  011620  052711  000400                BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(SET DTR)
1561  011624  012761  021264  000006        MOV     #021264,6(R1)   ;LOAD INSTRUCTION
1562  011632  052711  000400                BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(READ MODEM REG)
1563  011636  032761  000010  000004        BIT     #BIT3,4(R1)     ;IS MRDY SET NOW?
1564  011644  001402                        BEQ     20$             ;BR IF NO CONNECTOR
1565  011646  052712  040000                BIS     #BIT14,(R2)     ;SET STATUS BIT FOR CONNECTOR
1566  011652  005722                20$:    TST     (R2)+           ;POP POINTER
1567  011654  012761  021324  000006        MOV     #021324,6(R1)   ;PUT INSTRUCTION IN PORT6
1568  011662  012711  001400                MOV     #BIT9!BIT8,(R1) ;PORT4=LU 15
1569  011666  156122  000004                BISB    4(R1),(R2)+     ;STORE DDCMP LINE # IN TABLE
1570  011672  012761  021344  000006        MOV     #021344,6(R1)   ;PORT6=INSTRUCTION
1571  011700  012711  001400                MOV     #BIT8!BIT9,(R1) ;CLOCK INSTR.
1572  011704  156122  000004                BISB    4(R1),(R2)+     ;STORE BM873 ADD IN TABLE
1573  011710  005722                        TST     (R2)+           ;POP OVER STAT3
1574  011712  005011                        CLR     (R1)            ;CLEAR ROMI
1575  011714  005237  001310                INC     DMNUM           ;UPDATE DEVICE COUNTER
1576  011720  022737  000020  001310        CMP     #20,DMNUM       ;ARE MAX. NO. OF DEV FOUND?
1577  011726  001412                        BEQ     13$             ;YES DON'T LOOK FOR ANY MORE.
1578  011730  005011                3$:     CLR     (R1)            ;CLEAR BIT 10
1579  011732  005061  000006                CLR     6(R1)           ;CLEAR SEL 6
1580  011736  062701  000010        14$:    ADD     #10,R1          ;UPDATE CSR POINTER ADDRESS
1581  011742  022701  170510                CMP     #170510,R1
1582  011746  001402                        BEQ     13$             ;BR IF DONE
1583  011750  000137  011334                JMP     2$              ;JUMP IF NOT
1584  011754  005037  001306        13$:    CLR     DMACTV
1585  011760  005737  001310                TST     DMNUM           ;WERE ANY M8200-YC'S FOUND AT ALL?
1586  011764  001423                        BEQ     5$              ;ERROR AUTO SIZER FOUND NO M8200-YC'S IN THIS SYS.
1587  011766  013701  001310                MOV     DMNUM,R1
1588  011772  010137  001314                MOV     R1,SAVNUM       ;SAVE NUMBER OF DEVICES
1589  011776  000241                4$:     CLC
1590  012000  006137  001306                ROL     DMACTV          ;GENERATE ACTIVE REGISTER OF DEVICES.
1591  012004  005237  001306                INC     DMACTV          ;SET THE BIT
1592  012010  005301                        DEC     R1
1593  012012  001371                        BNE     4$              ;BR IF MORE TO GENERATE
1594  012014  012737  000006  000004        MOV     #6,@#4          ;RESTORE TRAP VECTOR
1595  012022  013737  001306  001312        MOV     DMACTV,SAVACT   ;SAVE ACTIVE REGISTER
1596  012030  000137  012062                JMP     VECMAP          ;GO FIND THE VECTOR NOW.
```

```
1597  012034  104402  005770          5$:     TYPE    ,MERR2          ;NOTIFY OPR THAT NO M8200-YC'S FOUND.
1598  012040  005000                          CLR     R0              ;MAKE DATA LIGHTS ZERO
1599  012042  000000                          HALT                    ;STOP THE SHOW
1600  012044  000776                          BR      .-2             ;DISABLE CONT. SW.
1601  012046  012716  011736          6$:     MOV     #14$,(SP)       ;ENTERED BY NON-EXISTANT TIME-OUT.
1602  012052  000002                          RTI                     ;RETURN TO MAINSTREAM
1603
1604  012054  000001                  WHICH:  1
1605  012056      002      002                .BYTE   2,2
1606  012060  001256                          TEMP5
1607
1608  012062  032737  000001  001236  VECMAP: BIT     #SW00,STRTSW
1609  012070  001114                          BNE     5$
1610  012072  012737  000340  000022          MOV     #340,@#22       ;SET IOT TRAP PRIO TO 7
1611  012100  012737  012254  000020          MOV     #4$,@#20        ;SET IOT TRAP VECTOR
1612  012106  012702  001500                  MOV     #DM.MAP,R2      ;SET SOFTWARE POINTER
1613  012112  012700  000300                  MOV     #300,R0         ;FLOATING VECTORS START HERE.
1614  012116  012701  000302                  MOV     #302,R1         ;PC OF IOT INSTR.
1615  012122  010120                  1$:     MOV     R1,(R0)+        ;START FILLING VECTOR AREA
1616  012124  012721  000004                  MOV     #4,(R1)+        ;WITH .+2; IOT
1617  012130  022021                          CMP     (R0)+,(R1)+     ;ADD 2 TO R0 +R1
1618  012132  020127  001000                  CMP     R1,#1000
1619  012136  101771                          BLOS    1$              ;BR IF MORE TO FILL
1620  012140  013737  001306  001246          MOV     DMACTV,TEMP1    ;STORE TEMPORALLY
1621  012146  006037  001246          2$:     ROR     TEMP1           ;BRING OUT A BIT
1622  012152  103063                          BCC     5$              ;BR IF ALL DONE
1623  012154  012704  000012                  MOV     #12,R4          ;R4 IS INDEX REGISTER
1624  012160  016437  012324  177776          MOV     BRLVL(R4),PS    ;SET PS TO 7
1625  012166  011201                          MOV     (R2),R1
1626  012170  012761  000200  000004          MOV     #200,4(R1)
1627  012176  012711  001000                  MOV     #BIT9,(R1)      ;SET ROMI
1628  012202  012761  121111  000006          MOV     #121111,6(R1)   ;PUT INSTRUCTION IN PORT6
1629  012210  012711  001400                  MOV     #BIT9!BIT8,(R1) ;FORCE AN INTERRUPT
1630  012214  105200                  7$:     INCB    R0              ;STALL
1631  012216  001376                          BNE     .-2             ;FOR TIME TO iNTERUPT
1632  012220  162704  000002                  SUB     #2,R4           ;GET NEXT LOWEST PS LEVEL
1633  012224  001404                          BEQ     6$              ;BR IF R4 = 0
1634  012226  016437  012324  177776          MOV     BRLVL(R4),PS    ;MOVE NEXT LOWER LEVEL IN PS
1635  012234  000767                          BR      7$              ;BR TO DELAY
1636  012236  052762  005300  000002  6$:     BIS     #5300,2(R2)     ;NO INTERUPT ASSUME 300 AT LEVEL 5 AND FIX M8200-YC LATE
1637  012244  005011                  3$:     CLR     (R1)            ;CLEAR ROMI
1638  012246  062702  000010                  ADD     #10,R2          ;POP SOFTWARE POINTER
1639  012252  000735                          BR      2$              ;KEEP GOING
1640  012254  051662  000002          4$:     BIS     (SP),2(R2)      ;GET VECTOR ADDRESS
1641  012260  042762  000007  000002          BIC     #7,2(R2)        ;CLEAR JUNK
1642  012266  016405  012326                  MOV     BRLVL+2(R4),R5  ;GET BR LEVEL OF M8200-YC
1643  012272  006305                          ASL     R5              ;SHIFT LEVEL 4 PLACES
1644  012274  006305                          ASL     R5              ;TO THE LEFT FOR THE
1645  012276  006305                          ASL     R5              ;STATUS TABLE
1646  012300  006305                          ASL     R5
1647  012302  042705  170777                  BIC     #170777,R5      ;CLEAR UNWANTED BITS
1648  012306  050562  000002                  BIS     R5,2(R2)        ;PUT BR LEVEL IN STATUS TABLE
1649  012312  022626                          CMP     (SP)+,(SP)+     ;POP IOT JUNK OFF STACK
1650  012314  012716  012244                  MOV     #3$,(SP)        ;SET FOR RETURN
```

```
1651  012320  000002                        RTI
1652  012322  000207              5$:       RTS     PC              ;ALL DONE WITH "AUTO SIZING"
1653
1654  012324  000000              BRLVL:    0       ;LEVEL 0
1655  012326  000000                        0       ;LEVEL 0
1656  012330  000200                        200     ;LEVEL 4
1657  012332  000240                        240     ;LEVEL 5
1658  012334  000300                        300     ;LEVEL 6
1659  012336  000340                        340     ;LEVEL 7
1660
1661
1662  012340  105777  166640      INTTY:    TSTB    @TKCSR          ;WAIT FOR DONe
1663  012344  100375                        BPL     .-4
1664  012346  017703  166634                MOV     @TKDBR,R3       ;PUT CHAR IN R3
1665  012352  105777  166632                TSTB    @TPCSR          ;WAIT UNTIL PRINTER IS READY
1666  012356  100375                        BPL     .-4
1667  012360  010377  166626                MOV     R3,@TPDBR       ;ECHO CHAR
1668  012364  042703  000240                BIC     #BIT7!BIT5,R3   ;MASK OFF LOWER CASE
1669  012370  000207                        RTS     PC              ;RETURN
1670
1671
1672  012372  000000              ROMMAP:   0                       ;POINTER TO HI OR LO SPEED MICRO-CODE
1673
1674  012374                      LOMAP:                            ;MICRO-CODE
1675  012374                      HIMAP:
1676
```

```
                                        ;*************************** TEST 1 ****************************
                                        ;*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 1
                                        ;--------------
016376  012737  000001  001226  TST1:   MOV     #1,TSTNO
016404  012737  016470  001216          MOV     #TST2,NEXT
                                                                ;R1 CONTAINS BASE M8200-YC ADDRESS
016412  004737  022426                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
016416  032737  100000  001366          BIT     #BIT15,STAT1    ;BE SURE DMC HAS CRAM
016424  001420                          BEQ     2$              ;SKIP IF NO CRAM
016426  005000                          CLR     R0              ;R0=CRAM ADDRESS
016430  013702  012372                  MOV     ROMMAP,R2       ;R2 POINTS TO ROMMAP
016434  012711  002000          1$:     MOV     #BIT10,(R1)     ;SET ROM0
016440  010061  000004                  MOV     R0,4(R1)        ;LOAD CRAM ADDRESS
016444  012261  000006                  MOV     (R2)+,6(R1)     ;LOAD WORD TO BE WRITTEN
016450  052711  020000                  BIS     #BIT13,(R1)     ;WRITE IT!
016454  005200                          INC     R0              ;NEXT ADDRESS
016456  022700  002000                  CMP     #2000,R0        ;DONE YET?
016462  001364                          BNE     1$              ;BR IF NO
016464  005011                          CLR     (R1)            ;CLEAR SEL0
016466  104400                  2$:     SCOPE                   ;SCOPE THIS TEST


                                        ;*************************** TEST 2 ****************************
                                        ;*TEST OF BR RIGHT SHIFT
                                        ;*VERIFY THAT A DEST OF BR RSH (011) OF A MICRO-INSTRUCTION
                                        ;*SHIFTS THE RESULTING BR DATA RIGHT ONCE.
                                        ;:***********************************************************

                                        ;  TEST 2
                                        ;--------------
016470  012737  000002  001226  TST2:   MOV     #2,TSTNO
016476  012737  016602  001216          MOV     #TST3,NEXT
                                                                ;R1 CONTAINS BASE M8200-YC ADDRESS
016504  104412                          MSTCLR                  ;MASTER CLEAR M8200-YC
016506  013701  001404                  MOV     DMCSR,R1        ;R1 = DMC BASE ADDRESS
016512  005011                          CLR     (R1)            ;CLEAR SEL0
016514  012705  052525                  MOV     #52525,R5       ;START WITH 125
016520  010561  000004                  MOV     R5,4(R1)        ;PORT4<-125
016524  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016526  120500                          120500                  ;BR <- PORT4
016530  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016532  061620                          061620                  ;BR RSH<-BR, SHIFT BR RIGHT
```

```
1731  016534  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1732  016536  061225                          061225                  ;PORT5←BR
1733  016540  006005                          ROR     R5              ;R5 = "EXPECTED"
1734  016542  116104  000005                  MOVB    5(R1),R4        ;R4 = "FOUND"
1735  016546  120504                          CMPB    R5,R4           ;DID BR SHIFT RIGHT ONCE?
1736  016550  001401                          BEQ     1$              ;BR IF YES
1737  016552  104012                          HLT     12              ;BR RIGHT SHIFT ERROR
1738  016554                          1$:
1739  016554  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1740  016556  061620                          061620                  ;BR RSH←BR, SHFT BR RIGHT AGAIN
1741  016560  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1742  016562  061225                          061225                  ;PORT5←BR
1743  016564  006005                          ROR     R5              ;R5 = "EXPECTED"
1744  016566  116104  000005                  MOVB    5(R1),R4        ;R4 = "FOUND"
1745  016572  120504                          CMPB    R5,R4           ;DID BR SHIFT RIGHT?
1746  016574  001401                          BEQ     2$              ;BR IF YES
1747  016576  104012                          HLT     12              ;BR RIGHT SHIFT ERROR
1748  016600  104400                  2$:     SCOPE                   ;SCOPE THIS TEST
1749
1750
1751                                          ;************************** TEST 3 **************************
1752                                          ;*CROM READ TEST
1753                                          ;*THIS TEST READS EACH ROM LOCATION AND COMPARES
1754                                          ;*IT TO A SOFTWARE DUPLICATE OF THE CROM. THIS TEST
1755                                          ;*ALSO TESTS THE JUMP(I) MICRO-PROCESSOR INSTRUCTION.
1756                                          ;*IF THIS TEST FAILS CHECK YOUR CROM PART NUMBERS.
1757                                          ;*DRLPM-A SUPPORTS THE FOLLOWING PART NUMBERS:
1758                                          ;*
1759                                          ;*M8200-YC-AR  (M8200-YA)
1760                                          ;;*********************************************************************
1761
1762                                          ;  TEST 3
1763                                          ;--------------
1764  016602  012737  000003  001226  TST3:   MOV     #3,TSTNO
1765  016610  012737  016776  001216          MOV     #TST4,NEXT
1766  016616  012737  016654  001220          MOV     #1$,LOCK
1767                                                                  ;R1 CONTAINS BASE M8200-YC ADDRESS
1768  016624  104412                          MSTCLR                  ;MASTER CLEAR M8200-YC
1769  016626  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT RAM OR ROM
1770  016634  001057                          BNE     4$              ;SKIP TEST IF CRAM
1771  016636  004737  022426                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
1772  016642  005011                          CLR     (R1)            ;CLEAR RUN
1773  016644  013700  012372                  MOV     ROMMAP,R0       ;R0 POINTS TO SOFTWARE ROM MAP
1774  016650  005002                          CLR     R2              ;R2 CONTAINS ROM ADDRESS BITS 0-7
1775  016652  005003                          CLR     R3              ;R3 CONTAINS ROM ADDRESS BITS 8&9 IN BITS 11&12
1776  016654  042737  014377  016674  1$:     BIC     #14377,2$       ;CLEAR ADDRESS FIELDS OF INSTRUCTION
1777  016662  050237  016674                  BIS     R2,2$           ;ADD BITS 0-7 TO INSTRUCTION
1778  016666  050337  016674                  BIS     R3,2$           ;ADD BITS 11&12 TO INSTRUCTION
1779  016672  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1780  016674  100400                  2$:     100400                  ;JUMP(I) TO ROM ADDRESS IN R2 & R3
1781  016676  012711  002000                  MOV     #BIT10,(R1)     ;SET ROM0
1782  016702  011005                          MOV     (R0),R5         ;PUT "EXPECTED" IN R5
1783  016704  016104  000006                  MOV     6(R1),R4        ;PUT "FOUND" IN R4
1784  016710  020504                          CMP     R5,R4           ;COMPARE ROM CONTENTS TO SOFT DUP
```

```
1785  016712  001414                          BEQ     3$              ;BR IF OK
1786  016714  010337  001252                  MOV     R3,TEMp3        ;PUT ROM ADDRESS IN TEMP3
1787  016720  000241                          CLC                     ;FOR ERROR TYPEOUT
1788  016722  006037  001252                  ROR     TEMP3
1789  016726  006037  001252                  ROR     TEMP3
1790  016732  006037  001252                  ROR     TEMP3
1791  016736  050237  001252                  BIS     R2,TEMP3        ;TEMP3 NOW CONTAINS CORRECT ADDRESS
1792  016742  104004                          HLT     4               ;ROM READ ERROR
1793  016744  104401                3$:       SCOP1                   ;LOOP TO 1$ IF SW09=1
1794  016746  005720                          TST     (R0)+           ;BUMP SOFT POINTER
1795  016750  005202                          INC     R2              ;BUMP ROM ADDRESS
1796  016752  022702  000400                  CMP     #400,R2         ;IS R2 TO MAX YET?
1797  016756  001336                          BNE     1$              ;BR IF NO
1798  016760  005002                          CLR     R2              ;YES, RESET R2 TO 0
1799  016762  062703  004000                  ADD     #4000,R3        ;INC  TO NEXT PAGE OF ROM
1800  016766  022703  020000                  CMP     #20000,R3       ;DONE YET?
1801  016772  001330                          BNE     1$              ;BR IF NO
1802  016774  104400                4$:       SCOPE                   ;SCOPE THIS TEST
1803
1804
1805                                          ;*************************** TEST 4 ****************************
1806                                          ;*CROM TEST OF JUMP(I) NEVER MICRO-PROCESSOR INSTRUCTION.
1807                                          ;*PERFORM THE JUMP INSTRUCTION
1808                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
1809                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
1810                                          ;:***************************************************************
1811
1812                                          ;  TEST 4
1813                                          ;--------------
1814  016776  012737  000004  001226  TST4:   MOV     #4,TSTNO
1815  017004  012737  017172  001216          MOV     #TST5,NEXT
1816  017012  012737  017036  001220          MOV     #1$,LOCK
1817                                                                  ;R1 CONTAINS BASE M8200-YC ADDRESS
1818  017020  104412                          MSTCLR                  ;MASTER CLEAR M8200-YC
1819  017022  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
1820  017030  001057                          BNE     6$+2            ;SKIP TEST IF YES
1821  017032  004737  022426                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
1822  017036                          1$:
1823  017036  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
1824  017042  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1825  017044  100400                          100400                  ;START AT ROM PC=0
1826  017046  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1827  017050  114377                          114377!<400*0>          ;JUMP TO rOM PC OF 1777
1828  017052  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1829  017056  000002                          2                       ;INDEX
1830  017060  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1831  017062  001401                          BEQ     2$              ;BR IF YES
1832  017064  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
1833  017066  104401                2$:       SCOP1                   ;LOOP TO 1$ IF SW09=1
1834  017070  012737  017076  001220          MOV     #3$,LOCK        ;NEW SCOP1
1835  017076                          3$:
1836  017076  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
1837  017102  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1838  017104  100403                          100403                  ;START AT ROM PC=3
```

```
1839  017106  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1840  017110  100000                          100000!<400*0>  ;JUMP TO ROM PC OF 0
1841  017112  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1842  017116  000010                          10                      ;INDEX
1843  017120  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1844  017122  001401                          BEQ     4$              ;BR IF YES
1845  017124  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
1846  017126  104401              4$:         SCOP1                   ;LOOP TO 3$ IF SW09=1
1847  017130  012737  017136  001220          MOV     #5$,LOCK        ;NEW SCOP1
1848  017136                      5$:
1849  017136  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
1850  017142  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1851  017144  100406                          100406                  ;START AT ROM PC=6
1852  017146  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1853  017150  104125                          104125!<400*0>          ;JUMP TO ROM PC OF 525
1854  017152  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1855  017156  000016                          16                      ;INDEX
1856  017160  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
1857  017162  001401                          BEQ     6$              ;BR IF YES
1858  017164  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
1859  017166  104401              6$:         SCOP1                   ;LOOP TO 5$ IF SW59=1
1860  017170  104400                          SCOPE                   ;SCOPE THIS TEST
1861
1862                                          ;*************************** TEST 5 ****************************
1863                                          ;*CROM TEST OF JUMP(I) ALWAYS MICRO-PROCESSOR INSTRUCTION.
1864                                          ;*PERFORM THE JUMP INSTRUCTION
1865                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
1866                                          ;:************************************************************
1867
1868
1869                                          ;  TEST 5
1870                                          ;--------------
1871  017172  012737  000005  001226  TST5:   MOV     #5,TSTNO
1872  017200  012737  017352  001216          MOV     #TST6,NEXT
1873  017206  012737  017232  001220          MOV     #1$,LOCK
1874                                                                  ;R1 CONTAINS BASE M8200-YC ADDRESS
1875  017214  104412                          MSTCLR                  ;MASTER CLEAR M8200-YC
1876  017216  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
1877  017224  001051                          BNE     6$+2            ;SKIP TEST IF YES
1878  017226  004737  022426                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
1879  017232                      1$:
1880  017232  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1881  017234  100400                          100400                  ;START AT ROM PC=0
1882  017236  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1883  017240  114777                          114377!<400*1>          ;JUMP TO ROM PC OF 1777
1884  017242  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1885  017246  003776                          3776                    ;INDEX
1886  017250  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1887  017252  001401                          BEQ     2$              ;BR IF YES
1888  017254  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
1889  017256  104401              2$:         SCOP1                   ;LOOP TO 1$ IF SW09=1
1890  017260  012737  017266  001220          MOV     #3$,LOCK        ;NEW SCOP1
1891  017266                      3$:
1892  017266  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
 1893  017270  100403                          100403                  ;START AT ROM PC=3
 1894  017272  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1895  017274  100400                          100000!<400*1>  ;JUMP TO ROM PC OF 0
 1896  017276  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
 1897  017302  000000                          0                       ;INDEX
 1898  017304  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
 1899  017306  001401                          BEQ     4$              ;BR IF YES
 1900  017310  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
 1901  017312  104401              4$:         SCOP1                   ;LOOP TO 3$ IF SW09=1
 1902  017314  012737  017322  001220          MOV     #5$,LOCK        ;NEW SCOP1
 1903  017322                      5$:
 1904  017322  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1905  017324  100406                          100406                  ;START AT ROM PC=6
 1906  017326  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1907  017330  104525                          104125!<400*1>          ;JUMP TO ROM PC OF 525
 1908  017332  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
 1909  017336  001252                          1252                    ;INDEX
 1910  017340  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
 1911  017342  001401                          BEQ     6$              ;BR IF YES
 1912  017344  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
 1913  017346  104401              6$:         SCOP1                   ;LOOP TO 5$ IF SW59=1
 1914  017350  104400                          SCOPE                   ;SCOPE THIS TEST
 1915
 1916
 1917                                          ;************************** TEST 6 **************************
 1918                                          ;*CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
 1919                                          ;*SET THE C BIT, PERFORM THE JUMP INSTRUCTION,
 1920                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
 1921                                          ;************************************************************
 1922
 1923                                          ;  TEST 6
 1924                                          ;--------------
 1925  017352  012737  000006  001226  TST6:   MOV     #6,TSTNO
 1926  017360  012737  017546  001216          MOV     #TST7,NEXT
 1927  017366  012737  017412  001220          MOV     #1$,LOCK
 1928                                                                  ;R1 CONTAINS BASE M8200-YC ADDRESS
 1929  017374  104412                          MSTCLR                  ;MASTER CLEAR M8200-YC
 1930  017376  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
 1931  017404  001057                          BNE     6$+2            ;SKIP TEST IF YES
 1932  017406  004737  022426                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
 1933  017412                      1$:
 1934  017412  004737  022340                  JSR     PC,SETC ;SET THE C BIT'
 1935  017416  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1936  017420  100400                          100400                  ;START AT ROM PC=0
 1937  017422  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1938  017424  115377                          114377!<400*2>          ;JUMP TO rOM PC OF 1777
 1939  017426  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
 1940  017432  003776                          3776                    ;INDEX
 1941  017434  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
 1942  017436  001401                          BEQ     2$              ;BR IF YES
 1943  017440  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
 1944  017442  104401              2$:         SCOP1                   ;LOOP TO 1$ IF SW09=1
 1945  017444  012737  017452  001220          MOV     #3$,LOCK        ;NEW SCOP1
 1946  017452                      3$:
```

```
 1947  017452  004737  022340                        JSR     PC,SETC ;SET THE C BIT'
 1948  017456  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1949  017460  100403                                100403                  ;START AT ROM PC=3
 1950  017462  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1951  017464  101000                                100000!<400*2>  ;JUMP TO rOM PC OF 0
 1952  017466  004737  022364                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
 1953  017472  000000                                0                       ;INDEX
 1954  017474  020504                                CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
 1955  017476  001401                                BEQ     4$              ;BR IF YES
 1956  017500  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
 1957  017502  104401                        4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
 1958  017504  012737  017512  001220                MOV     #5$,LOCK        ;NEW SCOP1
 1959  017512                                5$:
 1960  017512  004737  022340                        JSR     PC,SETC ;SET THE C BIT'
 1961  017516  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1962  017520  100406                                100406                  ;START AT ROM PC=6
 1963  017522  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1964  017524  105125                                104125!<400*2>          ;JUMP TO ROM PC OF 525
 1965  017526  004737  022364                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
 1966  017532  001252                                1252                    ;INDEX
 1967  017534  020504                                CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
 1968  017536  001401                                BEQ     6$              ;BR IF YES
 1969  017540  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
 1970  017542  104401                        6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
 1971  017544  104400                                SCOPE                   ;SCOPE THIS TEST
 1972
 1973
 1974                                                ;************************** TEST 7 **************************
 1975                                                ;*CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
 1976                                                ;*SET THE Z BIT, PERFORM THE JUMP INSTRUCTION,
 1977                                                ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
 1978                                                ;;**********************************************************
 1979
 1980                                                ;  TEST 7
 1981                                                ;--------------
 1982  017546  012737  000007  001226        TST7:   MOV     #7,TSTNO
 1983  017554  012737  017742  001216                MOV     #TST10,NEXT
 1984  017562  012737  017606  001220                MOV     #1$,LOCK
 1985                                                                        ;R1 CONTAINS BASE M8200-YC ADDRESS
 1986  017570  104412                                MSTCLR                  ;MASTER CLEAR M8200-YC
 1987  017572  032737  100000  001366                BIT     #BIT15,STAT1    ;IS IT CRAM?
 1988  017600  001057                                BNE     6$+2            ;SKIP TEST IF YES
 1989  017602  004737  022426                        JSR     PC,MAPCK        ;CHECK FOR HI OR LO
 1990  017606                                1$:
 1991  017606  004737  022356                        JSR     PC,SETZ ;SET THE Z BIT'
 1992  017612  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1993  017614  100400                                100400                  ;START AT ROM PC=0
 1994  017616  104414                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1995  017620  115777                                114377!<400*3>          ;JUMP TO ROM PC OF 1777
 1996  017622  004737  022364                        JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
 1997  017626  003776                                3776                    ;INDEX
 1998  017630  020504                                CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
 1999  017632  001401                                BEQ     2$              ;BR IF YES
 2000  017634  104006                                HLT     6               ;ERROR, CROM PC IS WRONG
```

```
2001  017636  104401                    2$:   SCOP1                    ;LOOP TO 1$ IF SW09=1
2002  017640  012737  017646  001220          MOV     #3$,LOCK          ;NEW SCOP1
2003  017646                            3$:
2004  017646  004737  022356                  JSR     PC,SETZ ;SET THE Z BIT'
2005  017652  104414                          ROMCLK                    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2006  017654  100403                          100403                    ;START AT ROM PC=3
2007  017656  104414                          ROMCLK                    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2008  017660  101400                          100000!<400*3>  ;JUMP TO  ROM PC OF 0
2009  017662  004737  022364                  JSR     PC,ROMDAT         ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2010  017666  000000                          0                         ;INDEX
2011  017670  020504                          CMP     R5,R4             ;ARE NEW PC CONTENTS CORRECT?
2012  017672  001401                          BEQ     4$                ;BR IF YES
2013  017674  104006                          HLT     6                 ;ERROR, CROM PC IS WRONG
2014  017676  104401                    4$:   SCOP1                     ;LOOP TO 3$ IF SW09=1
2015  017700  012737  017706  001220          MOV     #5$,LOCK          ;NEW SCOP1
2016  017706                            5$:
2017  017706  004737  022356                  JSR     PC,SETZ ;SET THE Z BIT'
2018  017712  104414                          ROMCLK                    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2019  017714  100406                          100406                    ;START AT ROM PC=6
2020  017716  104414                          ROMCLK                    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2021  017720  105525                          104125!<400*3>            ;JUMP TO ROM PC OF 525
2022  017722  004737  022364                  JSR     PC,ROMDAT         ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2023  017726  001252                          1252                      ;INDEX
2024  017730  020504                          CMP     R5,R4             ;ARE NEW ROM PC CONTENTS CORRECT?
2025  017732  001401                          BEQ     6$                ;BR IF YES
2026  017734  104006                          HLT     6                 ;ERROR, CROM PC IS WRONG
2027  017736  104401                    6$:   SCOP1                     ;LOOP TO 5$ IF SW59=1
2028  017740  104400                          SCOPE                     ;SCOPE THIS TEST
2029
2030                                          ;*************************** TEST 10 ****************************
2031                                          ;*CROM TEST OF JUMP(I) ON BRO SET MICRO-PROCESSOR INSTRUCTION.
2032                                          ;*SET THE BRO BIT, PERFORM THE JUMP INSTRUCTION,
2033                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2034                                          ;*****************************************************************
2035
2036                                          ;  TEST 10
2037                                          ;--------------
2038
2039  017742  012737  000010  001226  TST10:  MOV     #10,TSTNO
2040  017750  012737  020136  001216          MOV     #TST11,NEXT
2041  017756  012737  020002  001220          MOV     #1$,LOCK
2042                                                                    ;R1 CONTAINS BASE M8200-YC ADDRESS
2043  017764  104412                          MSTCLR                    ;MASTER CLEAR M8200-YC
2044  017766  032737  100000  001366          BIT     #BIT15,STAT1      ;IS IT CRAM?
2045  017774  001057                          BNE     6$+2              ;SKIP TEST IF YES
2046  017776  004737  022426                  JSR     PC,MAPCK          ;CHECK FOR HI OR LO
2047  020002                            1$:
2048  020002  004737  022310                  JSR     PC,SETBRO         ;SET THE BRO BIT'
2049  020006  104414                          ROMCLK                    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2050  020010  100400                          100400                    ;START AT ROM PC=0
2051  020012  104414                          ROMCLK                    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2052  020014  116377                          114377!<400*4>            ;JUMP TO ROM PC OF 1777
2053  020016  004737  022364                  JSR     PC,ROMDAT         ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2054  020022  003776                          3776                      ;INDEX
```

```
2055  020024  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2056  020026  001401                        BEQ     2$              ;BR IF YES
2057  020030  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2058  020032  104401                2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2059  020034  012737  020042  001220        MOV     #3$,LOCK        ;NEW SCOPE
2060  020042                        3$:
2061  020042  004737  022310                JSR     PC,SETBR0       ;SET THE BR0 BIT'
2062  020046  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2063  020050  100403                        100403                  ;START AT ROM PC=3
2064  020052  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2065  020054  102000                        100000!<400*4>  ;JUMP TO rOM PC OF 0
2066  020056  004737  022364                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2067  020062  000000                        0                       ;INDEX
2068  020064  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2069  020066  001401                        BEQ     4$              ;BR IF YES
2070  020070  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2071  020072  104401                4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2072  020074  012737  020102  001220        MOV     #5$,LOCK        ;NEW SCOPE
2073  020102                        5$:
2074  020102  004737  022310                JSR     PC,SETBR0       ;SET THE BR0 BIT'
2075  020106  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2076  020110  100406                        100406                  ;START AT ROM PC=6
2077  020112  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2078  020114  106125                        104125!<400*4>          ;JUMP TO rOM PC OF 525
2079  020116  004737  022364                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2080  020122  001252                        1252                    ;INDEX
2081  020124  020504                        CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2082  020126  001401                        BEQ     6$              ;BR IF YES
2083  020130  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2084  020132  104401                6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2085  020134  104400                        SCOPE                   ;SCOPE THIS TEST
2086
2087
2088                                        ;************************** TEST 11 **************************
2089                                        ;*CROM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION.
2090                                        ;*SET THE BR1 BIT, PERFORM THE JUMP INSTRUCTION.
2091                                        ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2092                                        ;:************************************************************
2093
2094                                        ;  TEST 11
2095                                        ;--------------
2096  020136  012737  000011  001226  TST11: MOV     #11,TSTNO
2097  020144  012737  020332  001216        MOV     #TST12,NEXT
2098  020152  012737  020176  001220        MOV     #1$,LOCK
2099                                                                ;R1 CONTAINS BASE M8200-YC ADDRESS
2100  020160  104412                        MSTCLR                  ;MASTER CLEAR M8200-YC
2101  020162  032737  100000  001366        BIT     #BIT15,STAT1    ;IS IT CRAM?
2102  020170  001057                        BNE     6$+2            ;SKIP TEST IF YES
2103  020172  004737  022426                JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2104  020176                        1$:
2105  020176  004737  022316                JSR     PC,SETBR1       ;SET THE BR1 BIT'
2106  020202  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2107  020204  100400                        100400                  ;START AT ROM PC=0
2108  020206  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
2109  020210  116777                        114377!<400*5>          ;JUMP TO ROM PC OF 1777
2110  020212  004737  022364                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2111  020216  003776                        3776                    ;INDEX
2112  020220  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2113  020222  001401                        BEQ     2$              ;BR IF YES
2114  020224  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2115  020226  104401              2$:       SCOP1                   ;LOOP TO 1$ IF SW09=1
2116  020230  012737  020236  001220        MOV     #3$,LOCK        ;NEW SCOP1
2117  020236                    3$:
2118  020236  004737  022316                JSR     PC,SETBR1       ;SET THE BR1 BIT'
2119  020242  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2120  020244  100403                        100403                  ;START AT ROM PC=3
2121  020246  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2122  020250  102400                        100000!<400*5>  ;JUMP TO ROM PC OF 0
2123  020252  004737  022364                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2124  020256  000000                        0                       ;INDEX
2125  020260  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2126  020262  001401                        BEQ     4$              ;BR IF YES
2127  020264  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2128  020266  104401              4$:       SCOP1                   ;LOOP TO 3$ IF SW09=1
2129  020270  012737  020276  001220        MOV     #5$,LOCK        ;NEW SCOP1
2130  020276                    5$:
2131  020276  004737  022316                JSR     PC,SETBR1       ;SET THE BR1 BIT'
2132  020302  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2133  020304  100406                        100406                  ;START AT ROM PC=6
2134  020306  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2135  020310  106525                        104125!<400*5>          ;JUMP TO ROM PC OF 525
2136  020312  004737  022364                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2137  020316  001252                        1252                    ;INDEX
2138  020320  020504                        CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2139  020322  001401                        BEQ     6$              ;BR IF YES
2140  020324  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2141  020326  104401              6$:       SCOP1                   ;LOOP TO 5$ IF SW59=1
2142  020330  104400                        SCOPE                   ;SCOPE THIS TEST
2143
2144
2145                                        ;*************************** TEST 12 ****************************
2146                                        ;*CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION.
2147                                        ;*SET THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
2148                                        ;*VERIFY THE JUMP'BY READING THE CONTENTS OF THE NEW ROM PC
2149                                        ;:****************************************************************
2150
2151                                        ;  TEST 12
2152                                        ;--------------
2153  020332  012737  000012  001226  TST12:  MOV     #12,TSTNO
2154  020340  012737  020526  001216        MOV     #TST13,NEXT
2155  020346  012737  020372  001220        MOV     #1$,LOCK
2156                                                                ;R1 CONTAINS BASE M8200-YC ADDRESS
2157  020354  104412                        MSTCLR                  ;MASTER CLEAR M8200-YC
2158  020356  032737  100000  001366        BIT     #BIT15,STAT1    ;IS IT CRAM?
2159  020364  001057                        BNE     6$+2            ;SKIP TEST IF YES
2160  020366  004737  022426                JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2161  020372                    1$:
2162  020372  004737  022324                JSR     PC,SETBR4       ;SET THE BR4 BIT'
```

```
2163  020376  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2164  020400  100400                          100400                  ;START AT ROM PC=0
2165  020402  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2166  020404  117377                          114377!<400*6>          ;JUMP TO ROM PC OF 1777
2167  020406  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2168  020412  003776                          3776                    ;INDEX
2169  020414  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2170  020416  001401                          BEQ     2$              ;BR IF YES
2171  020420  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2172  020422  104401              2$:         SCOP1                   ;LOOP TO 1$ IF SW09=1
2173  020424  012737  020432  001220          MOV     #3$,LOCK        ;NEW SCOP1
2174  020432                      3$:
2175  020432  004737  022324                  JSR     PC,SETBR4       ;SET THE BR4 BIT'
2176  020436  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2177  020440  100403                          100403                  ;START AT ROM PC=3
2178  020442  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2179  020444  103000                          100000!<400*6>  ;JUMP TO ROM PC OF 0
2180  020446  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2181  020452  000000                          0                       ;INDEX
2182  020454  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2183  020456  001401                          BEQ     4$              ;BR IF YES
2184  020460  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2185  020462  104401              4$:         SCOP1                   ;LOOP TO 3$ IF SW09=1
2186  020464  012737  020472  001220          MOV     #5$,LOCK        ;NEW SCOP1
2187  020472                      5$:
2188  020472  004737  022324                  JSR     PC,SETBR4       ;SET THE BR4 BIT'
2189  020476  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2190  020500  100406                          100406                  ;START AT ROM PC=6
2191  020502  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2192  020504  107125                          104125!<400*6>          ;JUMP TO ROM PC OF 525
2193  020506  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2194  020512  001252                          1252                    ;INDEX
2195  020514  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2196  020516  001401                          BEQ     6$              ;BR IF YES
2197  020520  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2198  020522  104401              6$:         SCOP1                   ;LOOP TO 5$ IF SW59=1
2199  020524  104400                          SCOPE                   ;SCOPE THIS TEST
2200
2201
2202                                          ;*************************** TEST 13 ***************************
2203                                          ;*CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION.
2204                                          ;*SET THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
2205                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2206                                          ;:************************************************************
2207
2208                                          ;  TEST 13
2209                                          ;--------------
2210  020526  012737  000013  001226  TST13:  MOV     #13,TSTNO
2211  020534  012737  020722  001216          MOV     #TST14,NEXT
2212  020542  012737  020566  001220          MOV     #1$,LOCK
2213                                                                  ;R1 CONTAINS BASE M8200-YC ADDRESS
2214  020550  104412                          MSTCLR                  ;MASTER CLEAR M8200-YC
2215  020552  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2216  020560  001057                          BNE     6$+2            ;SKIP TEST IF YES
```

```
2217  020562  004737  022426              1$:   JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2218  020566
2219  020566  004737  022332                    JSR     PC,SETBR7       ;SET THE BR7 BIT'
2220  020572  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2221  020574  100400                            100400                  ;START AT ROM PC=0
2222  020576  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2223  020600  117777                            114377!<400*7>          ;JUMP TO ROM PC OF 1777
2224  020602  004737  022364                    JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2225  020606  003776                            3776                    ;INDEX
2226  020610  020504                            CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2227  020612  001401                            BEQ     2$              ;BR IF YES
2228  020614  104006                            HLT     6               ;ERROR, CROM PC IS WRONG
2229  020616  104401              2$:           SCOP1                   ;LOOP TO 1$ IF SW09=1
2230  020620  012737  020626  001220            MOV     #3$,LOCK        ;NEW SCOP1
2231  020626                      3$:
2232  020626  004737  022332                    JSR     PC,SETBR7       ;SET THE BR7 BIT'
2233  020632  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2234  020634  100403                            100403                  ;START AT ROM PC=3
2235  020636  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2236  020640  103400                            100000!<400*7>  ;JUMP TO ROM PC OF 0
2237  020642  004737  022364                    JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2238  020646  000000                            0                       ;INDEX
2239  020650  020504                            CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2240  020652  001401                            BEQ     4$              ;BR IF YES
2241  020654  104006                            HLT     6               ;ERROR, CROM PC IS WRONG
2242  020656  104401              4$:           SCOP1                   ;LOOP TO 3$ IF SW09=1
2243  020660  012737  020666  001220            MOV     #5$,LOCK        ;NEW SCOP1
2244  020666                      5$:
2245  020666  004737  022332                    JSR     PC,SETBR7       ;SET THE BR7 BIT'
2246  020672  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2247  020674  100406                            100406                  ;START AT ROM PC=6
2248  020676  104414                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2249  020700  107525                            104125!<400*7>          ;JUMP TO ROM PC OF 525
2250  020702  004737  022364                    JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2251  020706  001252                            1252                    ;INDEX
2252  020710  020504                            CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2253  020712  001401                            BEQ     6$              ;BR IF YES
2254  020714  104006                            HLT     6               ;ERROR, CROM PC IS WRONG
2255  020716  104401              6$:           SCOP1                   ;LOOP TO 5$ IF SW59=1
2256  020720  104400                            SCOPE                   ;SCOPE THIS TEST
2257
2258
2259                                            ;*************************** TEST 14 ****************************
2260                                            ;*CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
2261                                            ;*CLEAR THE C BIT, PERFORM THE JUMP INSTRUCTION,
2262                                            ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2263                                            ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2264                                            ;*****************************************************************
2265
2266                                            ;  TEST 14
2267                                            ;---------------
2268  020722  012737  000014  001226  TST14:    MOV     #14,TSTNO
2269  020730  012737  021116  001216            MOV     #TST15,NEXT
2270  020736  012737  020762  001220            MOV     #1$,LOCK
```

```
2271
2272  020744  104412                          MSTCLR                  ;R1 CONTAINS BASE M8200-YC ADDRESS
2273  020746  032737  100000  001366          BIT     #BIT15,STAT1    ;MASTER CLEAR M8200-YC
2274  020754  001057                          BNE     6$+2            ;IS IT CRAM?
2275  020756  004737  022426                  JSR     PC,MAPCK        ;SKIP TEST IF YES
2276  020762                          1$:     ;CHECK FOR HI OR LO
2277  020762  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2278  020766  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2279  020770  100400                          100400                  ;START AT ROM PC=0
2280  020772  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2281  020774  115377                          114377!<400*2>          ;JUMP TO ROM PC OF 1777
2282  020776  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2283  021002  000002                          2                       ;INDEX
2284  021004  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2285  021006  001401                          BEQ     2$              ;BR IF YES
2286  021010  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2287  021012  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2288  021014  012737  021022  001220          MOV     #3$,LOCK        ;NEW SCOP1
2289  021022                          3$:
2290  021022  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2291  021026  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2292  021030  100403                          100403                  ;START AT ROM PC=3
2293  021032  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2294  021034  101000                          100000!<400*2>  ;JUMP TO ROM PC OF 0
2295  021036  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2296  021042  000010                          10                      ;INDEX
2297  021044  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2298  021046  001401                          BEQ     4$              ;BR IF YES
2299  021050  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2300  021052  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2301  021054  012737  021062  001220          MOV     #5$,LOCK        ;NEW SCOP1
2302  021062                          5$:
2303  021062  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2304  021066  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2305  021070  100406                          100406                  ;START AT ROM PC=6
2306  021072  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2307  021074  105125                          104125!<400*2>          ;JUMP TO rOM PC OF 525
2308  021076  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2309  021102  000016                          16                      ;INDEX
2310  021104  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2311  021106  001401                          BEQ     6$              ;BR IF YES
2312  021110  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2313  021112  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2314  021114  104400                          SCOPE                   ;SCOPE THIS TEST
2315
2316
2317                                          ;************************** TEST 15 **************************
2318                                          ;*CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
2319                                          ;*CLEAR THE Z BIT, PERFORM THE JUMP INSTRUCTION,
2320                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2321                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2322                                          ;:************************************************************
2323
2324                                          ;  TEST 15
```

```
2325
2326  021116  012737  000015  001226  TST15:  ;---------------
2327  021124  012737  021312  001216           MOV     #15,TSTNO
2328  021132  012737  021156  001220           MOV     #TST16,NEXT
2329                                           MOV     #1$,LOCK
2330  021140  104412                                                 ;R1 CONTAINS BASE M8200-YC ADDRESS
2331  021142  032737  100000  001366           MSTCLR                ;MASTER CLEAR M8200-YC
2332  021150  001057                           BIT     #BIT15,STAT1  ;IS IT CRAM?
2333  021152  004737  022426                   BNE     6$+2          ;SKIP TEST IF YES
2334  021156                           1$:     JSR     PC,MAPCK      ;CHECK FOR HI OR LO
2335  021156  004737  022272                   JSR     PC,CLRALL     ;CLEAR ALL  CONDITIONS
2336  021162  104414                           ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2337  021164  100400                           100400                ;START AT ROM PC=0
2338  021166  104414                           ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2339  021170  115777                           114377!<400*3>        ;JUMP TO ROM PC OF 1777
2340  021172  004737  022364                   JSR     PC,ROMDAT     ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2341  021176  000002                           2                     ;INDEX
2342  021200  020504                           CMP     R5,R4         ;ARE NEW PC CONTENTS CORRECT?
2343  021202  001401                           BEQ     2$            ;BR IF YES
2344  021204  104006                           HLT     6             ;ERROR, CROM PC IS WRONG
2345  021206  104401                   2$:     SCOP1                 ;LOOP TO 1$ IF SW09=1
2346  021210  012737  021216  001220           MOV     #3$,LOCK      ;NEW SCOP1
2347  021216                           3$:
2348  021216  004737  022272                   JSR     PC,CLRALL     ;CLEAR ALL  CONDITIONS
2349  021222  104414                           ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2350  021224  100403                           100403                ;START AT ROM PC=3
2351  021226  104414                           ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2352  021230  101400                           100000!<400*3>  ;JUMP TO ROM PC OF 0
2353  021232  004737  022364                   JSR     PC,ROMDAT     ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2354  021236  000010                           10                    ;INDEX
2355  021240  020504                           CMP     R5,R4         ;ARE NEW PC CONTENTS CORRECT?
2356  021242  001401                           BEQ     4$            ;BR IF YES
2357  021244  104006                           HLT     6             ;ERROR, CROM PC IS WRONG
2358  021246  104401                   4$:     SCOP1                 ;LOOP TO 3$ IF SW09=1
2359  021250  012737  021256  001220           MOV     #5$,LOCK      ;NEW SCOP1
2360  021256                           5$:
2361  021256  004737  022272                   JSR     PC,CLRALL     ;CLEAR ALL  CONDITIONS
2362  021262  104414                           ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2363  021264  100406                           100406                ;START AT ROM PC=6
2364  021266  104414                           ROMCLK                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2365  021270  105525                           104125!<400*3>        ;JUMP TO ROM PC OF 525
2366  021272  004737  022364                   JSR     PC,ROMDAT     ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2367  021276  000016                           16                    ;INDEX
2368  021300  020504                           CMP     R5,R4         ;ARE NEW ROM PC CONTENTS CORRECT?
2369  021302  001401                           BEQ     6$            ;BR IF YES
2370  021304  104006                           HLT     6             ;ERROR, CROM PC IS WRONG
2371  021306  104401                   6$:     SCOP1                 ;LOOP TO 5$ IF SW59=1
2372  021310  104400                           SCOPE                 ;SCOPE THIS TEST
2373
2374
2375                                           ;*************************** TEST 16 ****************************
2376                                           ;*CROM TEST OF JUMP(I) ON BRO SET MICRO-PROCESSOR INSTRUCTION.
2377                                           ;*CLEAR THE BRO BIT, PERFORM THE JUMP INSTRUCTION,
2378                                           ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
```

```
                                        ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
                                        ;:*****************************************************************

                                        ;  TEST 16
                                        ;--------------
021312  012737  000016  001226  TST16:  MOV     #16,TSTNO
021320  012737  021506  001216          MOV     #TST17,NEXT
021326  012737  021352  001220          MOV     #1$,LOCK
                                                                ;R1 CONTAINS BASE M8200-YC ADDRESS
021334  104412                          MSTCLR                  ;MASTER CLEAR M8200-YC
021336  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
021344  001057                          BNE     6$+2            ;SKIP TEST IF YES
021346  004737  022426                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
021352                          1$:
021352  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
021356  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021360  100400                          100400                  ;START AT ROM PC=0
021362  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021364  116377                          114377!<400*4>          ;JUMP TO rOM PC OF 1777
021366  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021372  000002                          2                       ;INDEX
021374  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021376  001401                          BEQ     2$              ;BR IF YES
021400  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
021402  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
021404  012737  021412  001220          MOV     #3$,LOCK        ;NEW SCOP1
021412                          3$:
021412  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
021416  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021420  100403                          100403                  ;START AT ROM PC=3
021422  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021424  102000                          100000!<400*4>  ;JUMP TO ROM PC OF 0
021426  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021432  000010                          10                      ;INDEX
021434  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021436  001401                          BEQ     4$              ;BR IF YES
021440  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
021442  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
021444  012737  021452  001220          MOV     #5$,LOCK        ;NEW SCOP1
021452                          5$:
021452  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
021456  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021460  100406                          100406                  ;START AT ROM PC=6
021462  104414                          ROMCLK                  ;NEXT WORd IS INSTRUCTION, ROMCLK PC=5304
021464  106125                          104125!<400*4>          ;JUMP TO ROM PC OF 525
021466  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021472  000016                          16                      ;INDEX
021474  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
021476  001401                          BEQ     6$              ;BR IF YES
021500  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
021502  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
021504  104400                          SCOPE                   ;SCOPE THIS TEST
```

```
                                        ;**********************************  TEST 17  ****************************
                                        ;*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).
                                        ;:*************************************************************************

                                        ;  TEST 17
                                        ;--------------
021506  012737  000017  001226  TST17:  MOV     #17,TSTNO
021514  012737  021702  001216         MOV     #TST20,NEXT
021522  012737  021546  001220         MOV     #1$,LOCK
                                                                ;R1 CONTAINS BASE M8200-YC ADDRESS
021530  104412                         MSTCLR                  ;MASTER CLEAR M8200-YC
021532  032737  100000  001366         BIT     #BIT15,STAT1    ;IS IT CRAM?
021540  001057                         BNE     6$+2            ;SKIP TEST IF YES
021542  004737  022426                 JSR     PC,MAPCK        ;CHECK FOR HI OR LO
021546                          1$:
021546  004737  022272                 JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
021552  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021554  100400                         100400                  ;START AT ROM PC=0
021556  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021560  116777                         114377!<400*5>          ;JUMP TO ROM PC OF 1777
021562  004737  022364                 JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021566  000002                         2                       ;INDEX
021570  020504                         CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021572  001401                         BEQ     2$              ;BR IF YES
021574  104006                         HLT     6               ;ERROR, CROM PC IS WRONG
021576  104401                  2$:    SCOP1                   ;LOOP TO 1$ IF SW09=1
021600  012737  021606  001220         MOV     #3$,LOCK        ;NEW SCOP1
021606                          3$:
021606  004737  022272                 JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
021612  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021614  100403                         100403                  ;START AT ROM PC=3
021616  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021620  102400                         100000!<400*5>  ;JUMP TO ROM PC OF 0
021622  004737  022364                 JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021626  000010                         10                      ;INDEX
021630  020504                         CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
021632  001401                         BEQ     4$              ;BR IF YES
021634  104006                         HLT     6               ;ERROR, CROM PC IS WRONG
021636  104401                  4$:    SCOP1                   ;LOOP TO 3$ IF SW09=1
021640  012737  021646  001220         MOV     #5$,LOCK        ;NEW SCOP1
021646                          5$:
021646  004737  022272                 JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
021652  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021654  100406                         100406                  ;START AT ROM PC=6
021656  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021660  106525                         104125!<400*5>          ;JUMP TO ROM PC OF 525
021662  004737  022364                 JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
021666  000016                         16                      ;INDEX
021670  020504                         CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
021672  001401                         BEQ     6$              ;BR IF YES
021674  104006                         HLT     6               ;ERROR, CROM PC IS WRONG
```

```
2487  021676  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2488  021700  104400                          SCOPE                   ;SCOPE THIS TEST
2489
2490
2491                                          ;************************** TEST 20 **************************
2492                                          ;*CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION.
2493                                          ;*CLEAR THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
2494                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2495                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2496                                          ;:**********************************************************************
2497
2498                                          ;  TEST 20
2499                                          ;--------------
2500  021702  012737  000020  001226  TST20:  MOV     #20,TSTNO
2501  021710  012737  022076  001216          MOV     #TST21,NEXT
2502  021716  012737  021742  001220          MOV     #1$,LOCK
2503                                                                  ;R1 CONTAINS BASE M8200-YC ADDRESS
2504  021724  104412                         MSTCLR                  ;MASTER CLEAR M8200-YC
2505  021726  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2506  021734  001057                         BNE     6$+2            ;SKIP TEST IF YES
2507  021736  004737  022426                 JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2508  021742                          1$:
2509  021742  004737  022272                 JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2510  021746  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2511  021750  100400                         100400                  ;START AT ROM PC=0
2512  021752  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2513  021754  117377                         114377!<400*6>          ;JUMP TO ROM PC OF 1777
2514  021756  004737  022364                 JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2515  021762  000002                         2                       ;INDEX
2516  021764  020504                         CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2517  021766  001401                         BEQ     2$              ;BR IF YES
2518  021770  104006                         HLT     6               ;ERROR, CROM PC IS WRONG
2519  021772  104401                  2$:    SCOP1                   ;LOOP TO 1$ IF SW09=1
2520  021774  012737  022002  001220         MOV     #3$,LOCK        ;NEW SCOP1
2521  022002                          3$:
2522  022002  004737  022272                 JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2523  022006  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2524  022010  100403                         100403                  ;START AT ROM PC=3
2525  022012  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2526  022014  103000                         100000!<400*6>  ;JUMP TO ROM PC OF 0
2527  022016  004737  022364                 JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2528  022022  000010                         10                      ;INDEX
2529  022024  020504                         CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2530  022026  001401                         BEQ     4$              ;BR IF YES
2531  022030  104006                         HLT     6               ;ERROR, CROM PC IS WRONG
2532  022032  104401                  4$:    SCOP1                   ;LOOP TO 3$ IF SW09=1
2533  022034  012737  022042  001220         MOV     #5$,LOCK        ;NEW SCOP1
2534  022042                          5$:
2535  022042  004737  022272                 JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2536  022046  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2537  022050  100406                         100406                  ;START AT ROM PC=6
2538  022052  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2539  022054  107125                         104125!<400*6>          ;JUMP TO ROM PC OF 525
2540  022056  004737  022364                 JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
```

```
2541  022062  000016                          16                      ;INDEX
2542  022064  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2543  022066  001401                          BEQ     6$              ;BR IF YES
2544  022070  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2545  022072  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2546  022074  104400                          SCOPE                   ;SCOPE THIS TEST
2547
2548
2549                                          ;*************************** TEST 21 ****************************
2550                                          ;*CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION.
2551                                          ;*CLEAR THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
2552                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2553                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2554                                          ;:************************************************************
2555
2556                                          ;  TEST 21
2557                                          ;--------------
2558  022076  012737  000021  001226  TST21:  MOV     #21,TSTNO
2559  022104  012737  003374  001216          MOV     #.EOP,NEXT
2560  022112  012737  022136  001220          MOV     #1$,LOCK
2561                                                                  ;R1 CONTAINS BASE M8200-YC ADDRESS
2562  022120  104412                          MSTCLR                  ;MASTER CLEAR M8200-YC
2563  022122  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2564  022130  001057                          BNE     6$+2            ;SKIP TEST IF YES
2565  022132  004737  022426                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2566  022136                          1$:
2567  022136  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2568  022142  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2569  022144  100400                          100400                  ;START AT ROM PC=0
2570  022146  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2571  022150  117777                          114377!<400*7>          ;JUMP TO ROM PC OF 1777
2572  022152  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2573  022156  000002                          2                       ;INDEX
2574  022160  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2575  022162  001401                          BEQ     2$              ;BR IF YES
2576  022164  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2577  022166  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2578  022170  012737  022176  001220          MOV     #3$,LOCK        ;NEW SCOP1
2579  022176                          3$:
2580  022176  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2581  022202  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2582  022204  100403                          100403                  ;START AT ROM PC=3
2583  022206  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2584  022210  103400                          100000!<400*7>  ;JUMP TO ROM PC OF 0
2585  022212  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2586  022216  000010                          10                      ;INDEX
2587  022220  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2588  022222  001401                          BEQ     4$              ;BR IF YES
2589  022224  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2590  022226  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2591  022230  012737  022236  001220          MOV     #5$,LOCK        ;NEW SCOP1
2592  022236                          5$:
2593  022236  004737  022272                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2594  022242  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
2595  022244  100406                          100406                  ;START AT ROM PC=6
2596  022246  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2597  022250  107525                          104125!<400*7>          ;JUMP TO ROM PC OF 525
2598  022252  004737  022364                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2599  022256  000016                          16                      ;INDEX
2600  022260  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2601  022262  001401                          BEQ     6$              ;BR IF YES
2602  022264  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2603  022266  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2604  022270  104400                          SCOPE                   ;SCOPE THIS TEST
2605
2606
2607                                  ;SUBROUTINES
2608                                  ;-----------
2609
2610  022272                          CLRALL:
2611                                          ;THIS SUBROUTINE CLEARS THE C&Z BITS AND THE BR
2612
2613  022272  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2614  022274  000400                          000400                  ;BR←0
2615  022276  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2616  022300  063220                          063220                  ;SP(0)←BR
2617  022302  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2618  022304  060400                          060400                  ;BR←SP(0)+BR
2619  022306  000207                          RTS     PC
2620
2621
2622  022310                          SETBR0:
2623                                          ;THIS SUBROUTINE SETS BR0 BIT
2624
2625  022310  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2626  022312  000401                          000401                  ;BR←001
2627  022314  000207                          RTS     PC
2628
2629
2630  022316                          SETBR1:
2631                                          ;THIS SUBROUTINE SETS BR1 BIT
2632
2633  022316  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2634  022320  000402                          000402                  ;BR←002
2635  022322  000207                          RTS     PC
2636
2637
2638  022324                          SETBR4:
2639                                          ;THIS SUBROUTINE SETS BR4 BIT
2640
2641  022324  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2542  022326  000420                          000420                  ;BR←020
2643  022330  000207                          RTS     PC
2644
2645
2646  022332                          SETBR7:
2647                                          ;THIS SUBROUTINE SETS BR7 BIT
2648
```

```
2649  022332  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2650  022334  000600                        000600                  ;BR←200
2651  022336  000207                        RTS     PC
2652
2653
2654  022340                          SETC:
2655                                        ;THIS SUBROUTINE SETS THE C BIT
2656
2657  022340  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2658  022342  000777                        000777                  ;BR←377
2659  022344  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2660  022346  063220                        063220                  ;SP(0)←BR
2661  022350  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2662  022352  060400                        060400                  ;BR←SP(0)+BR
2663  022354  000207                        RTS     PC
2664
2665
2666  022356                          SETZ:
2667                                        ;THIS SUBROUTINE SETS THE Z BIT
2668
2669  022356  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2670  022360  000777                        000777                  ;BR←377
2671  022362  000207                        RTS     PC
2672
2673
2674  022364                          ROMDAT:
2675                                        ;THIS SUBROUTINE LOADS R5 WITH EXPECTED ROM CONTENTS
2676                                        ;AND LOADS R4 WITH ACTUAL ROM CONTENTS
2677
2678  022364  017600  000000                MOV     @(SP),R0        ;INDEX FOR COMPARE
2679  022370  062716  000002                ADD     #2,(SP)         ;ADJUST STACK
2680  022374  012711  002000                MOV     #BIT10,(R1)     ;SET ROM0
2681  022400  016005  012374                MOV     LOMAP(R0),R5    ;PUT EXPECTED IN R5 (LOSPEED)
2682  022404  032737  000002  001372        BIT     #BIT1,STAT3     ;LOW OR HIGH SPEED?
2683  022412  001402                        BEQ     1$              ;BR IF LOW SPEED
2684  022414  016005  012374                MOV     HIMAP(R0),R5    ;PUT EXPECTED IN R5 (HISPEED)
2685  022420  016104  000006          1$:   MOV     6(R1),R4        ;PUT "FOUND" IN R4
2686  022424  000207                        RTS     PC              ;RETURN
2687
2688  022426                          MAPCK:
2689                                        ;THIS SUBROUTINE CHECKS THE STATUS TABLE AND LOADS
2690                                        ;THE ROMMAP POINTER TO POINT TO EITHER THE HIGH OR
2691                                        ;LOW SPEED MICRO-CODE.
2692
2693  022426  012737  012374  012372        MOV     #LOMAP,ROMMAP   ;LOAD POINTER TO LOW SPEED
2694  022434  032737  000002  001372        BIT     #BIT1,STAT3     ;CHECK STATUS TABLE
2695  022442  001403                        BEQ     1$              ;BR IF LOW SPEED
2696  022444  012737  012374  012372        MOV     #HIMAP,ROMMAP   ;LOAD POINTER TO HIGH SPEED
2697  022452  000207                  1$:   RTS     PC              ;RETURN
2698
2699  022454  020200  020040  020040  MESWCH: .ASCII  <200>#     NOTE:#
      022467     200  047506  020122          .ASCII  <200>#FOR THIS PROGRAM TO RUN PROPERLY, SWITCH#
      022540  033600  020054  043117          .ASCII  <200>#7, OF THE VECTOR ADDRESS SWITCH PACK (E76),#
      022614  046600  051525  020124          .ASCIZ  <200>#MUST BE ON. (M8200-YC BOARD)#<200>
```

```
022653  377     051103  046501  EM1:    .ASCIZ  <377>/CRAM DATA ERROR/
022674  041777  040522  020115  EM2:    .ASCIZ  <377>/CRAM DUAL ADDRESSING ERROR/
022730  041777  047522  020115  EM3:    .ASCIZ  <377>/CROM DATA ERROR/
022751  377     052512  050115  EM4:    .ASCIZ  <377>/JUMP ERROR/
022765  377     042117  020124  EM5:    .ASCIZ  <377>/ODT ERROR IN IBUS* REG10/
023017  377     047511  020120  EM7:    .ASCIZ  <377>/IOP MAR TEST/
023035  377     051102  051040  EM10:   .ASCIZ  <377>/BR RIGHT SHIFT TEST/
023062  051377  041505  044505  EM11:   .ASCIZ  <377>/RECEIVE DATA ERROR/
023106  043377  042522  020105  EM12:   .ASCIZ  <377>/FREE RUNNING ERROR/
023132  041777  047117  051124  EM13:   .ASCIZ  <377>/CONTROL OUT ERROR/

023155  377     054105  042520  DH1:    .ASCIZ  <377>/EXPECTED  FOUND  ADDRESS/
023207  377     054105  042520  DH2:    .ASCIZ  <377>/EXPECTED  FOUND/
023230  020377  042523  032114  DH3:    .ASCIZ  <377>/ SEL4      SEL6/
023251  377     042012  046122  ROM1:   .ASCII  <377><12>/DRLPM-A SUPPORTS THE FOLLOWING CROM VERSIONS:/
023330  005377  050114  026501          .ASCIZ  <377><12>/LPA- M8200-YC  VERSION 4 MICRO CODE/<377><12>
                                .EVEN

023400  000003                  DT1:    3
023402  006     004                     .BYTE   6,4
023404  001264                          $AVR2
023406  006     004                     .BYTE   6,4
023410  001270                          $AVR4
023412  004     002                     .BYTE   4,2
023414  001260                          $AVR0
023416  000003                  DT2:    3
023420  006     004                     .BYTE   6,4
023422  001272                          $AVR5
023424  006     004                     .BYTE   6,4
023426  001270                          $AVR4
023430  004     002                     .BYTE   4,2
023432  001264                          $AVR2
023434  000003                  DT3:    3
023436  006     004                     .BYTE   6,4
023440  001272                          $AVR5
023442  006     004                     .BYTE   6,4
023444  001270                          $AVR4
023446  004     002                     .BYTE   4,2
023450  001252                          TEMP3
023452  000002                  DT4:    2
023454  003     007                     .BYTE   3,7
023456  001272                          $AVR5
023460  003     002                     .BYTE   3,2
023462  001270                          $AVR4
023464  000002                  DT5:    2
023466  006     004                     .BYTE   6,4
023470  001272                          $AVR5
023472  006     002                     .BYTE   6,2
023474  001270                          $AVR4
023476  000003                  DT7:    3
023500  003     010                     .BYTE   3,10
023502  001272                          $AVR5
023504  003     004                     .BYTE   3,4
```

```
023506  001270                          SAVR4
023510      004     002                 .BYTE   4,2
023512  001264                          SAVR2
023514  000003                  DT10:   3
023516      003     007                 .BYTE   3,7
023520  001272                          SAVR5
023522      003     004                 .BYTE   3,4
023524  001270                          SAVR4
023526      006     002                 .BYTE   6,2
023530  001252                          TEMp3
023532  000002                  DT11:   2
023534      006     004                 .BYTE   6,4
023536  001252                          TEMP3
023540      006     002                 .BYTE   6,2
023542  001254                          TEMP4

023544                          .ERRTAB:
023544  000000                          0
023546  000000                          0
023550  000000                          0
023552  022653                  EM1
023554  023155                  DH1     ;HLT    1
023556  023400                  DT1
023560  022674                  EM2
023562  023155                  DH1     ;HLT    2
023564  023400                  DT1
023566  022653                  EM1
023570  023155                  DH1     ;HLT    3
023572  023416                  DT2
023574  022730                  EM3
023576  023155                  DH1     ;HLT    4
023600  023434                  DT3
023602  022751                  EM4
023604  023207                  DH2     ;HLT    5
023606  023452                  DT4
023610  022751                  EM4
023612  023207                  DH2     ;HLT    6
023614  023464                  DT5
023616  022765                  EM5
023620  023207                  DH2     ;HLT    7
023622  023452                  DT4
023624  000000                  0
023626  000000                  0
023630  000000                  0
023632  023017                  EM7
023634  023155                  DH1     ;HLT    11
023636  023476                  DT7
023640  023035                  EM10
023642  023207                  DH2     ;HLT    12
023644  023452                  DT4
023646  023062                  EM11
023650  023155                  DH1     ;HLT    13
023652  023514                  DT10
023654  023106                  EM12
```

```
023656  000000                  0       ;HLT    14
023660  000000                  0
023662  023106                  EM12
023664  023207                  DH2     ;HLT    15
023666  023464                  DT5
023670  023132                  EM13
023672  023230                  DH3     ;HLT    16
023674  023532                  DT11

023676                          CORMAX:
        000001                  .END
```

```
ADRCNT= 004403   AUDONE  003034   AUSTRT  002456   AUTO.S  010552
BINWRD  004724   BIT0   = 000001   BIT1   = 000002   BIT10  = 002000
BIT11  = 004000   BIT12  = 010000   BIT13  = 020000   BIT14  = 040000
BIT15  = 100000   BIT2   = 000004   BIT3   = 000010   BIT4   = 000020
BIT5   = 000040   BIT6   = 000100   BIT7   = 000200   BIT8   = 000400
BIT9   = 001000   BM     007075   BRLVL  012324   BRW    003740
BRX     003742   BUF1.   012374 G  CHRCNT  004722   CKSWR  007646
CKSWR1  007726   CKSWR2  007740   CKSWR3  007744   CKSWR4  007750
CKSWR5  010054   CLKX    001242   CLRALL  022272   CNERR  007323
CNT.MA  001702   CNVRT  = 104411   CONERR  007244   CONN   007135
CONTAB  003006   CONVRT= 104410   CORMAX  023676   CRAM   006627
CREAM   001320   CSR     006531   CSRMAP  010554   CYCLE  010120
DATABP  005226   DATACL= 104415   DATAHD  005214   DELAY  = 104413
DEVADR  004400   DEVTAB  003020   DH1     023155   DH2    023207
DH3     023230   DISPLA  001200   DISPRE  000174   DMACTV  001306
DMCM    007344   DMCr00  001500   DMCR01  001510   DMCR02  001520
DMCR03  001530   DMCR04  001540   DMCR05  001550   DMCr06  001560
DMCr07  001570   DMCR10  001600   DMCR11  001610   DMCR12  001620
DMCR13  001630   DMCR14  001640   DMCr15  001650   DMCR16  001660
DMCR17  001670   DMCSR   001404   DMCSRH  001406   DMCTL   001410
DMNUM   001310   DMP04   001412   DMP06   001414   DMRLVL  001376
DMRVEC  001374   DMS100  001502   DMS101  001512   DMS102  001522
DMS103  001532   DMS104  001542   DMS105  001552   DMS106  001562
DMS107  001572   DMS110  001602   DMS111  001612   DMS112  001622
DMS113  001632   DMS114  001642   DMS115  001652   DMS116  001662
DMS117  001672   DMS200  001504   DMS201  001514   DMS202  001524
DMS203  001534   DMS204  001544   DMS205  001554   DMS206  001564
DMS207  001574   DMS210  001604   DMS211  001614   DMS212  001624
DMS213  001634   DMS214  001644   DMS215  001654   DMS216  001664
DMS217  001674   DMS300  001506   DMS301  001516   DMS302  001526
DMS303  001536   DMS304  001546   DMS305  001556   DMS306  001566
DMS307  001576   DMS310  001606   DMS311  001616   DMS312  001626
DMS313  001636   DMS314  001646   DMS315  001656   DMS316  001666
DMS317  001676   DMTLVL  001402   DMTVEC  001400   DM.END  001700
DM.MAP  001500   DONE    003744   DT1     023400   DT10   023514
DT11    023532   DT2     023416   DT3     023434   DT4    023452
DT5     023464   DT7     023476   EM1     022653   EM10   023035
EM11    023062   EM12    023106   EM13    023132   EM2    022674
EM3     022730   EM4     022751   EM5     022765   EM7    023017
ERCT00  001704   ERCT01  001710   ERCT02  001714   ERCT03  001720
ERCT04  001724   ERCT05  001730   ERCT06  001734   ERCT07  001740
ERCT10  001744   ERCT11  001750   ERCT12  001754   ERCT13  001760
ERCT14  001764   ERCT15  001770   ERCT16  001774   ERCT17  002000
ERR     002710   ERRCNT  001232   ERRFLG  001325   ERRMSG  005202
ERRPC   003000   ERTABO  005332   EXIT   = 000205   EXITER  005262
FLOAT   002546   FY      002576   HALTS   005232   HILIM  004376
HIMAP   012374   ICOUNT  001222   INBUF   007542   INCHAR  010060
INIFLG  001324   INSTER= 104404   INSTR  = 104403   INSTR2  004176
INTTY   012340   KMCM    007361   LIMITS  004324   LINE   007037
LOBITS  004402   LOCK    001220   LOKFLG  001326   LOLIM  004374
LOMAP   012374   LPCNT   001224   LSTERR  001234   MAPCK  022426
MASKX   001244   MASTEK  006152   MCRLF   005702   MCSRX  006102
MDATA   007604   MEMLIM  001304   MEPASS  005743   MERRPC  006232
MERRX   006127   MERR2   005770   MERR3   006015   MESWCH  022454
```

```
MILK    001322          MLOCK   006053          MNEW    006154          MODU    006725
MPASSX  006116          MPFAIL  005705          MQM     005676          MR      005765
MRESET= 004000          MSTCLR= 104412          MTITLE  001000          MTSTN   006140
MTSTPC  006041          MVECX   006110          NEXT    001216          NOACT   007175
NODEV   002704          NUM     006466          OK      002656          ONE     001302
PACT00  001702          PACT01  001706          PACT02  001712          PACT03  001716
PACT04  001722          PACT05  001726          PACT06  001732          PACT07  001736
PACT10  001742          PACT11  001746          PACT12  001752          PACT13  001756
PACT14  001762          PACT15  001766          PACT16  001772          PACT17  001776
PARAM = 104405          PARAM1  004244          PARBIT= 040000          PARERR  004320
PASCNT  001230          PC    =%000007          PERFOR= 004537          PFTAB   005440
POPRO = 012600          POP1SP= 005726          POP2SP= 022626          PRIO    006570
PS    = 177776          PUSHRO= 010046          PUSH1S= 005746          PUSH2S= 024646
QV.FLG  001327          RESREG  005230          RESTAR  005360          RESTRT  003544
RESO5 = 104407          RETURN  001214          ROMCLK= 104414          ROMDAT  022364
ROMMAP  012372          ROM1    023251          RUN     001316          R0    =%000000
R1    =%000001          R2    =%000002          R3    =%000003          R4    =%000004
R5    =%000005          SAVACT  001312          SAVNUM  001314          SAVPC   001276
SAVR0   001260          SAVR1   001262          SAVR2   001264          SAVR3   001266
SAVR4   001270          SAVR5   001272          SAVSP   001274          SAVO5 = 104406
SCOPE = 104400          SCOP1 = 104401          SETBR0  022310          SETBR1  022316
SETBR4  022324          SETBR7  022332          SETC    022340          SETZ    022356
SKIP    002642          SOFTSW  010112          SP    =%000006          SPACNT= 004723
SPEED   007371          STACK = 001200          STAT    001240          STAT1   001366
STAT2   001370          STAT3   001372          STRTSW  001236          SVO5    004412
SWFLG   010056          SWMES   007226          SWMES1  007236          SWR     001202
SWREG   000176          SW00  = 000001          SW01  = 000002          SW02  = 000004
SW03  = 000010          SW04  = 000020          SW05  = 000040          SW06  = 000100
SW07  = 000200          SW08  = 000400          SW09  = 001000          SW10  = 002000
SW11  = 004000          SW12  = 010000          SW13  = 020000          SW14  = 040000
SW15  = 100000          TEMP    001416          TEMP1   001246          TEMP2   001250
TEMp3   001252          TEMP4   001254          TEMP5   001256          TIMER = 104416
TKCSR   001204          TKDBR   001206          TLAST = 022076          TPCSR   001210
TPDBR   001212          TRPOK   004740          TSTNO   001226          TST1    016376
TST10   017742          TST11   020136          TST12   020332          TST13   020526
TST14   020722          TST15   021116          TST16   021312          TST17   021506
TST2    016470          TST20   021702          TST21   022076          TST3    016602
TST4    016776          TST5    017172          TST6    017352          TST7    017546
TTST    003622          TWOSYN= 010000          TYPDAT  005216          TYPE  = 104402
TYPMSG  005116          VEC     006547          VECMAP  012062          WHICH   012054
WRDCNT  004720          WRKO.F  005204          XBX     005010          XCSR    003556
XERR    003600          XHEAD   006237          XLOC    003032          XPASS   003572
XSTATQ  007514          XTSTN   005340          XVEC    003564          ZERO    001300
$CRAP = 177777          $ENDAD  003532          $N    = 000021          $S    = 000023
$Y    = 000017          .BEGIN  003162          .CNVRT  004502          .CONVR  004476
.DATAC  005562          .DELAY  005446          .EOP    003374          .ERRTA  023544
.HLT    004760          .INSTE  004164          .INSTR  004060          .INST1  004100
.MSG    004102          .MSTCL  005476          .PARAM  004204          .PFAIL  005346
.RESO5  004444          .ROMCL  005514          .SAVO5  004404          .SCOPE  003606
.SCOP1  003746          .START  002002          .TIMER  005626          .TRPSR  004726
.TRPTA  001330          .TYPE   003776          .     = 023676
```

ERRORS DETECTED: 0

```
DRLPM   MACY11 27(654)  13-DEC-77  15:46  PAGE 61                    SEQ 0077
DRLPM.P11

 *DRLPM,DRLPM/SOL=DRLPM.MAC,DRLPM
 RUN-TIME:  12 15 0 SECONDS
 CORE USED:  21K
EOF1DRLPMASEQ                00010000      780223            PDP10 411
```