

LPA/DMC-11

DIAGNOSTIC TEST 1
MD-11-DRLPL-A

EP-DRLPL-A-DL
COPYRIGHT © 1978
FICHE 1 OF 1

MAR 1978
digital
MADE IN USA

This microfiche card contains a grid of frames, each displaying diagnostic test data. The frames are arranged in approximately 15 rows and 10 columns. Each frame contains a table of data, likely representing test results for various components or systems. The data is organized into columns and rows, with some frames showing headers and footers. The overall layout is a structured grid of information.

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DRLPL-A-D
PRODUCT NAME: LPA/DMC-11 DIAGNOSTIC TEST 1
DATE: JAN 1978
MAINTAINER: DIAGNOSTICS

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such license.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by Digital.

Copyright (C) 1978 by Digital Equipment Corporation

1. ABSTRACT

this diagnostic is one of a series of diagnostics aimed at 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 verify 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.

performs write/read tests on the m8200-yc unibus registers, checks the micro-processor operation, checks out Main Memory, scratch pad memory, the ALU functions as well as interrupts and NPR operation.

NOTE: This diagnostic will run on a KMC11 (MB204), however it is not advised that this diagnostic be used to check a KMC11, rather you should check a KMC11 with the KMC11 diagnostic package.

NOTE: This diagnostic will run on a dmc11, however, it is not recommended that it be used to check a dmc-11.

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] m8200-yc Basic W/R and Micro-processor tests
2. DRLPM [REV] m8200-yc Jump and CROM tests

2. REQUIREMENTS

2.1 EQUIPMENT

Any PDP11 family CPU (except an LSI-11) with minimum 8k memory
ASR 33 (or equivalent)
M8200-YC

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 PROCEDURE

- 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 carriage 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 switches greater than SW 03 in the up position. this would be a fatal error. do not select more active m8200-ycs 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. It continues 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 intermittent errors; SW14=1 will loop on test regardless of error or not error. (SW14=1, SW10=0, SW09=0, SW08=1,0)
3. SW11 Inhibit interactions.
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 possibly 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 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 regardless of how program was started. Also it is important to use this listing along with the information printed on the TTY to completely 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 overlaid. The current parameters in the STATUS TABLE are used when SW07=1 on start up.

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.

m8200-yc must be in the unibus.

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 (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 DRLPL CSR: 175000 VEC: 0300 PASSES: 000001
ERRORS: 000000
```

NOTE: The pass count and error counts are cummulative 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:.

5.3

POWER FAIL TEST (#136)

THIS TEST MAY HANG ON SOME PROCESSORS IF AN M9301 IS PRESENT.
TO AVOID HANGING, SW02 (power on reboot enable) on the M9301
must be in the off position. This test will also fail if the
CPU power fail vector is set to any location other than 24.
If this test hangs or fails due to either reason above, the
following patch may be installed to skip this test:

LOC 33430 WAS 33600 SB 33772

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

DMC00-DMCR17
DMST00-DMST17
(1500)-(1640)

These locations contain the information needed to test up to 16 (decimal) m8200-yics sequentially. 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 CROM
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 (BM373 BOOT ADD)

STAT3: BIT0=1 RUN FREE RUNNING TESTS ON KMC11
BIT1=0 M8200-YC LPA MICRO CODE VERSION 3

B.5 METHOD OF AUTO SIZING

B.5.1 FINDING THE CONTROL STATUS REGISTER.

The auto-sizing routine finds a m8200-yc as follows: It starts at address 160000 and tests all address in increments of 10 up to and including address 167760. If the address does not time out, the following is done, the first CROM address is written to a 125252 then it is read back. If it contains a -1 or 125252 or 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.

B.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 interrupt occurs 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 interrupt occurred; the address to which the m8200-yc interrupted 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.

B.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 lpa's i/o bus.

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

the lpall-kx diagnostic kit will include:

<u>option</u>	<u>group</u>	<u>diag. #</u>	<u>diag. title</u>
lpall-kx	level 2 diag.	md-11-drlpa	lpall-kx system
m8254	"b"	md-11-drm8a	m8254 (jpbm) diag.
aall-k	a	md-11-drlpb	aall-k diag.
	b	md-11-dzaac	aall-k diag.
arll	a	md-11-drlpc	lpa/arll diag. #1
	a	md-11-drlpd	lpa/arll diag. #2
	a	md-11-drlpe	lpa/arll diag. #3
	b	md-11-dzara	arll diag. #1
	b	md-11-dzarb	arll diag. #2
	b	md-11-dzarc	arll diag. #3
drll-k	a	md-11-drlpf	lpa/drll-k diag.
	b	md-11-dzdrp	drll-k diag.
kull-k	a	md-11-drlpg	lpa/kull-k diag.
	b	md-11-dzkwk	kull-k diag.
lpsll	a	md-11-drlph	lpa/lpsll diag. #1
	a	md-11-drlpi	lpa/lpsll diag. #2
	a	md-11-drlpj	lpa/lpsll diag. #3
	b	md-11-dzlpd	lpsll diag. #1
	b	md-11-dzlpd	lpsll diag. #2
	b	md-11-dzlpd	lpsll diag. #3
adll-k	a	md-11-drlpk	lpa/adll-k diag.
	b	md-11-dzadi	adll-k diag.
m8200-yc	b	md-11-dzlpd	lpa/m8200-yc basic micro-cpu r/w test
	b	md-11-dzlpd	lpa/m8200-yc jmp+rom read test

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45

;*MAINDEC-11-DRLPL-A BASIC LPA-M8200-YC CONTROLLER TEST
;*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-DRLPL-A BASIC LPA-M8200-YC CONTROLLER TEST"
; 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
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

SW15=100000 ;=1, HALT ON ERROR
SW14=40000 ;=1, LOOP ON CURRENT TEST
SW13=20000 ;=1, INHIBIT ERROR TIMEOUT
SW12=10000 ;=1, DELETE TIMEOUT/BELL ON ERROR.
SW11=4000 ;=1, INHIBIT ITERATIONS
SW10=2000 ;=1, ESCAPE TO NEXT TEST ON ERROR
SW09=1000 ;=1, LOOP WITH CURRENT DATA
SW08=400 ;=1, LOOP ON ERROR
SW07=200 ;=1, USE CURRENT M8200-YC PARAMETERS, =0, AUTOSIZE M8200-YC
SW06=100 ;=1, HALT BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION

; RESELECT M8200-YC'S TO BE TESTED (ACTIVE)
; LOCK ON TEST SELECT
; RESTART PROGRAM AT SELECTED TEST
; INPUT M8200-YC PARAMETERS
SW05=40
SW04=20
SW03=10
SW02=4
SW01=2
SW00=1

```

46
47
48           ;REGISTER DEFINITIONS
49           ;-----
50
51           000000      R0=%0           ;GENERAL REGISTER
52           000001      R1=%1           ;GENERAL REGISTER
53           000002      R2=%2           ;GENERAL REGISTER
54           000003      R3=%3           ;GENERAL REGISTER
55           000004      R4=%4           ;GENERAL REGISTER
56           000005      R5=%5           ;GENERAL REGISTER
57           000006      SP=%6          ;PROCESSOR STACK POINTER
58           000007      PC=%7          ;PROGRAM COUNTER
59
60           ;LOCATION EQUIVALENCIES
61           ;-----
62
63           177776      PS=177776      ;PROCESSOR STATUS WORD
64           001200      STACK=1200     ;START OF PROCESSOR STACK
65
66           ;INSTRUCTION DEFINITIONS
67           ;-----
68
69           005746      PUSH1SP=5746   ;DECREMENT PROCESSOR STACK 1 WORD
70           005726      POP1SP=5726    ;INCREMENT PROCESSOR STACK 1 WORD
71           010046      PUSHRO=10046   ;SAVE R0 ON STACK
72           012600      POPRO=12600    ;RESTORE R0 FROM STACK
73           024646      PUSH2SP=24646 ;DECREMENT STACK TWICE
74           022626      POP2SP=22626  ;INCREMENT STACK TWICE
75           .EQUIV EMT,HLT ;BASIC DEFINITION OF ERROR CALL
76
77           ;BIT DEFINITIONS
78           ;-----
79
80           100000      BIT15=100000
81           040000      BIT14=40000
82           020000      BIT13=20000
83           010000      BIT12=10000
84           004000      BIT11=4000
85           002000      BIT10=2000
86           001000      BIT9=1000
87           000400      BIT8=400
88           000200      BIT7=200
89           000100      BIT6=100
90           000040      BIT5=40
91           000020      BIT4=20
92           000010      BIT3=10
93           000004      BIT2=4
94           000002      BIT1=2
95           000001      BIT0=1
96
97

```

98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
(2)
(2)
137
138
139
140
141
142
143

000000

000024 000024
000026 005346
000030 000340
000032 004760
000034 000340
000036 004726
000036 000340
000040 000040
000040 000000
000042 000000
000044 000000
000046 003532
000052 000052
000052 000000

000174 000174
000176 000000
000176 000000

000200 000200 002002
000137 000137

001000 001000
005377 040515 047111
001025 102 051501 041511

001200

001200 177570
001202 177570

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

=0
STANDARD INTERRUPT VECTORS

=24
.PFAIL ;POWER FAIL HANDLER
340 ;SERVICE AT LEVEL 7
.HLT ;ERROR HANDLER
340 ;SERVICE AT LEVEL 7
.TRPSRV ;GENERAL HANDLER DISPATCH SERVICE
340 ;SERVICE AT LEVEL 7
=40
0 ;SAVE FOR ACT-11 OR XXDP
0 ;RETURN ADDRESS IF UNDER ACT-11 OR XXDP
0 ;SAVE FOR ACT-11 OR XXDP
SENDAD ;FOR USE WITH ACT-11 OR XXDP
=52
0 ;ACT-11 PROGRAM CHARACTERISTICS
=174
DISPREG:0 ;SOFTWARE DISPLAY REGISTER
SWREG: 0 ;SOFTWARE SWITCH REGISTER

=200
JMP .START ;GO TO sTART OF PROGRAM

=1000
MTITLE: .ASCII <377><12>/MAINDEC-11-DRLPL-A/<377>
.ASCIZ /BASIC LPA-M8200-YC CONTROLLER TEST/<377>

=1200
;INDIRECT POINTERs TO sWITCH REGISTER AND LIGHT DISPLAY

DISPLAY:177570
SWR: 177570

```

144
145 ;INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS
146 ;-----
147
148 001204 177560 TKCSR: 177560 ; TELETYPE KEYBOARD CONTROL REGISTER
149 001206 177562 TKDBR: 177562 ; TELETYPE KEYBOARD DATA BUFFER
150 001210 177564 TPCSR: 177564 ; TELEPRINTER CONTROL REGISTER
151 001212 177566 TPDBR: 177566 ; TELEPRINTER DATA BUFFER
152
153 ;PROGRAM CONTROL PARAMETERS
154 ;-----
155
156 001214 000000 RETURN: 0 ; SCOPE ADDRESS FOR LOOP ON TEST
157 001216 000000 NEXT: 0 ; ADDRESS OF NEXT TEST TO BE EXECUTED
158 001220 000000 LOCK: 0 ; ADDRESS FOR LOCK ON CURRENT DATA
159 001222 000003 ICOUNT: 3 ; NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE EXECUTED
160 001224 000000 LPCNT: 0 ; NUMBER OF ITERATIONS COMPLETED
161 001226 000000 TSTNO: 0 ; NUMBER OF TEST IN PROGRESS
162 001230 000000 PASCNT: 0 ; NUMBER OF PASSES COMPLETED
163 001232 000000 ERRCNT: 0 ; TOTAL NUMBER OF ERRORS
164 001234 000000 LSTERR: 0 ; PC OF LAST ERROR CALL
165
166 ;PROGRAM VARIABLES
167 ;-----
168
169 001236 000000 STRTSW: 0 ; SWITCHES AT START OF PROGRAM
170 001240 000000 STAT: 0 ; DM STATUS WORD STORAGE
171 001242 000000 CLKX: 0
172 001244 000000 MASKX: 0
173 001246 000000 TEMP1: 0 ; TEMPORARY STORAGE
174 001250 000000 TEMP2: 0 ; TEMPORARY STORAGE
175 001252 000000 TEMP3: 0 ; TEMPORARY STORAGE
176 001254 000000 TEMP4: 0 ; TEMPORARY STORAGE
177 001256 000000 TEMP5: 0 ; TEMPORARY STORAGE
178 001260 000000 SAVR0: 0 ; R0 STORAGE
179 001262 000000 SAVR1: 0 ; R1 STORAGE
180 001264 000000 SAVR2: 0 ; R2 STORAGE
181 001266 000000 SAVR3: 0 ; R3 STORAGE
182 001270 000000 SAVR4: 0 ; R4 STORAGE
183 001272 000000 SAVR5: 0 ; R5 STORAGE
184 001274 000000 SAVSP: 0 ; STACK POINTER STORAGE
185 001276 000000 SAVPC: 0 ; PROGRAM COUNTER STORAGE
186 001300 000000 ZERO: 0
187 001302 000001 ONE: 1
188 001304 000000 MEMLIM: 0 ; HIGHEST LOCATION FOR NPR'S
189 001306 000001 DMACTV: .BLKW 1 ; M8200-YC'S SELECTED ACTIVE.
190 001310 000001 DMNUM: .BLKW 1 ; OCTAL NUMBER OF M8200-YC'S.
191 001312 000001 SAVACT: .BLKW 1 ; ORIGINAL ACTV DEVICES
192 001314 000001 SAVNUM: .BLKW 1 ; WORKABLE NUMBER
193 001316 000000 RUN: 0 ; POINTER TO RUNNING DEVICE.
194 .EVEN
195 001320 001472 CREAM: DM.MAP-6 ; TABLE POINTER.
196 001322 001676 MILK: CNT.MAP-4 ; TABLE POINTER

```

```

197
198 ;PROGRAM CONTROL FLAGS
199 ;-----
200
201 001324 000 INIFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG
202 001325 000 ERRFLG: .BYTE 0 ;ERROR OCCURED FLAG
203 001326 000 LOKFLG: .BYTE 0 ;LOCK ON CURRENT TEST FLAG
204 001327 000 QV.FLG: .BYTE 0 ;QUICK VERIFY FLAG.
205 ;ON FIRST PASS OF EACH M8200-YC ITERATIONS WILL BE SUPPR
206 .EVEN
207
208 ;DEFINITIONS FOR TRAP SUBROUTINE CALLS
209 ;POINTERS TO SUBROUTINES CAN BE FOUND
210 ;IN THE TABLE IMMEDIATLY FOLLOWING THE DEFINITIONS
211
212 ;:*****
213 ;:-----
214 ;TRPTAB:
215 001330 104400 SCOPE=TRAP+0 ;CALL TO SCOPE LOOP AND ITERATION HANDLER
216 001330 003606 .SCOPE
217 104401 SCOP1=TRAP+1 ;CALL TO 100P ON CURRENT DATA HANDLER
218 001332 003746 .SCOP1
219 104402 TYPE=TRAP+2 ;CALL TO TELETYPE OUTPUT ROUTINE
220 001334 003776 .TYPE
221 104403 INSTR=TRAP+3 ;CALL TO ASCII STRING INPUT ROUTINE
222 001336 004060 .INSTR
223 104404 INSTER=TRAP+4 ;CALL TO INPUT ERROR HANDLER
224 001340 004164 .INSTER
225 104405 PARAM=TRAP+5 ;CALL TO NUMERICAL DATA INPUT ROUTINE
226 001342 004204 .PARAM
227 104406 SAVOS=TRAP+6 ;CALL TO REGISTER SAVE ROUTINE
228 001344 004404 .SAVOS
229 104407 RESOS=TRAP+7 ;CALL TO REGISTER RESTORE ROUTINE
230 001346 004444 .RESOS
231 104410 CONVRT=TRAP+10 ;CALL TO DATA OUTPUT ROUTINE
232 001350 004476 .CONVRT
233 104411 CNVRT=TRAP+11 ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
234 001352 004502 .CNVRT
235 104412 MSTCLR=TRAP+12 ;CALL TO ISSUE A MASTER CLEAR
236 001354 005476 .MSTCLR
237 104413 DELAY=TRAP+13 ;CALL TO DELAY
238 001356 005446 .DELAY
239 104414 ROMCLK=TRAP+14 ;CALL TO CLOCK ROM ONCE
240 001360 005514 .ROMCLK
241 104415 DATACLK=TRAP+15 ;CALL TO CLK DATA
242 001362 005562 .DATACLK
243 104416 TIMER=TRAP+16 ;CALL TO DELAY A CLOCK TICK
244 001364 005626 .TIMER
245
246 ;:*****
247
  
```

```

248 ;M8200-YC CONTROL INDICATORS FOR CURRENT M8200-YC UNDER TEST
249 -----
250
251 001366 000000 STAT1: 0
252 001370 000000 STAT2: 0
253 001372 000000 STAT3: 0
254
255 ;M8200-YC VECTOR AND REGISTER INDIRECT POINTERS
256 -----
257
258 001374 000000 DMRVEC: 0 ; POINTER TO M8200-YC RECEIVER INTERRUPT VECTOR
259 001376 000000 DMRLVL: 0 ; POINTER TO M8200-YC RECEIVER INTERRUPT SERVICE PS
260 001400 000000 DMTVEC: 0 ; POINTER TO M8200-YC TRANSMITTER INTERRUPT VECTOR
261 001402 000000 DMTLVL: 0 ; POINTER TO M8200-YC TRANSMITTER INTERRUPT SERVICE PS
262 001404 000000 DMCSR: 0 ; POINTER TO M8200-YC CONTROL STATUS REGISTER
263 001406 000000 DMCSRH: 0 ; POINTER TO M8200-YC CONTROL STATUS REGISTER HIGH BYTE.
264 001410 000000 DMCTL: 0 ; POINTER TO M8200-YC CONTROL OUT REGISTER
265 001412 000000 DMP04: 0 ; POINTER TO M8200-YC PORT REGISTER(SEL 4)
266 001414 000000 DMP06: 0 ; POINTER TO M8200-YC PORT REGISTER(SEL 6)
267
268 ;TEMP STORAGE
269 -----
270
271 001416 000000 TEMP: 0
272 001460 .=. +40
273
274 ;M8200-YC STATUS TABLE AND ADDRESS ASSIGNMENTS
275 -----
276
277 . =1500
278 001500 DM.MAP:
279 001500 000001 DMCr00: .BLKW 1 ; CONTROL STATUS REGISTER FOR M8200-YC NUMBER 00
280 001502 000001 DMS100: .BLKW 1 ; VECTOR FOR M8200-YC NUMBER 00
281 001504 000001 DMS200: .BLKW 1 ; DDCMP LINE# FOR M8200-YC NUMBER 00
282 001506 000001 DMS300: .BLKW 1 ; 3RD STATUS WORD
283
284 001510 000001 DMCr01: .BLKW 1 ; CONTROL STATUS REGISTER FOR M8200-YC NUMBER 01
285 001512 000001 DMS101: .BLKW 1 ; VECTOR FOR M8200-YC NUMBER 01
286 001514 000001 DMS201: .BLKW 1 ; DDCMP LINE# FOR M8200-YC NUMBER 01
287 001516 000001 DMS301: .BLKW 1 ; 3RD STATUS WORD
288
289 001520 000001 DMCr02: .BLKW 1 ; CONTROL STATUS REGISTER FOR M8200-YC NUMBER 02
290 001522 000001 DMS102: .BLKW 1 ; VECTOR FOR M8200-YC NUMBER 02
291 001524 000001 DMS202: .BLKW 1 ; DDCMP LINE# FOR M8200-YC NUMBER 02
292 001526 000001 DMS302: .BLKW 1 ; 3RD STATUS WORD
293
294 001530 000001 DMCr03: .BLKW 1 ; CONTROL STATUS REGISTER FOR M8200-YC NUMBER 03
295 001532 000001 DMS103: .BLKW 1 ; VECTOR FOR M8200-YC NUMBER 03
296 001534 000001 DMS203: .BLKW 1 ; DDCMP LINE# FOR M8200-YC NUMBER 03
297 001536 000001 DMS303: .BLKW 1 ; 3RD STATUS WORD
298
299 001540 000001 DMCr04: .BLKW 1 ; CONTROL STATUS REGISTER FOR M8200-YC NUMBER 04
300 001542 000001 DMS104: .BLKW 1 ; VECTOR FOR M8200-YC NUMBER 04
301 001544 000001 DMS204: .BLKW 1 ; DDCMP LINE# FOR M8200-YC NUMBER 04
  
```

302	001546	000001	DMS304: .BLKW	1	;3RD STATUS WORD
303					
304	001550	000001	DMCR05: .BLKW	1	;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 05
305	001552	000001	DMS105: .BLKW	1	;VECTOR FOR M8200-YC NUMBER 05
306	001554	000001	DMS205: .BLKW	1	;DDCMP LINE# FOR M8200-YC NUMBER 05
307	001556	000001	DMS305: .BLKW	1	;3RD STATUS WORD
308					
309	001560	000001	DMCR06: .BLKW	1	;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 06
310	001562	000001	DMS106: .BLKW	1	;VECTOR FOR M8200-YC NUMBER 06
311	001564	000001	DMS206: .BLKW	1	;DDCMP LINE# FOR M8200-YC NUMBER 06
312	001566	000001	DMS306: .BLKW	1	;3RD STATUS WORD
313					
314	001570	000001	DMCR07: .BLKW	1	;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 07
315	001572	000001	DMS107: .BLKW	1	;VECTOR FOR M8200-YC NUMBER 07
316	001574	000001	DMS207: .BLKW	1	;DDCMP LINE# FOR M8200-YC NUMBER 07
317	001576	000001	DMS307: .BLKW	1	;3RD STATUS WORD
318					
319	001600	000001	DMCR10: .BLKW	1	;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 10
320	001602	000001	DMS110: .BLKW	1	;VECTOR FOR M8200-YC NUMBER 10
321	001604	000001	DMS210: .BLKW	1	;DDCMP LINE# FOR M8200-YC NUMBER 10
322	001606	000001	DMS310: .BLKW	1	;3RD STATUS WORD
323					
324	001610	000001	DMCR11: .BLKW	1	;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 11
325	001612	000001	DMS111: .BLKW	1	;VECTOR FOR M8200-YC NUMBER 11
326	001614	000001	DMS211: .BLKW	1	;DDCMP LINE# FOR M8200-YC NUMBER 11
327	001616	000001	DMS311: .BLKW	1	;3RD STATUS WORD
328					
329	001620	000001	DMCR12: .BLKW	1	;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 12
330	001622	000001	DMS112: .BLKW	1	;VECTOR FOR M8200-YC NUMBER 12
331	001624	000001	DMS212: .BLKW	1	;DDCMP LINE# FOR M8200-YC NUMBER 12
332	001626	000001	DMS312: .BLKW	1	;3RD STATUS WORD
333					
334	001630	000001	DMCR13: .BLKW	1	;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 13
335	001632	000001	DMS113: .BLKW	1	;VECTOR FOR M8200-YC NUMBER 13
336	001634	000001	DMS213: .BLKW	1	;DDCMP LINE# FOR M8200-YC NUMBER 13
337	001636	000001	DMS313: .BLKW	1	;3RD STATUS WORD
338					
339	001640	000001	DMCR14: .BLKW	1	;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 14
340	001642	000001	DMS114: .BLKW	1	;VECTOR FOR M8200-YC NUMBER 14
341	001644	000001	DMS214: .BLKW	1	;DDCMP LINE# FOR M8200-YC NUMBER 14
342	001646	000001	DMS314: .BLKW	1	;3RD STATUS WORD
343					
344	001650	000001	DMCR15: .BLKW	1	;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 15
345	001652	000001	DMS115: .BLKW	1	;VECTOR FOR M8200-YC NUMBER 15
346	001654	000001	DMS215: .BLKW	1	;DDCMP LINE# FOR M8200-YC NUMBER 15
347	001656	000001	DMS315: .BLKW	1	;3RD STATUS WORD
348					
349	001660	000001	DMCR16: .BLKW	1	;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 16
350	001662	000001	DMS116: .BLKW	1	;VECTOR FOR M8200-YC NUMBER 16
351	001664	000001	DMS216: .BLKW	1	;DDCMP LINE# FOR M8200-YC NUMBER 16
352	001666	000001	DMS316: .BLKW	1	;3RD STATUS WORD
353					
354	001670	000001	DMCR17: .BLKW	1	;CONTROL STATUS REGISTER FOR M8200-YC NUMBER 17
355	001672	000001	DMS117: .BLKW	1	;VECTOR FOR M8200-YC NUMBER 17

L02

DRPLP MACY11 27(654) 13-DEC-77 11:41 PAGE 8
DRPLP.P11 PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

SEQ 0024

356 001674 000001
357 001676 000001
358
359 001700 000000

DMS217: .BLKW 1
DMS317: .BLKW 1
DM.END: 000000

:DDCMP LINE# FOR M8200-YC NUMBER 17
:3RD STATUS WORD

```

360
361          ;MB200-YC PASS COUNT AND ERROR COUNT TABLE
362          ;-----
363
364          CNT.MAP:
365          001702 000000          PACT00: 0          ;PASS COUNT FOR MB200-YC NUMBER 00
366          001704 000000          ERCT00: 0          ;ERROR COUNT FOR MB200-YC NUMBER 00
367
368          001706 000000          PACT01: 0          ;PASS COUNT FOR MB200-YC NUMBER 01
369          001710 000000          ERCT01: 0          ;ERROR COUNT FOR MB200-YC NUMBER 01
370
371          001712 000000          PACT02: 0          ;PASS COUNT FOR MB200-YC NUMBER 02
372          001714 000000          ERCT02: 0          ;ERROR COUNT FOR MB200-YC NUMBER 02
373
374          001716 000000          PACT03: 0          ;PASS COUNT FOR MB200-YC NUMBER 03
375          001720 000000          ERCT03: 0          ;ERROR COUNT FOR MB200-YC NUMBER 03
376
377          001722 000000          PACT04: 0          ;PASS COUNT FOR MB200-YC NUMBER 04
378          001724 000000          ERCT04: 0          ;ERROR COUNT FOR MB200-YC NUMBER 04
379
380          001726 000000          PACT05: 0          ;PASS COUNT FOR MB200-YC NUMBER 05
381          001730 000000          ERCT05: 0          ;ERROR COUNT FOR MB200-YC NUMBER 05
382
383          001732 000000          PACT06: 0          ;PASS COUNT FOR MB200-YC NUMBER 06
384          001734 000000          ERCT06: 0          ;ERROR COUNT FOR MB200-YC NUMBER 06
385
386          001736 000000          PACT07: 0          ;PASS COUNT FOR MB200-YC NUMBER 07
387          001740 000000          ERCT07: 0          ;ERROR COUNT FOR MB200-YC NUMBER 07
388
389          001742 000000          PACT10: 0         ;PASS COUNT FOR MB200-YC NUMBER 10
390          001744 000000          ERCT10: 0         ;ERROR COUNT FOR MB200-YC NUMBER 10
391
392          001746 000000          PACT11: 0         ;PASS COUNT FOR MB200-YC NUMBER 11
393          001750 000000          ERCT11: 0         ;ERROR COUNT FOR MB200-YC NUMBER 11
394
395          001752 000000          PACT12: 0         ;PASS COUNT FOR MB200-YC NUMBER 12
396          001754 000000          ERCT12: 0         ;ERROR COUNT FOR MB200-YC NUMBER 12
397
398          001756 000000          PACT13: 0         ;PASS COUNT FOR MB200-YC NUMBER 13
399          001760 000000          ERCT13: 0         ;ERROR COUNT FOR MB200-YC NUMBER 13
400
401          001762 000000          PACT14: 0         ;PASS COUNT FOR MB200-YC NUMBER 14
402          001764 000000          ERCT14: 0         ;ERROR COUNT FOR MB200-YC NUMBER 14
403
404          001766 000000          PACT15: 0         ;PASS COUNT FOR MB200-YC NUMBER 15
405          001770 000000          ERCT15: 0         ;ERROR COUNT FOR MB200-YC NUMBER 15
406
407          001772 000000          PACT16: 0         ;PASS COUNT FOR MB200-YC NUMBER 16
408          001774 000000          ERCT16: 0         ;ERROR COUNT FOR MB200-YC NUMBER 16
409
410          001776 000000          PACT17: 0         ;PASS COUNT FOR MB200-YC NUMBER 17
411          002000 000000          ERCT17: 0         ;ERROR COUNT FOR MB200-YC NUMBER 17
412

```

413

FORMAT OF STATUS TABLE

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

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

B03

DLPL MACY11 27(654) 13-DEC-77 11:41 PAGE 11
DLPL.P11 PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

SEQ 0027

```

468
469
470
471
472
473
474
475
476 002002 012737 000340 177776 .START: MOV #340,PS ;LOCK OUT INTERRUPTS
477 002010 012706 001200 MOV #STACK,SP ;SET UP STACK
478 002014 012737 005346 000024 MOV #.PFAIL,2#24 ;SET UP POWER FAIL VECTOR
479 002022 013737 001310 001314 MOV #DMNUM,SAVNUM ;SAVE NUMBER OF DEVICES IN SYSTEM.
480 002030 005037 010056 CLR SWFLG ;CLEAR SOFT TIMEOUT FLAG
481 002034 105037 001325 CLR# ERFLG ;CLEAR ERROR FLAG
482 002040 105037 001327 CLR# QV.FLG ;ZERO QUICK VERIFY FLAG
483 002044 012737 001470 001320 MOV #DM.MAP-10,CREAM ;GET MAP POINTER.
484 002052 012737 001676 001322 MOV #CNT.MAP-4,MILK ;GET PASS COUNT MAP POINTER
485 002060 012737 100000 001316 MOV #BIT15,RUN ;POINT POINTER TO FIRST DEVICE.
486 002066 012700 001702 MOV #CNT.MAP,RO ;PASS COUNT POINTER TO RO
487 002072 005020 23$: CLR (RO)+ ;CLEAR TABLE
488 002074 022700 002002 CMP #CNT.MAP+100,RO ;DONE YET?
489 002100 001374 BNE 23$ ;KEEP GOING
490 002102 005037 001234 CLR LSTERR ;CLEAR LAST ERROR POINTER
491 002106 012737 000001 001226 MOV #1,TSTNO ;SET UP FOR TEST 1
492 002114 012737 002002 001214 MOV #.START,RETURN ;SET UP FOR POWER FAIL BEFORE
493 ;TESTING STARTS
494 002122 013746 000006 MOV 2#6,-(SP) ;SAVE CURRENT VECTORS
495 002126 013746 000004 MOV 2#4,-(SP)
496 002132 012737 002166 000004 MOV #6$,2#4 ;SET UP FOR TIMEOUT
497 002140 012737 177570 001202 MOV #177570,SWR ;SET SWR TO HARD SWR ADDRESS
498 002146 012737 177570 001200 MOV #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
499 002154 022777 177777 177020 CMP #-1,2SWR ;REFERENCE HARDWARE SWITCH REGISTER
500 002162 001402 BEQ 6$+2 ;IF = -1 USE SOFT SWR ANYWAY
501 002164 000407 BR 7$ ;IF IT EXISTS AND NOT = -1 USE HARD SWR
502 002166 022626 6$: CMP (SP)+(SP)+ ;ADJUST STACK
503 002170 012737 000176 001202 MOV #SWREG,SWR ;POINTER TO SOFT SWR
504 002176 012737 000174 001200 MOV #DISPREG,DISPLAY ;POINTER TO SOFT DISPLAY REG
505 002204 012637 000004 7$: MOV (SP)+,2#4 ;RESTORE VECTORS
506 002210 012637 000006 MOV (SP)+,2#6
507 002214 105737 001324 TSTB INIFLG ;HAS INITIALIZATION BEEN PERFORMED
508 002220 001012 BNE 20$ ;BR IF YES
509 002222 022737 003532 000042 CMP #SENDAD,2#42 ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
510 002230 001406 BEQ 20$
511 002232 104402 001000 TYPE ,MTITLE ;TYPE TITLE MESSAGE
512 002236 104402 036450 TYPE ,ROM1 ;TYPE VERSION MESSAGE
513 002242 104402 035102 TYPE ,MESWCH ;TYPE SWITCH 7 MESSAGE
514 002246 004737 007646 20$: JSR PC,CKSWR ;CHECK FOR SOFT SWR
515 002252 017737 176724 001236 MOV 2SWR,STRTSW ;STORE STARTING SWITCHES
516 002260 005737 000042 TST 2#42 ;IS IT RUNNING IN AUTO MODE?
517 002264 001402 BEQ .+6 ;BR IF NO
518 002266 005037 001236 CLR STRTSW ;IF YES, CLEAR SWITCHES
519 002272 032737 000001 001236 BIT #SW00,STRTSW ;IF SW00=1, QUESTIONS ARE ASKED.
520 002300 001012 BNE 17$ ;BR IF SW00=1
521 002302 105737 001236 TSTB STRTSW ;BIT7=1??

```

```

522 002306 100007          BPL      17$          ;BR IF SW07=0
523 002310 005737 001306  TST      DMACTV      ;ARE ANY DEVICES SELECTED?
524 002314 001006          BNE      16$          ;BR IF YES
525 002316 104402 007175  TYPE,   NOACT          ;NO DEVICES SELECTED.
526 002322 000000          HALT                    ;STOP THE SHOW
527 002324 000776          BR       -2           ;DISQUALIFY CONTINUE SWITCH
528 002326 004737 010552  17$:   JSR      PC.AUTO.SIZE ;GO DO THE AUTO SIZE
529 002328 105737 001324  16$:   TSTB     INIFLG      ;FIRST TIME?
530 002336 001410          BEQ      21$          ;BR IF YES
531 002340 105737 001236  TSTB     STRTSW      ;IF USING SAME PARAMETERS DONT TYPE MAP
532 002344 100431          BMI      1$          ;
533 002346 032737 000006 001236  BIT      #BIT1!BIT2,STRTSW ;IS TEST NO. OR LOCK SELECTED
534 002354 001403          BEQ      24$          ;IF NO THEN TYPE STATUS
535 002356 000424          BR       1$          ;IF YES DO NOT TYPE STATUS
536 002360 005137 001324  21$:   COM      INIFLG      ;SET FLAG
537 002364 104402 006237  24$:   TYPE     XHEAD      ;TYPE HEADER
538 002370 012704 001500  MOV      #DM.MAP,R4   ;SET POINTER
539 002374 010437 001246  5$:   MOV      R4,TEMP1    ;SET ADDRESS
540 002400 012437 001250  MOV      (R4)+,TEMP2  ;SET CSR
541 002404 001411          BEQ      1$          ;ALL DONE IF ZERO
542 002406 012437 001252  MOV      (R4)+,TEMP3  ;SET STAT1
543 002412 012437 001254  MOV      (R4)+,TEMP4  ;SET STAT2
544 002416 012437 001256  MOV      (R4)+,TEMP5  ;SET STAT3
545 002422 104410          CONVRT   XSTATQ      ;TYPE OUT STATUS MAP
546 002424 007514          XSTATQ
547 002426 000762          BR       5$          ;
548 002430 012700 001500  1$:   MOV      #DM.MAP,R0 ;R0 POINTS TO STATUS TABLE
549
550 *****
551 *****
552 *****
553 *****
554 *****
555 *****
556 *****
557 *****
558 *****
559 *****
560 *****
561 *****
562 *****
563 *****
564 *****
565 002434 013746 000004          MOV      @#4,-(SP)    ;SAVE LOC 4
566 002440 013746 000006          MOV      @#6,-(SP)    ;SAVE LOC 6
567 002444 005037 000006          CLR      @#6          ;CLEAR VEC+2
568 002450 005037 001252          CLR      TEMP3       ;CLEAR FLAG
569 002454 005005          CLR      R5          ;R5=0=DMC, R5=-1=KMC
570 002456 011037 001404  AUSTRT: MOV      (R0),DMCSR   ;GET NEXT DMC CSR
571 002462 001564          BEQ      AUDONE      ;BR IF DONE
572 002464 005705          TST      R5          ;DMC OR KMC?
573 002466 001005          BNE      1$          ;BR IF KMC
574 002470 032760 100000 000002  BIT      #BIT15,2(R0) ;CHECK FOR DMC CSR
575 002476 001061          BNE      SKIP        ;SKIP IF NOT DMC

```

576	002500	000404				BR	2\$: ITS A DMC SO CONTINUE
577	002502	032760	100000	000002	1\$:	BIT	#BIT15,2(R0)	: CHECK FOR KMC CSR
578	002510	001454				BEQ	SKIP	: SKIP IF NOT KMC
579	002512	012737	002704	000004	2\$:	MOV	#NODEV,2#4	: SET UP FOR TIMEOUT
580	002520	005705				TST	R5	: DMC OR KMC?
581	002522	001003				BNE	3\$: BR IF KMC
582	002524	012703	000006			MOV	#6,R3	: R3 IS COUNT OF DEVICES BEFORE DMC
583	002530	000402				BR	4\$: GO ON
584	002532	012703	000010		3\$:	MOV	#10,R3	: R3 IS COUNT OF DEVICES BEFORE KMC
585	002536	012702	003020		4\$:	MOV	#DEVTAB,R2	: R2 IS DEVICE TABLE POINTER
586	002542	012701	160010			MOV	#160010,R1	: START WITH ADDRESS 160010
587	002546	005711			FLOAT:	TST	(R1)	: CHECK ADDRESS IN R1
588	002550	111204				MOVB	(R2),R4	: IF NO TIMEOUT, GET NEXT ADDRESS
589	002552	060401				ADD	R4,R1	: IN R1
590	002554	005201				INC	R1	
591	002556	040401				BIC	R4,R1	
592	002560	005703				TST	R3	: ANY MORE DEVICES TO CHECK FOR?
593	002562	001371				BNE	FLOAT	: BR IF YES
594	002564	012737	002710	000004		MOV	#ERR,2#4	: OK ONLY DMC'S ARE LEFT, SET UP FOR TIMEOUT
595	002572	010137	003032			MOV	R1,XLOC	: SAVE FIRST DMC/KMC ADDRESS
596	002576	005705			FY:	TST	R5	: DMC OR KMC?
597	002600	001005				BNE	1\$: BR IF KMC
598	002602	032760	100000	000002		BIT	#BIT15,2(R0)	: CHECK FOR DMC CSR
599	002610	001014				BNE	SKIP	: SKIP IF NOT DMC
600	002612	000404				BR	2\$: ITS A DMC SO CONTINUE
601	002614	032760	100000	000002	1\$:	BIT	#BIT15,2(R0)	: CHECK FOR KMC CSR
602	002622	001407				BEQ	SKIP	: SKIP IF NOT KMC
603	002624	005711			2\$:	TST	(R1)	: CHECK DMC ADDRESS
604	002626	020137	001404			CMP	R1,DMCSR	: DOES IT MATCH
605	002632	001411				BEQ	OK	: BR IF YES
606	002634	062701	000010			ADD	#10,R1	: GET NEXT DMC ADDRESS
607	002640	000756				BR	FY	: DO IT AGAIN
608	002642	062700	000010		SKIP:	ADD	#10,R0	: SKIP TO NEXT CSR IN TABLE
609	002646	011037	001404			MOV	(R0),DMCSR	: GET NEXT CSR
610	002652	001470				BEQ	AUDONE	: BR IF DONE
611	002654	000750				BR	FY	: ELSE CONTINUE
612	002656	062700	000010		OK:	ADD	#10,R0	: SKIP TO NEXT DMC CSR
613	002662	062737	000010	003032		ADD	#10,XLOC	: UPDATE EXPECTED DMC/KMC ADDRESS
614	002670	011037	001404			MOV	(R0),DMCSR	: GET NEXT DMC/KMC CSR
615	002674	001457				BEQ	AUDONE	: BR IF DONE
616	002676	013701	003032			MOV	XLOC,R1	: GET EXPECTED DMC/KMC ADDRESS
617	002702	000735				BR	FY	: CONTINUE
618	002704	122243			NODEV:	CMPB	(R2)+,-(R3)	: ON TIMEOUT, INC R2, DEC R3
619	002706	000002				RTI		: RETURN
620	002710	005737	001252		ERR:	TST	TEMP3	: CHECK FLAG IF = 0 TYPE HEADER
621	002714	001014				BNE	1\$: SKIP HEADER
622	002716	104402				TYPE		: TIMEOUT HEADER MESSAGE
623	002720	007244				CONERR		: CONFIGURATION ERROR!!!!
624	002722	012737	002710	001276		MOV	#ERR,SAVPC	: SAVE PC FOR TIMEOUT
625	002730	104411				CNVRT		: TYPE OUT ERROR PC
626	002732	003000				ERRPC		
627	002734	104402				TYPE		: TYPE REST OF HEADER
628	002736	007323				CNERR		
629	002740	012737	177777	001252		MOV	#-1,TEMP3	: SET FLAG SO IT ONLY GETS TYPED ONCE

```

630 002746 010137 001262 1$: MOV R1,SAVR1 ;SAVE R1 FOR TYPEOUT
631 002752 104410 CONVRT
632 002754 003006 CONTAB ;TYPE CSR VALUES
633 002756 005705 TST R5 ;DMC OR KMC ?
634 002760 001003 BNE 3$ ;BR IF KMC
635 002762 104402 TYPE
636 002764 007344 DMCM
637 002766 000402 BR 4$ ;CONTINUE
638 002770 104402 3$: TYPE
639 002772 007361 KMCM
640 002774 022626 4$: CMP (SP)+,(SP)+ ;ADJUST STACK
641 002776 000727 BR OK ;BR TO gET OUT
642 003000 000001 ERRPC: 1
643 003002 006 002 .BYTE 6,2
644 003004 001276 SAVPC
645 003006 000002 CONTAB: 2
646 003010 006 004 .BYTE 6,4
647 003012 003032 XLOC
648 003014 006 002 .BYTE 6,2
649 003016 001404 DMCSR
650 003020 007 DEVTAB: .BYTE 7 ;DJ
651 003021 017 .BYTE 17 ;DH
652 003022 007 .BYTE 7 ;DQ
653 003023 007 .BYTE 7 ;DU
654 003024 007 .BYTE 7 ;DUP
655 003025 007 .BYTE 7 ;LK
656 003026 007 .BYTE 7 ;DMC
657 003027 007 .BYTE 7 ;DZ
658 003030 007 .BYTE 7 ;KMC
659 003032 003032 .EVEN
660 003034 000000 XLOC: 0
661 003034 005705 AUDONE: TST R5 ;DMC?
662 003036 001005 BNE 1$ ;BR IF KMC AND ALL DONE
663 003040 012705 177777 MOV #-1,R5 ;SET R5 TO -1 (KMC)
664 003044 012700 001500 MOV #DM.MAP,RO ;RESET RO TO START OF TABLE
665 003050 000602 BR AUSTRT ;GO DO KMC'S
666 003052 012637 000006 1$: MOV (SP)+,@#6 ;RESTORE LOC 6
667 003056 012637 000004 MOV (SP)+,@#4 ;RESTORE LOC 4
668 003062 032737 000010 001236 BIT #SW03,STRTSW ;SELECT SPECIFIC DEVICES??
669 003070 001422 BEQ 3$ ;BR IF NO.
670 003072 104402 006154 TYPE MNEW ;TYPE THE MESSAGE.
671 003076 005000 CLR RO ;ZERO DATA LIGHTS
672 003100 000000 HALT ;WAIT FOR USER TO TELL WHAT DEVICES TO rUN
673 003102 027737 176074 001312 CMP @SWR,SAVACT ;IS THE NUMBER VALID?
674 003110 101404 BLOS 2$ ;BR IF NUMBER IS OK.
675 003112 104402 006015 TYPE ,MERR3 ;TELL USER OF INVALID NUMBER.
676 003116 000000 HALT ;STOP EVERY THING.
677 003120 000776 BR -2 ;RESTART THE PROGRAM AGAIN.
678 003122 017737 176054 001306 2$: MOV @SWR,DMACTV ;GET NEW DEVICE PATTERN
679 003130 013700 001306 MOV DMACTV,RO ;SHOW THE USER WHAT HE SELECTED.
680 003134 000000 HALT ;CONTINUE DYNAMIC SWITCHES.
681 003136 012700 000300 3$: MOV #300,RO ;PREPARE TO CLEAR THE FLOATING
682 003142 012701 000302 MOV #302,R1 ;VECTOR AREA. 300-776
683 003146 010120 4$: MOV R1,(RO)+ ;START PUTTING "PC+2 - HALT"
    
```

```

684 003150 005021          CLR      (R1)+      ; IN VECTOR AREA.
685 003152 022021          CMP      (R0)+,(R1)+ ; POP POINTERS
686 003154 022700 001000  CMP      #1000,RO    ; ALL DONE??
687 003160 001372          BNE      4$          ; BR IF NO.
688
689                               ; TEST START AND RESTART
690                               ; -----
691
692 003162 012706 001200    .BEGIN: MOV      #STACK,SP    ; SET UP STACK
693 003166 013746 000006    MOV      @#6,-(SP)      ; SAVE LOC 6
694 003172 013746 000004    MOV      @#4,-(SP)      ; SAVE LOC 4
695 003176 005000          CLR      RO            ; START AT 0
696 003200 012737 003244 000004  MOV      #2$,@#4        ; SET UP FOR TIME OUT
697 003206 005037 000006    CLR      @#6          ; TO AUTOSIZE MEMORY
698 003212 005720          6$: TST      (R0)+        ; CHECK ADDRESS IN RO
699 003214 022700 157776    CMP      #157776,RO    ; IS IT AT LEAST 28K
700 003220 001374          BNE      6$          ; BR IF NO
701 003222 162700 007776    SUB      #7776,RO      ; SAVE 2K FOR MONITORS
702 003226 010037 001304    7$: MOV      RO,MEMLIM  ; STORE MEMORY LIMIT
703 003232 012637 000004    MOV      (SP)+,@#4     ; RESTORE LOC 4
704 003236 012637 000006    MOV      (SP)+,@#6     ; RESTORE LOC 6
705 003242 000413          BR       10$         ; CONTINUE
706 003244 022626          2$: CMP      (SP)+,(SP)+ ; ADJUST STACK
707 003246 162700 000004    SUB      #4,RO         ; GET LAST GOOD ADDRESS
708 003252 162700 007776    SUB      #7776,RO      ; SAVE 2K FOR MONITORS
709 003256 022700 030000    CMP      #30000,RO    ; IS IT 8K?
710 003262 001361          BNE      7$          ; BR IF NO
711 003264 012700 037400    MOV      #37400,RO    ; IF 8K DON'T SAVE 2K
712 003270 000756          BR       7$
713 003272 012737 000340 177776 10$: MOV      #340,PS      ; LOCK OUT INTERRUPTS
714 003300 032737 000004 001236  BIT      #BIT2,STRTSW ; CHECK FOR LOCK ON TEST
715 003306 001411          BEQ      1$          ; BR IF NO LOCK DESIRED.
716 003310 104402 006053    TYPE    ,MLOCK        ; TYPE LOCK SELECTED.
717 003314 012737 000240 003622  MOV      #NOP,TTST     ; ADJUST SCOPE ROUTINE.
718 003322 012737 000240 003624  MOV      #NOP,TTST+2   ; SET UP TO LOCK
719 003330 000406          BR       3$          ; CONTINUE ALONG.
720 003332 013737 003740 003622 1$: MOV      BRW,TTST     ; PREPARE NORMAL SCOPE ROUTINE
721 003340 013737 003742 003624  MOV      BRX,TTST+2    ; LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
722 003346 012737 010120 001214 3$: MOV      #CYCLE,RETURN ; START AT "CYCLE" FIND WHICH DEVICE TO TEST
723 003354 032737 000002 001236 4$: BIT      #SW01,STRTSW ; IS TEST NO. SELECTED?
724 003362 001002          BNE      5$          ; BR IF YES
725 003364 104402 005765    TYPE    ,MR           ; TYPE R
726 003370 000177 175620    5$: JMP      @RETURN   ; START TESTING

```

```

727                                     ;END OF PASS
728                                     ;TYPE NAME OF TEST
729                                     ;UPDATE PASS COUNT
730                                     ;CHECK FOR EXIT TO ACT-11
731                                     ;RESTART TEST
732
733 003374 000005                                     .EOP: RESET                                     ;MAKE THE WORLD CLEAN AGAIN.
734 003376 005037 001234                             CLR          LSTERR                                     ;CLEAR LAST ERROR PC
735 003402 105037 001325                             CLR          ERRFLG                                    ;CLEAR ERROR FLAG
736 003406 005237 001230                             INC          PASCNT                                    ;UPDATE PASS COUNT
737 003412 013777 001230 175560                    MOV          PASCNT, @DISPLAY                         ;DISPLAY PASS COUNT
738 003420 104402 005743                             TYPE        ,MEPASS                                  ;TYPE END PASS
739 003424 104402 006102                             TYPE        ,MCSRX                                   ;TYPE CSR
740 003430 104411 003556                             CNVRT       ,XCSR                                    ;SHOW IT
741 003434 104402 006110                             TYPE        ,MVECX                                   ;TYPE VECTOR
742 003440 104411 003564                             CNVRT       ,XVEC                                    ;SHOW IT
743 003444 104402 006116                             TYPE        ,MPASSX                                  ;TYPE PASSES
744 003450 104411 003572                             CNVRT       ,XPASS                                    ;SHOW IT
745 003454 104402 006127                             TYPE        ,MERRX                                   ;TYPE ERRORS
746 003460 104411 003600                             CNVRT       ,XERR                                    ;SHOW IT
747 003464 013700 001322                             MOV          MILK,RO                                  ;GET POINTER TO PASS COUNT
748 003470 013720 001230                             MOV          PASCNT,(RO)+                             ;STORE PASS COUNT FOR THIS M8200-YC
749 003474 013720 001232                             MOV          ERRCNT,(RO)+                             ;STORE ERROR COUNT FOR THIS M8200-YC
750 003500 005337 001314                             DEC          SAVNUM                                   ;ARE ALL DEVICES TESTED?
751 003504 001017                                     BNE         RESTRT                                   ;BR IF NO.
752 003506 112737 000377 001327                    MOV          #377,QV.FLG                               ;SET THE QUICK VERIFY FLAG.
753 003514 013737 001310 001314                    MOV          DMNUM,SAVNUM                             ;RESTORE THE COUNT
754 003522 013701 000042                             MOV          @#42,R1                                 ;CHECK FOR ACT-11 OR DDP
755 003526 001406                                     BEQ         RESTRT                                   ;IF NOT, CONTINUE TESTING
756 003530 000005                                     RESET                                               ;STOP THE SHOW--CLEAR THE WORLD
757 003532
758 003532 004711                                     SENDAD: JSR          PC,(R1)
759 003534 000240                                     NOP
760 003536 000240                                     NOP
761 003540 000240                                     NOP
762 003542 000240                                     NOP
763 003544 012737 010120 001214 RESTRT: MOV          #CYCLE,RETURN
764 003552 000137 010120                             JMP         CYCLE
765 003556 000001                                     XCSR:      1
766 003560 006 002                                     .BYTE     6,2
767 003562 001404                                     DMCSR
768 003564 000001                                     XVEC:      1
769 003566 004 002                                     .BYTE     4,2
770 003570 001374                                     DMRVEC
771 003572 000001                                     XPASS:     1
772 003574 006 002                                     .BYTE     6,2
773 003576 001230                                     PASCNT
774 003600 000001                                     XERR:      1
775 003602 006 002                                     .BYTE     6,2
776 003604 001232                                     ERRCNT
777
778                                     ;SCOPE LOOP AND INTERATION HANDLER
779 -----
780

```

```

781 003606 004737 007646 .SCOPE: JSR PC,CKSWR ;CHECK FOR SOFT SWR
782 003612 010016 MOV RO,(SP) ;SAVE RO ON THE STACK
783 003614 032777 040000 175360 BIT #BIT14,@SWR ;"LOOP ON THIS TEST"?
784 003622 001407 TTS1: BEQ 1$ ;BR IF NO. (IF LOCK SW01=1; THIS LOC =240)
785 003624 000437 BR 3$ ;GOTO 3$ (IF LOCK SW01=1; THIS LOC =240)
786 003626 005737 003744 TST DONE ;WAS TKCSR DONE SET?
787 003632 001434 BEQ 3$ ;BR IF NO (LOCKED ON TEST)
788 003634 005037 003744 CLR DONE ;YES, CLEAR FLAG
789 003640 000415 BR 2$ ;GO TO NEXT TEST
790 003642 032777 004000 175332 1$: BIT #SW11,@SWR ;DELETE ITERATION? (QUICK PASS)
791 003650 001011 BNE 2$ ;BR IF YES
792 003652 105737 001327 TSTB QV.FLG ;HAVE PASSES BECOMPLETED?
793 003656 001406 BEQ 2$ ;BR IF QUICK PASS.
794 003660 005237 001224 INC LPCNT ;UPDATE ITERATION COUNTER
795 003664 023737 001224 001222 CMP LPCNT,ICOUNT ;ARE ALL ITERATIONS DONE??
796 003672 101414 BLOS 3$ ;BR IF NOT YET
797 003674 105037 001325 2$: CLRB ERRFLG ;PREPARE FOR NEW TEST
798 003700 005037 001224 CLR LPCNT ;START ICOUNTER AT 0
799 003704 005037 001220 CLR LOCK
800 003710 012737 000020 001222 MOV #20,ICOUNT ;RESET ITERATIONS
801 003716 013737 001216 001214 MOV NEXT,RETURN ;GET NEXT TEST
802 003724 011600 3$: MOV (SP),RO ;POP RO OFF OF THE STACK
803 003726 022626 POP2SP ;FAKE AN "RTI"
804 003730 013701 001404 MOV DMCSR,R1 ;R1 CONTAINS BASE MB200-YC ADDRESS
805 003734 000177 175254 JMP @RETURN ;GO DO THE TEST
806 003740 001407 BRW: 1407
807 003742 000437 BRX: 437
808 003744 000000 DONE: 0
  
```

;CHECK FOR FREEZE ON CURRENT DATA

```

813 003746 004737 007646 .SCOPE1: JSR PC,CKSWR ;CHECK FOR SOFT SWR
814 003752 032777 001000 175222 BIT #SW09,@SWR ;IS SW09=1(SET)?
815 003760 001405 BEQ 1$ ;BR IF NOT SET.
816 003762 005737 001220 TST LOCK
817 003766 001402 BEQ 1$
818 003770 013716 001220 MOV LOCK,(SP) ;GOTO THE ADDRESS IN LOCK.
819 003774 000002 1$: RTI ;GO BACK.
  
```

;TELETYPE OUTPUT ROUTINE

```

824 003776 010546 .TYPE: MOV R5,-(SP) ;SAVE R5 ON THE STACK.
825 004000 017605 000002 MOV @2(SP),R5 ;GET ADDRESS OF MESSAGE.
826 004004 062766 000002 000002 ADD #2,2(SP) ;POP OVER ADDRESS.
827 004012 005737 010056 4$: TST SWFLG ;SOFT SWR MESSAGE?
828 004016 001004 BNE 1$ ;IF YES TYPE IT OUT REGARDLESS OF SW12
829 004020 032777 010000 175154 BIT #SW12,@SWR ;INHIBIT ALL PRINT OUT??
830 004026 001012 BNE 3$ ;BR IF NO PRINT OUT WANTED (SW12=1)
831 004030 105715 1$: TSTB (R5) ;IS NUMBER MINUS? (MSB=1(BIT7))
832 004032 100002 BPL 2$ ;BR IF NUMBER IS PLUS
833 004034 104402 005702 TYPE MCRLF ;TYPE A CR/LF!
834 004040 105777 175144 2$: TSTB @TPCSR ;TTY READY?
  
```

```

835 004044 100375          BPL      2$          ;BR IF NO.
836 004046 112577 175140  MOVB     (R5)+, @TPDBR ;PRINT CURRENT CHAR.
837 004052 001357          BNE      4$          ;IF NOT ZERO KEEP PRINTING!
838 004054 012605          MOV      (SP)+, R5    ;END OF OUTPUT. RESTORE R5
839 004056 000002          RTI                    ;GO HOME
840
841 -----
842 004060 010346          .INSTR: MOV      R3, -(SP) ;SAVE R3 ON STACK
843 004062 010446          MOV      R4, -(SP) ;SAVE R4 ON STACK
844 004064 017637 000004 004102  MOV      @4(SP), MSG
845 004072 062756 000002 000004  ADD      #2, 4(SP)
846 004100 104402          .INST1: TYPE
847 004102 000000          .MSG:    0
848 004104 012704 007542          MOV      #INBUF, R4
849 004110 012703 000007          MOV      #7, R3
850 004114 105777 175064          1$:     TSTB     @TKCSR
851 004120 100375          BPL      1$
852 004122 117714 175060          MOVB     @TKDBR, (R4)
853 004126 142714 000200          BICB     #200, (R4)
854 004132 122427 000000          CMPB     (R4)+, #15
855 004136 001417          BEQ      INSTR2
856 004140 105777 175044          2$:     TSTB     @TPCSR
857 004144 100375          BPL      2$
858 004146 017777 175034 175036  MOV      @TKDBR, @TPDBR
859 004154 005303          DEC      R3
860 004156 001356          BNE      1$
861 004160 012604          MOV      (SP)+, R4
862 004162 012603          MOV      (SP)+, R3
863 004164 104402 005676          .INSTE: TYPE     MQM
864 004170 010346          MOV      R3, -(SP)
865 004172 010446          MOV      R4, -(SP)
866 004174 000741          BR       .INST1
867 004176 012604          INSTR2: MOV      (SP)+, R4 ;RESTORE R4
868 004200 012603          MOV      (SP)+, R3 ;RESTORE R3
869 004202 000002          RTI
870
871 ;CONVERT ASCII STRING TO OCTAL
872 -----
873
874 .PARAM: MOV      R5, -(SP)
875 004206 010446          MOV      R4, -(SP)
876 004210 016605 000004          MOV      4(SP), R5
877 004214 012537 004374          MOV      (R5)+, LOLIM
878 004220 012537 004376          MOV      (R5)+, HILIM
879 004224 012537 004400          MOV      (R5)+, DEVADR
880 004230 112537 004402          MOVB     (R5)+, LOBITS
881 004234 112537 004403          MOVB     (R5)+, ADRCNT
882 004240 010566 000004          MOV      R5, 4(SP)
883 004244 005005          PARAM1: CLR      R5
884 004246 012704 007542          MOV      #INBUF, R4
885 004252 122714 000015          CMPB     #15, (R4)
886 004256 001420          BEQ      PARERR
887 004260 121427 000060          1$:     CMPB     (R4), #60
888 004264 002415          BLT      PARERR
  
```

```

889 004266 121427 000067      CMPB      (R4), #67
890 004272 003012      BGT      PARERR
891 004274 142714 000060      BICB      #60, (R4)
892 004300 152405      BISB      (R4)+, R5
893 004302 122714 000015      CMPB      #15, (R4)
894 004306 001406      BEQ      LIMITS
895 004310 006305      ASL      R5
896 004312 006305      ASL      R5
897 004314 006305      ASL      R5
898 004316 000760      BR       1$
899 004320 104404      PARERR: INSTER
900 004322 000750      BR       PARAM1
901
902      ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
903      -----
904
905 004324 020537 004376      LIMITS: CMP      R5, HILIM
906 004330 101373      BHI      PARERR
907 004332 020537 004374      CMP      R5, LOLIM
908 004336 103770      BLO      PARERR
909 004340 133705 004402      BITB     LOBITS, R5
910 004344 001365      BNE      PARERR
911
912      ;STORE NUMBER AT SPECIFIED ADDRESS
913
914 004346 013704 004400      1$:      MOV      DEVADR, R4
915 004352 010524      MOV      R5, (R4)+
916 004354 062705 000002      ADD      #2, R5
917 004360 105337 004403      DECB     ADCNT
918 004364 001372      BNE      1$
919 004366 012604      MOV      (SP)+, R4
920 004370 012605      MOV      (SP)+, R5
921 004372 000002      RTI
922 004374 000000      LOLIM:  0
923 004376 000000      HILIM:  0
924 004400 000000      DEVADR: 0
925 004402 000000      LOBITS: 0
926      ADCNT=LOBITS+1
927
928      ;SAVE PC OF TEST THAT FAILED AND R0-R5
929      -----
930
931 004404 016637 000004 001276 .SAVOS: MOV      4(SP), SAVPC      ;SAVE R7 (PC)
932
933      ;SAVE R0-R5
934
935 004412 010537 001272      SVOS:   MOV      R5, SAVR5      ;SAVE R5
936 004416 010437 001270      MOV      R4, SAVR4      ;SAVE R4
937 004422 010337 001266      MOV      R3, SAVR3      ;SAVE R3
938 004426 010237 001264      MOV      R2, SAVR2      ;SAVE R2
939 004432 010137 001262      MOV      R1, SAVR1      ;SAVE R1
940 004436 010037 001260      MOV      R0, SAVR0      ;SAVE R0
941 004442 000002      RTI                          ;LEAVE.
942

```

```

943                                     ;RESTORE R0-R5
944
945 004444 013700 001260      .RES05: MOV      SAVR0,R0      ;RESTORE R0
946 004450 013701 001262      MOV      SAVR1,R1      ;RESTORE R1
947 004454 013702 001264      MOV      SAVR2,R2      ;RESTORE R2
948 004460 013703 001266      MOV      SAVR3,R3      ;RESTORE R3
949 004464 013704 001270      MOV      SAVR4,R4      ;RESTORE R4
950 004470 013705 001272      MOV      SAVR5,R5      ;RESTORE R5
951 004474 000002      RTI                    ;LEAVE
952
953                                     ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
954 -----
955
956 004476 104402 005702      .CONVR: TYPE      MCRLF
957 004502 010046      .CNVRT: MOV      R0,-(SP)
958 004504 010146      MOV      R1,-(SP)
959 004506 010346      MOV      R3,-(SP)
960 004510 010446      MOV      R4,-(SP)
961 004512 010546      MOV      R5,-(SP)
962 004514 017601 000012      MOV      @12(SP),R1
963 004520 062766 000002 000012      ADD      #2,12(SP)
964 004526 012137 004720      MOV      (R1)+,WRDCNT
965 004532 112137 004722      1$:     MOVB   (R1)+,CHRCNT
966 004536 112137 004723      MOVB   (R1)+,SPACNT
967 004542 013137 004724      MOV      @2(R1)+,BINWRD
968 004546 122737 000003 004722      CMPB   #3,CHRCNT
969 004554 001003      BNE     2$
970 004556 042737 177400 004724      BIC     #177400,BINWRD
971 004564 013704 004724      2$:     MOV      BINWRD,R4
972 004570 113705 004722      MOVB   CHRCNT,R5
973 004574 012700 001416      MOV      #TEMP,R0
974 004600 010403      3$:     MOV      R4,R3
975 004602 042703 177770      BIC     #177770,R3
976 004606 062703 000060      ADD      #060,R3
977 004612 110320      MOVB   R3,(R0)+
978 004614 000241      CLC
979 004616 006004      ROR     R4
980 004620 000241      CLC
981 004622 006004      ROR     R4
982 004624 000241      CLC
983 004626 006004      ROR     R4
984 004630 005305      DEC     R5
985 004632 001362      BNE     3$
986 004634 012703 007604      MOV      #MDATA,R3
987 004640 114023 004722      4$:     MOVB   -(R0),(R3)+
988 004642 105337 004722      DECB   CHRCNT
989 004646 001374      BNE     4$
990 004650 105737 004723      TSTB   SPACNT
991 004654 001405      BEQ     6$
992 004656 112723 000040      5$:     MOVB   #040,(R3)+
993 004662 105337 004723      DECB   SPACNT
994 004666 001373      BNE     5$
995 004670 105013      6$:     CLRB   (R3)
996 004672 104402 007604      TYPE   ,MDATA
    
```

997 004676 005337 004720
 998 004702 001313
 999 004704 012605
 1000 004706 012604
 1001 004710 012603
 1002 004712 012601
 1003 004714 012600
 1004 004716 000002
 1005 004720 000000
 1006 004722 000000
 1007 004723 004723
 1008 004724 000000
 1009
 1010
 1011
 1012
 1013
 1014
 1015
 1016 004726 011646
 1017 004730 162716 000002
 1018 004734 017616 000000
 1019 004740 006316
 1020 004742 042716 177001
 1021 004746 062716 001330
 1022 004752 017616 000000
 1023 004756 000136
 1024
 1025
 1026
 1027
 1028 004760 004737 007646
 1029 004764 032777 010000 174210
 1030 004772 001406
 1031 004774 105777 174210
 1032 005000 100003
 1033 005002 112777 000207 174202
 1034 005010 032777 020000 174164
 1035 005016 001105
 1036 005020 021637 001234
 1037 005024 001404
 1038 005026 011637 001234
 1039 005032 105037 001325
 1040 005036 104406
 1041 005040 011605
 1042 005042 162705 000002
 1043 005046 011504
 1044 005050 006304
 1045 005052 061504
 1046 005054 006304
 1047 005056 042704 177001
 1048 005062 062704 036574
 1049 005066 012437 005202
 1050 005072 012437 005214

DEC WRDCNT
 BNE 1\$
 MOV (SP)+,R5
 MOV (SP)+,R4
 MOV (SP)+,R3
 MOV (SP)+,R1
 MOV (SP)+,R0
 RTI
 WRDCNT: 0
 CHRCNT: 0
 SPACNT=CHRCNT+1
 BINWRD: 0

; TRAP DISPATCH SERVICE
 ; ARGUMENT OF TRAP IS EXTRACTED
 ; AND USED AS OFFSET TO OBTAIN POINTER
 ; TO SELECTED SUBROUTINE

.TRPSR: MOV (SP) -(SP) ; GET PC OF RETURN
 SUB #2, (SP) ; =PC OF TRAP
 TRPOK: MOV @ (SP), (SP) ; GET TRP
 ASL (SP) ; MULTIPLY TRAP ARG BY 2
 BIC #177001, (SP) ; CLEAR UNWANTED BITS
 ADD #.TRPTAB, (SP) ; POINTER TO SUBROUTINE ADDRESS
 MOV @ (SP), (SP) ; SUBROUTINE ADDRESS
 JMP @ (SP)+ ; GO TO SUBROUTINE

; ERROR HANDLER

.HLT: JSR PC, CKSWR ; CHECK FOR SOFT SWR
 BIT #SW12, @SWR ; BELL ON ERROR?
 BEQ XBX ; BR IF NO BELL
 TSTB @TPCSR ; TTY READY.
 BPL XBX ; DON'T WAIT IF TTY NOT READY.
 MOVB #207, @TPDBR ; PUSH A BELL AT THE TTY.
 BIT #SW13, @SWR ; DELETE ERROR PRINT OUT?
 BNE HALTS ; BR IF NO PRINT OUT WANTED.
 CMP (SP), LSTERR ; WAS THIS ERROR FOUND LAST TIME?
 BEQ 1\$; BR IF YES
 MOV (SP), LSTERR ; RECORD BEING HERE
 CLRB ERRFLG ; PREPARE HEADER
 1\$: SAVOS ; SAVE ALL PROC REGISTERS
 MOV (SP), R5 ; GET THE PC OF ERROR
 SUB #2, R5 ; GET ADDRESS OF TRAP CALL
 MOV (R5), R4 ; GET HLT INSTRUCTION
 ASL R4 ; MULT BY TWO
 ADD (R5), R4 ; DOUBLE IT
 ASL R4 ; MULT AGAIN
 BIC #177001, R4 ; CLEAR JUNK
 ADD #.ERRTAB, R4 ; GET POINTER
 MOV (R4)+, ERRMSG ; GET ERROR MESSAGE
 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	MOVW	#-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:	RESOS		:RESTORE PROC REGISTERS
1081	005232	022737	003532	000042	HALTS:	CMP	#SENDAD, @#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			POPPO			: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			1			
1103	005342	003	002		.BYTE	3, 2		
1104	005344	001226			TSTNO			

```

1105                                     ;ENTER HERE ON POWER FAILURE
1106                                     ;-----
1107
1108
1109 005346 .PFAIL:
1110 005346 012737 005360 000024 MOV #PESTART,24 ;SET UP FOR POWER UP TRAP
1111 005354 000000 HALT ;HALT ON POWER DOWN NORMAL
1112 005356 000777 BR .
1113
1114                                     ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
1115
1116 005360 RESTAR:
1117 005360 012737 005346 000024 MOV #.PFAIL,24 ;SET UP FOR POWER FAILURE
1118 005366 012706 001200 MOV #STACK,SP ;RESET THE STACK POINTER
1119 005372 013701 001404 MOV DMCSR,R1 ;RESTORE R1
1120 005376 005037 001416 CLR TEMP ;READY FOR TIMER
1121 005402 005237 001416 INC TEMP ;PLUS ONE TO THE TIMER!
1122 005406 001375 BNE -4 ;BR IF MORE TO GO
1123 005410 104402 005705 TYPE ,MPFAIL ;TYPE THE MESSAGE
1124 005414 104411 005440 CNVRT ,PFTAB ;TELL WHAT TEST TO RETURN TO.
1125 005420 105037 001325 CLRB ERRFLG ;START CLEAN
1126 005424 005037 001234 CLR LSTERR
1127 005430 005011 CLR (R1) ;CLEAR MAINT BITS
1128 005432 104412 MSTCLR ;START CLEAN UP OF DEVICE
1129 005434 000177 173554 JMP @RETURN ;START DOING THAT TEST AGAIN.
1130 005440 000001 PFTAB: 1
1131 005442 003 002 .BYTE 3,2
1132 005444 001226 TSTNO
1133
1134 005446 .DELAY:
1135 005446 012777 000020 173736 MOV #20,@DMP04
1136 005454 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1137 005456 121111 121111 ;POKE CLOCK DELAY BIT
1138 005460 1S:
1139 005460 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1140 005462 121224 121224 ;PORT4+IBUS*11
1141 005464 032777 000020 173720 BIT #BIT4,@DMP04 ;IS CLOCK BIT SET?
1142 005472 001772 BEQ 1S ;BR IF NO
1143 005474 000002 RTI
1144
1145 005476 .MSTCLR:
1146 005476 152777 000100 173702 BISB #BIT6,@DMCSRH ;SET MASTER CLEAR
1147 005504 142777 000300 173674 BICB #BIT6!BIT7,@DMCSRH ;CLEAR MASTER CLEAR AND RUN
1148 005512 000002 RTI ;RETURN
1149
1150 005514 .ROMCLK:
1151 005514 152777 000002 173664 BISB #BIT1,@DMCSRH ;SET ROMI
1152 005522 013677 173666 MOV @SP+,@DMP06 ;LOAD INSTRUCTION IN SEL6
1153 005526 062746 000002 ADD #2,-(SP) ;ADJUST STACK
1154 005532 032777 000100 173442 BIT #SW06,@SWR ;HALT IF SW06 =1
1155 005540 001401 BEQ 1S ;BR IF SW06 =0
1156 005542 000000 HALT ;HALT BEFORE CLOCKING INSTRUCTION
1157 005544 152777 000003 173634 1S: BISB #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
1158 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 1S:  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      1S              ; BR IF NOT DONE
1169 005622 000002          RTI                    ; RETURN
1170 005624 000001          3S:  .BLKW 1
1171
1172          .TIMER:
1173 005626 013637 001416      MOV      @ (SP)+, TEMP    ; MOVE COUNT TO TEMP
1174 005632 062746 000002      ADD      #2, -(SP)        ; ADJUST STACK
1175 005636
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      1S              ; BR IF YES
1180 005652
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      2S              ; BR IF YES
1185 005666 005337 001416      DEC      TEMP             ; DEC COUNT
1186 005672 001361          BNE      1S              ; 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 DRLPL /
(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>/
(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 TESTED?/
(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 (DCMP 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 TEMPS
1200 .EVEN
1201 ;BUFFERS FOR INPUT-OUTPUT
1202
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 CKSWRS ;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 CKSWRS ;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

```

DRLPL MACY11 27(654) 13-DEC-77 11:41 PAGE 27
 DRLPL.P11 GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

SEQ 0043

```

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,JSWR    ;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      .BYTE   6,2
1272 010116 000176      SWREG

```

```

1273
1274
1275
1276
1277
1278
1279
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)+,DMC#R ;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,DMRLVL ;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
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 012372      MOV      #TST1,R0
1340 010450 022710          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$: TST      (R0)+      ;POP R0
1352 010520 020027 034006      CMP      R0,#TLAST+10      ;AT END YET?
1353 010524 001351          BNE      5$      ;BR IF NO
1354 010526 104402 005676          TYPE     MQM      ;YES ILLEGAL TEST NO.
1355 010532 000730          BR       4$      ;TRY AGAIN
1356
1357 010534 012737 012372 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
1363
1364
1365
1366
1367
1368
1369
1370
1371 010552          AUTO.SIZE:
1372 010554 000005          RESET
1373 010560 012702 001500      CSRMAP: MOV      #DM.MAP,R2      ;INSURE A BUS INIT.
1374 010562 005022          1$: CLR      (R2)+      ;LOAD MAP POINTER.
1375 010566 022702 001700      CMP      #DM.#ND,R2      ;ZERO ENTIRE MAP
1376 010570 001374          BNE      1$      ;ALL DONE?
1377 010574 005037 001310      CLR      DMNUM      ;BR IF NO
1378 010600 012702 001500      MOV      #DM.MAP,R2      ;SET OCTAL NUMBER OF M8200-YC'S TO 0
1379 010604 005037 001306      CLR      DMACTV      ;R2 POINTS TO M8200-YC MAP
1380 010612 032737 000001 001236      BIT      #SW00,STRTSW      ;CLEAR ACTIVE
1380 010612 001002          BNE      .+6      ;QUESTIONS?
1380 010612 001002          BNE      .+6      ;BR IF YES
  
```

```

;ROUTINE USED TO "AUTO SIZE" THE M8200-YC
;CSR AND VECTOR.
;NOTE: THE CSR MAY BE ANY WHERE IN THE
;ADDRESS RANGE (170440:170510)
;AND THE VECTOR MAY BE ANY WHERE IN THE
;FLOATING VECTOR RANGE (300:770)
  
```

```

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 0
1391 010644 013737 001252 001310      MOV      TEMP3,DMNUM ;DMNUM = HOW MANY
1392 010652 104402 005702          TYPE ,MCRLF
1393 010656 104410          CONVRT ;TYPE WHICH DMC IS BEING DONE
1394 010660 012054          WHICH ;TEMPS IS WHICH DMC
1395 010662 005237 001256      INC      TEMPS
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          TYPE
1415 010734 006570          PRIO
1416 010736 004737 012340      JSR      PC,INTTY ;ASK WHAT BR LEVEL
1417 010742 022703 000024      CMP      #24,R3 ;GET RESPONSE
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          TYPE
1429 011002 005676          MQM
1430 011004 000752          BR      10$ ;RESPONSE IS OUT OF LIMITS
1431 011006          BR      8$ ;TRY AGAIN
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	011235	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$: MOV #170440,R1 ;SET FOR FIRST ADDRESS TO BE TESTED
1512 011326 012737 012046 000004 MOV #6$ ,2#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 DMR1C S/B =0!
1519 011354 001165 BNE 3$ ;BR IF NOT M8200-YC
1520 011356 012711 002000 MOV #BIT10,(R1) ;SET ROMO
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 CROM
1526 011410 052762 100000 000002 BIS #BIT15,2(R2) ;SET BIT15 IF CROM
1527 011416 000431 BR 22$
1528 011420 012711 001000 21$: MOV #BIT9,(R1) ;SET ROMI
1529 011424 012761 100400 000006 MOV #100400,6(R1) ;PUT INSTRUCTION IN SEL6
1530 011432 012711 001400 MOV #BIT9!BIT8,(R1) ;CLOCK INSTRUCTION (MICRO PROC PC TO 0)
1531 011436 012711 002000 MOV #BIT10,(R1) ;SET ROMO
1532 011442 022761 000456 000006 CMP #456,6(R1) ;IS IT LOCAL CROM
1533 011450 001411 BEQ 23$ ;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			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 IS
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			CLR	(R1)	;CLEAR BIT 10
1579	011732	005061	000006		CLR	6(R1)	;CLEAR SEL 6
1580	011736	062701	000010		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		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		MC	DMNUM,R1	
1588	011772	010137	001314		MC	R1,SAVNUM	;SAVE NUMBER OF DEVICES
1589	011776	000241			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	#6,2#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$:  TYPE      MERR2      ;NOTIFY OPR THAT NO M8200-YC'S FOUND.
1598 012040 005000              CLR      RO          ;MAKE DATA LIGHTS ZERO
1599 012042 000000              HALT                    ;STOP THE SHOW
1600 012044 000776              BR      -2           ;DISABLE CONT. SW.
1601 012046 012716 011736      6$:  MOV      #14$, (SP)  ;ENTERED BY NON-EXISTANT TIME-OUT.
1602 012052 000002              RTI                    ;RETURN TO MAINSTREAM
1603
1604 012054 000001              WHICH:  1
1605 012056      002      002      .BYTE      2,2
1606 012060 001256              TEMPS
1607
1608 012062 032737 000001 001236  VECMAP: BIT      #SW00, STRTSW
1609 012070 001114              BNE      5$
1610 012072 012737 000340 000022  MOV      #340, 2#22  ;SET IOT TRAP PRIO TO 7
1611 012100 012737 012254 000020  MOV      #4$, 2#20  ;SET IOT TRAP VECTOR
1612 012106 012702 001500  MOV      #DM.MAP, R2 ;SET SOFTWARE POINTER
1613 012112 012700 000300  MOV      #300, RO    ;FLOATING VECTORS START HERE.
1614 012116 012701 000302  MOV      #302, R1    ;PC OF IOT INSTR.
1615 012122 010120              1$:  MOV      R1, (RO)+  ;START FILLING VECTOR AREA
1616 012124 012721 000004  MOV      #4, (R1)+  ;WITH .+2; IOT
1617 012130 022021              CMP      (RO)+, (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)
1628 012202 012761 121111 000006  MOV      #121111, 6(R1) ;SET ROMI
1629 012210 012711 001400  MOV      #BIT9:BIT8, (R1) ;PUT INSTRUCTION IN PORT6
1630 012214 105200              7$:  INCB     RO          ;FORCE AN INTERRUPT
1631 012216 001376              BNE      -2         ;STALL
1632 012220 162704 000002  SUB      #2, R4     ;FOR TIME TO INTERRUPT
1633 012224 001404              BEQ      6$          ;GET NEXT LOWEST PS LEVEL
1634 012226 016437 012324 177776  MOV      BRLVL(R4), PS ;BR IF R4 = 0
1635 012234 000767              BR      7$          ;MOVE NEXT LOWER LEVEL IN PS
1636 012236 052762 005300 000002  6$:  BIS      #5300, 2(R2) ;BR TO DELAY
1637 012244 005011              3$:  CLR      (R1)      ;NO INTERRUPT ASSUME 300 AT LEVEL 5 AND FIX M8200-YC LATE
1638 012246 062702 000010  ADD      #10, R2    ;CLEAR ROMI
1639 012252 000735              BR      2$          ;POP SOFTWARE POINTER
1640 012254 051662 000002  4$:  BIS      (SP), 2(R2) ;KEEP GOING
1641 012260 042762 000007 000002  BIC      #7, 2(R2)  ;GET VECTOR ADDRESS
1642 012266 016405 012326  MOV      BRLVL+2(R4), R5 ;CLEAR JUNK
1643 012272 006305              ASL      R5          ;GET BR LEVEL OF M8200-YC
1644 012274 006305              ASL      R5          ;SHIFT LEVEL 4 PLACES
1645 012276 006305              ASL      R5          ;TO THE LEFT FOR THE
1646 012300 006305              ASL      R5          ;STATUS TABLE
1647 012302 042705 170777  BIC      #170777, R5 ;CLEAR UNWANTED BITS
1648 012306 050562 000002  BIS      R5, 2(R2)  ;PUT BR LEVEL IN STATUS TABLE
1649 012312 022626              CMP      (SP)+, (SP)+ ;POP IOT JUNK OFF STACK
1650 012314 012716 012244  MOV      #3$, (SP)  ;SET FOR RETURN
  
```

```

1651 012320 000002
1652 012322 000207
1653
1654 012324 000000
1655 012326 000000
1656 012330 000200
1657 012332 000240
1658 012334 000300
1659 012336 000340
1660

```

SS: RTI PC ;ALL DONE WITH "AUTO SIZING"

BRLVL: 0 ;LEVEL 0
0 ;LEVEL 0
200 ;LEVEL 4
240 ;LEVEL 5
300 ;LEVEL 6
340 ;LEVEL 7

```

1661
1662 012340 105777 166640
1663 012344 100375
1664 012346 017703 166634
1665 012352 105777 166632
1666 012356 100375
1667 012360 010377 166626
1668 012364 042703 000240
1669 012370 000207
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679

```

INTTY: TSTB @TKCSR ;WAIT FOR DONE
BPL -4
MOV @TKDBR,R3 ;PUT CHAR IN R3
TSTB @TPCSR ;WAIT UNTIL PRINTER IS READY
BPL -4
MOV R3,@TPDBR ;ECHO CHAR
BIC #BIT7:BITS,R3 ;MASK OFF LOWER CASE
RTS PC ;RETURN

***** TEST 1 *****
:VERIFY THAT REFERENCING UNIBUS DEVICE REGISTERS
:DOES NOT CAUSE A TIME OUT TRAP

```

1680 012372 012737 000001 001226 TST1: MOV #1,TSTNO
1681 012400 012737 012500 001216 MOV #TST2,NEXT
1682 012406 012737 012440 001220 MOV #1$,LOCK
1683
1684 012414 013701 001404 MOV DMC,R,R1 ;R1 CONTAINS BASE M8200-YC ADDRESS
1685 012420 012700 000004 MOV #4,R0 ;R1 CONTAINS BASE M8200-YC ADDRESS
1686 012424 012737 012472 000004 MOV #2$,4 ;4 REGISTERS TO BE TESTED
1687 012432 012737 000340 000006 MOV #340,6 ;SET UP TIMEOUT TRAP
1688 012440 005711 1$: TST (R1) ;LEVEL 7
1689 012442 000240 ;REFERENCE DEVICE REGISTER
1690 012444 104401 SCOP1 ;SWD9=1?
1691 012446 062701 000002 ADD #2,R1 ;NEXT REGISTER
1692 012452 005300 DEC R0 ;DEC REGISTER COUNT
1693 012454 001371 BNE 1$ ;BR IF NOT LAST REGISTER
1694 012456 012737 000006 000004 MOV #6,4 ;RESTORE LOC 4
1695 012464 005037 000006 CLR 6 ;RESTORE LOC 6
1696 012470 104400 SCOPE ;SCOPE THIS TEST
1697 012472 011602 2$: MOV (SP),R2 ;GET PC OF TRAP
1698 012474 104001 HLT 1 ;TIME-OUT ERROR
1699 012476 000002 RTI
1700
1701
1702
1703
1704

```

***** TEST 2 *****
:VERIFY THAT RUN CAN BE CLEARED

```

1705
1706
1707 ; TEST 2
1708 012500 012737 000002 001226 TST2: MOV #2,TSTNO
1709 012506 012737 012530 001216 MOV #TST3,NEXT
1710 ;R1 CONTAINS BASE M8200-YC ADDRESS
1711 012514 005011 CLR (R1) ;CLEAR DMCSR
1712 012516 005005 CLR R5 ;CLEAR "EXPECTED"
1713 012520 011104 MOV (R1),R4 ;PUT DMCSR IN "FOUND"
1714 012522 001401 BEQ 1$ ;BR IF CLEARED
1715 012524 104002 HLT 2 ;ERROR DMCSR NOT CLEARED
1716 012526 104400 1$: SCOPE ;SCOPE THIS TEST
1717
1718
1719

```

```

***** TEST 3 *****
*UNIBUS REGISTER WORD DUAL ADDRESSING TEST
*LOAD ALL REGISTERS WITH INCREMENTING PATTERN
*READ BACK ALL REGISTERS TO VERIFY CORRECT ADDRESSING
*****

```

```

1720 ; TEST 3
1721
1722
1723
1724
1725
1726
1727 012530 012737 000003 001226 TST3: MOV #3,TSTNO
1728 012536 012737 012660 001216 MOV #TST4,NEXT
1729 012544 012737 012560 001220 MOV #1$,LOCK
1730 ;R1 CONTAINS BASE M8200-YC ADDRESS
1731 012552 104412 MSTCLR ;MASTER CLEAR M8200-YC
1732 012554 012700 000001 MOV #1,R0 ;START PATTERN AT 1
1733 012560 005011 1$: CLR (R1) ;CLEAR REGISTER
1734 012562 010005 MOV R0,R5 ;PUT DATA IN "EXPECTED"
1735 012564 010011 MOV R0,(R1) ;WRITE DMC REGISTER WITH PATTERN
1736 012566 011104 MOV (R1),R4 ;READ DMC REGISTER INTO "FOUND"
1737 012570 020504 CMP R5,R4 ;IS DATA CORRECT
1738 012572 001401 BEQ 2$ ;BR IF YES
1739 012574 104002 HLT 2 ;DATA ERROR
1740 012576 104401 2$: SCOP1 ;SW09=1?
1741 012600 005721 TST (R1)+ ;NEXT REGISTER
1742 012602 005200 INC R0 ;INCREMENT DATA PATTERN
1743 012604 022700 000005 CMP #5,R0 ;LAST REGISTER?
1744 012610 001363 BNE 1$ ;BR IF NO
1745 012612 013701 001404 MOV DMCSR,R1 ;BASE M8200-YC ADDRESS TO R1
1746 012616 012700 000001 MOV #1,R0 ;RESTART PATTERN AT 1
1747 012622 012737 012630 001220 3$: MOV #3$,LOCK ;NEW SCOP1
1748 012630 010005 MOV R0,R5 ;PUT DATA IN "EXPECTED"
1749 012632 011104 MOV (R1),R4 ;READ DMC REGISTER INTO "FOUND"
1750 012634 020504 CMP R5,R4 ;IS DATA CORRECT
1751 012636 001401 BEQ 4$ ;BR IF YES
1752 012640 104002 HLT 2 ;DUAL ADDRESSING ERROR
1753 012642 104401 4$: SCOP1 ;SW09=1?
1754 012644 005721 TST (R1)+ ;NEXT REGISTER
1755 012646 005200 INC R0 ;INCREMENT PATTERN
1756 012650 022700 000005 CMP #5,R0 ;LAST REGISTER?
1757 012654 001363 BNE 3$ ;BR IF NO
1758 012656 104400 SCOPE ;SCOPE THIS TEST

```

1759
 1760
 1761
 1762
 1763
 1764
 1765
 1766
 1767
 1768
 1769
 1770
 1771
 1772
 1773
 1774
 1775
 1776
 1777
 1778
 1779
 1780
 1781
 1782
 1783
 1784
 1785
 1786
 1787
 1788
 1789
 1790
 1791
 1792
 1793
 1794
 1795
 1796
 1797
 1798

012660	012737	000004	001226	TST4:
012666	012737	012756	001216	
012674	012737	012704	001220	
012702	104412			
012704	013701	001404		1\$:
012710	012705	000001		
012714	010511			
012716	011104			
012720	020504			
012722	001401			
012724	104002			
012726	104401			2\$:
012730	012737	012736	001220	
012736	042711	000001		3\$:
012742	005005			
012744	011104			
012746	001402			
012750	104002			
012752	104401			
012754	104400			4\$:

```

***** TEST 4 *****
*CONTROL STATUS REGISTER WRITE/READ TEST
*SET BIT0, VERIFY BIT0 WAS SET
*CLEAR BIT0, VERIFY BIT0 WAS CLEARED
*****
    
```

TEST 4

```

-----
MOV #4,TSTNO
MOV #TST5,NEXT
MOV #1$,LOCK
MSTCLR
MOV DMCSR,R1 ;MASTER CLEAR M8200-YC
MOV #BIT0,R5 ;PUT REGISTER ADDRESS IN R1
MOV R5,(R1) ;PUT DATA IN "EXPECTED"
MOV (R1),R4 ;WRITE BIT 0
CMP R5,R4 ;READ CONTROL STATUS REGISTER
BEQ 2$ ;IS DATA CORRECT
HLT 2 ;BR IF YES
SCOPI ;DATA ERROR
MOV #3$,LOCK ;SW09 UP?
BIC #BIT0,(R1) ;NEW SCOPI
CLR R5 ;CLEAR BIT 0
MOV (R1),R4 ;CLEAR "EXPECTED"
BEQ 4$ ;READ CONTROL STATUS REGISTER
HLT 2 ;BR IF ZERO
SCOPI ;DATA ERROR BIT0 NOT CLEARED
SCOPE ;SW09 UP?
SCOPE ;SCOPE THIS TEST
    
```

```

***** TEST 5 *****
*CONTROL STATUS REGISTER WRITE/READ TEST
*SET BIT1, VERIFY BIT1 WAS SET
*CLEAR BIT1, VERIFY BIT1 WAS CLEARED
*****
    
```

TEST 5

012756	012737	000005	001226	TST5:
012764	012737	013054	001216	
012772	012737	013002	001220	
013000	104412			
013002	013701	001404		1\$:
013006	012705	000002		
013012	010511			
013014	011104			
013016	020504			
013020	001401			
013022	104002			
013024	104401			2\$:
013026	012737	013034	001220	
013034	042711	000002		3\$:

```

-----
MOV #5,TSTNO
MOV #TST6,NEXT
MOV #1$,LOCK
MSTCLR
MOV DMCSR,R1 ;MASTER CLEAR M8200-YC
MOV #BIT1,R5 ;PUT REGISTER ADDRESS IN R1
MOV R5,(R1) ;PUT DATA IN "EXPECTED"
MOV (R1),R4 ;WRITE BIT 1
CMP R5,R4 ;READ CONTROL STATUS REGISTER
BEQ 2$ ;IS DATA CORRECT
HLT 2 ;BR IF YES
SCOPI ;DATA ERROR
MOV #3$,LOCK ;SW09 UP?
BIC #BIT1,(R1) ;NEW SCOPI
CLR R5 ;CLEAR BIT 1
    
```

```

1813 013040 005005 CLR R5 ;CLEAR "EXPECTED"
1814 013042 011104 MOV (R1),R4 ;READ CONTROL STATUS REGISTER
1815 013044 001402 BEQ 4$ ;BR IF ZERO
1816 013046 104002 HLT 2 ;DATA ERROR BIT1 NOT CLEARED
1817 013050 104401 SCOPE1 ;SW09 UP?
1818 013052 104400 4$: SCOPE ;SCOPE THIS TEST
    
```

```

1821 ***** TEST 6 *****
1822 *CONTROL STATUS REGISTER WRITE/READ TEST
1823 *SET BIT2, VERIFY BIT2 WAS SET
1824 *CLEAR BIT2, VERIFY BIT2 WAS CLEARED
1825 *****
    
```

```

1827 ; TEST 6
1828 -----
1829 013054 012737 000006 001226 TST6: MOV #6,TSTNO
1830 013062 012737 013152 001216 MOV #TST7,NEXT
1831 013070 012737 013100 001220 MOV #1$,LOCK
1832 013076 104412 MSTCLR ;MASTER CLEAR M8200-YC
1833 013100 013701 001404 1$: MOV DMCSR,R1 ;PUT REGISTER ADDRESS IN R1
1834 013104 012705 000004 MOV #BIT2,R5 ;PUT DATA IN "EXPECTED"
1835 013110 010511 MOV R5,(R1) ;WRITE BIT 2
1836 013112 011104 MOV (R1),R4 ;READ CONTROL STATUS REGISTER
1837 013114 020504 CMP R5,R4 ;IS DATA CORRECT
1838 013116 001401 BEQ 2$ ;BR IF YES
1839 013120 104002 HLT 2 ;DATA ERROR
1840 013122 104401 2$: SCOPE1 ;SW09 UP?
1841 013124 012737 013132 001220 3$: MOV #3$,LOCK ;NEW SCOPE1
1842 013132 042711 000004 BIC #BIT2,(R1) ;CLEAR BIT 2
1843 013136 005005 CLR R5 ;CLEAR "EXPECTED"
1844 013140 011104 MOV (R1),R4 ;READ CONTROL STATUS REGISTER
1845 013142 001402 BEQ 4$ ;BR IF ZERO
1846 013144 104002 HLT 2 ;DATA ERROR BIT2 NOT CLEARED
1847 013146 104401 SCOPE1 ;SW09 UP?
1848 013150 104400 4$: SCOPE ;SCOPE THIS TEST
    
```

```

1851 ***** TEST 7 *****
1852 *CONTROL STATUS REGISTER WRITE/READ TEST
1853 *SET BITS, VERIFY BITS WAS SET
1854 *CLEAR BITS, VERIFY BITS WAS CLEARED
1855 *****
    
```

```

1857 ; TEST 7
1858 -----
1859 013152 012737 000007 001226 TST7: MOV #7,TSTNO
1860 013160 012737 013250 001216 MOV #TST10,NEXT
1861 013166 012737 013176 001220 MOV #1$,LOCK
1862 013174 104412 MSTCLR ;MASTER CLEAR M8200-YC
1863 013176 013701 001404 1$: MOV DMCSR,R1 ;PUT REGISTER ADDRESS IN R1
1864 013202 012705 000040 MOV #BIT5,R5 ;PUT DATA IN "EXPECTED"
1865 013206 010511 MOV R5,(R1) ;WRITE BIT 5
1866 013210 011104 MOV (R1),R4 ;READ CONTROL STATUS REGISTER
    
```

```

1867 013212 020504          CMP      R5,R4          ; IS DATA CORRECT
1868 013214 001401          BEQ      2$             ; BR IF YES
1869 013216 104002          HLT      2             ; DATA ERROR
1870 013220 104401          SCOPE1 2             ; SW09 UP?
1871 013222 012737 013230 001220 2$:  MOV      #3$,LOCK      ; NEW SCOPE1
1872 013230 042711 000040 3$:  BIC      #BIT5,(R1)    ; CLEAR BIT 5
1873 013234 005005          CLR      R5            ; CLEAR "EXPECTED"
1874 013236 011104          MOV      (R1),R4       ; READ CONTROL STATUS REGISTER
1875 013240 001402          BEQ      4$             ; BR IF ZERO
1876 013242 104002          HLT      2             ; DATA ERROR BITS NOT CLEARED
1877 013244 104401          SCOPE1 4$             ; SW09 UP?
1878 013246 104400          SCOPE 4$              ; SCOPE THIS TEST

```

```

1881 ;***** TEST 10 *****
1882 ;*CONTROL STATUS REGISTER WRITE/READ TEST
1883 ;*SET BIT6, VERIFY BIT6 WAS SET
1884 ;*CLEAR BIT6, VERIFY BIT6 WAS CLEARED
1885 ;*****

```

```

1887 ; TEST 10
1888 -----
1889 013250 012737 000010 001226 TST10: MOV      #10,TSTNO
1890 013256 012737 013346 001216      MOV      #TST11,NEXT
1891 013264 012737 013274 001220      MOV      #1$,LOCK
1892 013272 104412          MSTCLR                    ; MASTER CLEAR M8200-YC
1893 013274 013701 001404          MOV      DMCSR,R1        ; PUT REGISTER ADDRESS IN R1
1894 013300 012705 000100 1$:  MOV      #BIT6,RE        ; PUT DATA IN "EXPECTED"
1895 013304 010511          MOV      R5,(R1)        ; WRITE BIT 6
1896 013306 011104          MOV      (R1),R4        ; READ CONTROL STATUS REGISTER
1897 013310 020504          CMP      R5,R4          ; IS DATA CORRECT
1898 013312 001401          BEQ      2$             ; BR IF YES
1899 013314 104002          HLT      2             ; DATA ERROR
1900 013316 104401          SCOPE1 2             ; SW09 UP?
1901 013320 012737 013326 001220 2$:  MOV      #3$,LOCK      ; NEW SCOPE1
1902 013326 042711 000100 3$:  BIC      #BIT6,(R1)    ; CLEAR BIT 6
1903 013332 005005          CLR      R5            ; CLEAR "EXPECTED"
1904 013334 011104          MOV      (R1),R4       ; READ CONTROL STATUS REGISTER
1905 013336 001402          BEQ      4$             ; BR IF ZERO
1906 013340 104002          HLT      2             ; DATA ERROR BIT6 NOT CLEARED
1907 013342 104401          SCOPE1 4$             ; SW09 UP?
1908 013344 104400          SCOPE 4$              ; SCOPE THIS TEST

```

```

1911 ;***** TEST 11 *****
1912 ;*CONTROL STATUS REGISTER WRITE/READ TEST
1913 ;*SET BIT7, VERIFY BIT7 WAS SET
1914 ;*CLEAR BIT7, VERIFY BIT7 WAS CLEARED
1915 ;*****

```

```

1917 ; TEST 11
1918 -----
1919 013346 012737 000011 001226 TST11: MOV      #11,TSTNO
1920 013354 012737 013444 001216      MOV      #TST12,NEXT

```

```

1921 013362 012737 013372 001220      MOV      #1$,LOCK
1922 013370 104412      MSTCLR
1923 013372 013701 001404      1$: MOV      DMCSR,R1      ;MASTER CLEAR M8200-YC
1924 013376 012705 000200      MOV      #BIT7,R5      ;PUT REGISTER ADDRESS IN R1
1925 013402 010511      MOV      R5,(R1)      ;PUT DATA IN "EXPECTED"
1926 013404 011104      MOV      (R1),R4      ;WRITE BIT 7
1927 013406 020504      CMP      R5,R4      ;READ CONTROL STATUS REGISTER
1928 013410 001401      BEQ      2$,          ;IS DATA CORRECT
1929 013412 104002      HLT      2           ;BR IF YES
1930 013414 104401      2$: SCOPE1          ;DATA ERROR
1931 013416 012737 013424 001220      MOV      #3$,LOCK      ;SW09 UP?
1932 013424 042711 000200      3$: BIC      #BIT7,(R1)    ;NEW SCOPE1
1933 013430 005005      CLR      R5           ;CLEAR BIT 7
1934 013432 011104      MOV      (R1),R4      ;CLEAR "EXPECTED"
1935 013434 001402      BEQ      4$,          ;READ CONTROL STATUS REGISTER
1936 013436 104002      HLT      2           ;BR IF ZERO
1937 013440 104401      SCOPE1          ;DATA ERROR BIT7 NOT CLEARED
1938 013442 104400      4$: SCOPE          ;SW09 UP?
                               ;SCOPE THIS TEST

```

```

1939
1940
1941      ;***** TEST 12 *****
1942      ;*CONTROL STATUS REGISTER WRITE/READ TEST
1943      ;*SET BIT9, VERIFY BIT9 WAS SET
1944      ;*CLEAR BIT9, VERIFY BIT9 WAS CLEARED
1945      ;*****

```

```

1946
1947      ; TEST 12
1948      ;-----
1949 013444 012737 000012 001226 TST12: MOV      #12,TSTNO
1950 013452 012737 013542 001216 MOV      #TST13,NEXT
1951 013460 012737 013470 001220 MOV      #1$,LOCK
1952 013466 104412      MSTCLR
1953 013470 013701 001404      1$: MOV      DMCSR,R1      ;MASTER CLEAR M8200-YC
1954 013474 012705 001000      MOV      #BIT9,R5      ;PUT REGISTER ADDRESS IN R1
1955 013500 010511      MOV      R5,(R1)      ;PUT DATA IN "EXPECTED"
1956 013502 011104      MOV      (R1),R4      ;WRITE BIT 9
1957 013504 020504      CMP      R5,R4      ;READ CONTROL STATUS REGISTER
1958 013506 001401      BEQ      2$,          ;IS DATA CORRECT
1959 013510 104002      HLT      2           ;BR IF YES
1960 013512 104401      2$: SCOPE1          ;DATA ERROR
1961 013514 012737 013522 001220      MOV      #3$,LOCK      ;SW09 UP?
1962 013522 042711 001000      3$: BIC      #BIT9,(R1)    ;NEW SCOPE1
1963 013526 005005      CLR      R5           ;CLEAR BIT 9
1964 013530 011104      MOV      (R1),R4      ;CLEAR "EXPECTED"
1965 013532 001402      BEQ      4$,          ;READ CONTROL STATUS REGISTER
1966 013534 104002      HLT      2           ;BR IF ZERO
1967 013536 104401      SCOPE1          ;DATA ERROR BIT9 NOT CLEARED
1968 013540 104400      4$: SCOPE          ;SW09 UP?
                               ;SCOPE THIS TEST

```

```

1969
1970
1971      ;***** TEST 13 *****
1972      ;*CONTROL STATUS REGISTER WRITE/READ TEST
1973      ;*SET BIT11, VERIFY BIT11 WAS SET
1974      ;*CLEAR BIT11, VERIFY BIT11 WAS CLEARED

```

```

1975
1976
1977
1978
1979 013542 012737 000013 001226 TST13:
1980 013550 012737 013640 001216
1981 013556 012737 013566 001220
1982 013564 104412
1983 013566 013701 001404 1$:
1984 013572 012705 004000
1985 013576 010511
1986 013600 011104
1987 013602 020504
1988 013604 001401
1989 013606 104002
1990 013610 104401 2$:
1991 013612 012737 013620 001220
1992 013620 042711 3$:
1993 013624 005005
1994 013626 011104
1995 013630 001402
1996 013632 104002
1997 013634 104401
1998 013636 104400 4$:
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009 013640 012737 000014 001226 TST14:
2010 013646 012737 013736 001216
2011 013654 012737 013664 001220
2012 013662 104412
2013 013664 013701 001404 1$:
2014 013670 012705 010000
2015 013674 010511
2016 013676 011104
2017 013700 020504
2018 013702 001401
2019 013704 104002
2020 013706 104401 2$:
2021 013710 012737 013716 001220
2022 013716 042711 3$:
2023 013722 005005
2024 013724 011104
2025 013726 001402
2026 013730 104002
2027 013732 104401
2028 013734 104400 4$:

```

TEST 13

```

-----
MOV #13,TSTNO
MOV #TST14,NEXT
MOV #1$,LOCK
MSTCLR
MOV DMCsR,R1 ;MASTER CLEAR M8200-YC
PUT REGISTER ADDRESS IN R1
MOV #BIT11,R5 ;PUT DATA IN "EXPECTED"
MOV R5,(R1) ;WRITE BIT 11
MOV (R1),R4 ;READ CONTROL STATUS REGISTER
CMP R5,R4 ;IS DATA CORRECT
BEQ 2$ ;BR IF YES
HLT 2 ;DATA ERROR
SCOPI ;SW09 UP?
MOV #3$,LOCK ;NEW SCOPI
BIC #BIT11,(R1) ;CLEAR BIT 11
CLR R5 ;CLEAR "EXPECTED"
MOV (R1),R4 ;READ CONTROL STATUS REGISTER
BEQ 4$ ;BR IF ZERO
HLT 2 ;DATA ERROR BIT11 NOT CLEARED
SCOPI ;SW09 UP?
SCOPE ;SCOPE THIS TEST

```

```

***** TEST 14 *****
*CONTROL STATUS REGISTER WRITE/READ TEST
*SET BIT12, VERIFY BIT12 WAS SET
*CLEAR BIT12, VERIFY BIT12 WAS CLEARED
*****

```

TEST 14

```

-----
MOV #14,TSTNO
MOV #TST15,NEXT
MOV #1$,LOCK
MSTCLR
MOV DMCsR,R1 ;MASTER CLEAR M8200-YC
PUT REGISTER ADDRESS IN R1
MOV #BIT12,R5 ;PUT DATA IN "EXPECTED"
MOV R5,(R1) ;WRITE BIT 12
MOV (R1),R4 ;READ CONTROL STATUS REGISTER
CMP R5,R4 ;IS DATA CORRECT
BEQ 2$ ;BR IF YES
HLT 2 ;DATA ERROR
SCOPI ;SW09 UP?
MOV #3$,LOCK ;NEW SCOPI
BIC #BIT12,(R1) ;CLEAR BIT 12
CLR R5 ;CLEAR "EXPECTED"
MOV (R1),R4 ;READ CONTROL STATUS REGISTER
BEQ 4$ ;BR IF ZERO
HLT 2 ;DATA ERROR BIT12 NOT CLEARED
SCOPI ;SW09 UP?
SCOPE ;SCOPE THIS TEST

```

2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082

013736 012737 000015 001226 TST15:
013744 012737 014034 001216
013752 012737 013762 001220
013760 104412
013762 013701 001410 1\$:
013766 012705 000001
013772 010511
013774 011104
013776 020504
014000 001401
014002 104002
014004 104401 2\$:
014006 012737 014014 001220
014014 042711 000001 3\$:
014020 005005
014022 011104
014024 001402
014026 104002
014030 104401
014032 104400 4\$:

***** TEST 15 *****
*CONTROL OUT REGISTER WRITE/READ TEST
*SET BIT0, VERIFY BIT0 WAS SET
*CLEAR BIT0, VERIFY BIT0 WAS CLEARED

TEST 15

MOV #15, TSTNO
MOV #TST16, NEXT
MOV #1\$, LOCK
MSTCLR ; MASTER CLEAR MB200-YC
MOV DMCTL, R1 ; PUT REGISTER ADDRESS IN R1
MOV #BIT0, R5 ; PUT DATA IN "EXPECTED"
MOV R5, (R1) ; WRITE BIT 0
MOV (R1), R4 ; READ CONTROL OUT REGISTER
CMP R5, R4 ; IS DATA CORRECT
BEQ 2\$; BR IF YES
HLT 2 ; DATA ERROR
SCOPE1 ; SW09 UP?
MOV #3\$, LOCK ; NEW SCOPE1
BIC #BIT0, (R1) ; CLEAR BIT 0
CLR R5 ; CLEAR "EXPECTED"
MOV (R1), R4 ; READ CONTROL OUT REGISTER
BEQ 4\$; BR IF ZERO
HLT 2 ; DATA ERROR BIT0 NOT CLEARED
SCOPE1 ; SW09 UP?
SCOPE ; SCOPE THIS TEST

***** TEST 16 *****
*CONTROL OUT REGISTER WRITE/READ TEST
*SET BIT1, VERIFY BIT1 WAS SET
*CLEAR BIT1, VERIFY BIT1 WAS CLEARED

TEST 16

014034 012737 000016 001226 TST16:
014042 012737 014132 001216
014050 012737 014060 001220
014056 104412
014060 013701 001410 1\$:
014064 012705 000002
014070 010511
014072 011104
014074 020504
014076 001401
014100 104002
014102 104401 2\$:
014104 012737 014112 001220
014112 042711 000002 3\$:

MOV #16, TSTNO
MOV #TST17, NEXT
MOV #1\$, LOCK
MSTCLR ; MASTER CLEAR MB200-YC
MOV DMCTL, R1 ; PUT REGISTER ADDRESS IN R1
MOV #BIT1, R5 ; PUT DATA IN "EXPECTED"
MOV R5, (R1) ; WRITE BIT 1
MOV (R1), R4 ; READ CONTROL OUT REGISTER
CMP R5, R4 ; IS DATA CORRECT
BEQ 2\$; BR IF YES
HLT 2 ; DATA ERROR
SCOPE1 ; SW09 UP?
MOV #3\$, LOCK ; NEW SCOPE1
BIC #BIT1, (R1) ; CLEAR BIT 1

```

2083 014116 005005 CLR R5 ;CLEAR "EXPECTED"
2084 014120 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
2085 014122 001402 BEQ 4$ ;BR IF ZERO
2086 014124 104002 HLT 2 ;DATA ERROR BIT1 NOT CLEARED
2087 014126 104401 SCOPE1 ;SW09 UP?
2088 014130 104400 4$: SCOPE ;SCOPE THIS TEST

```

```

;***** TEST 17 *****
;CONTROL OUT REGISTER WRITE/READ TEST
;SET BIT2, VERIFY BIT2 WAS SET
;CLEAR BIT2, VERIFY BIT2 WAS CLEARED
;*****

```

TEST 17

```

2099 014132 012737 000017 001226 TST17: MOV #17,TSTNO
2100 014140 012737 014230 001216 MOV #TST20,NEXT
2101 014146 012737 014156 001220 MOV #1$,LOCK
2102 014154 104412 MSTCLR ;MASTER CLEAR M8200-YC
2103 014156 013701 001410 1$: MOV DMCTL,R1 ;PUT REGISTER ADDRESS IN R1
2104 014162 012705 000004 MOV #BIT2,R5 ;PUT DATA IN "EXPECTED"
2105 014166 010511 MOV R5,(R1) ;WRITE BIT 2
2106 014170 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
2107 014172 020504 CMP R5,R4 ;IS DATA CORRECT
2108 014174 001401 BEQ 2$ ;BR IF YES
2109 014176 104002 HLT 2 ;DATA ERROR
2110 014200 104401 SCOPE1 ;SW09 UP?
2111 014202 012737 014210 001220 2$: MOV #3$,LOCK ;NEW SCOPE1
2112 014210 042711 000004 3$: BIC #BIT2,(R1) ;CLEAR BIT 2
2113 014214 005005 CLR R5 ;CLEAR "EXPECTED"
2114 014216 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
2115 014220 001402 BEQ 4$ ;BR IF ZERO
2116 014222 104002 HLT 2 ;DATA ERROR BIT2 NOT CLEARED
2117 014224 104401 SCOPE1 ;SW09 UP?
2118 014226 104400 4$: SCOPE ;SCOPE THIS TEST

```

```

;***** TEST 20 *****
;CONTROL OUT REGISTER WRITE/READ TEST
;SET BIT6, VERIFY BIT6 WAS SET
;CLEAR BIT6, VERIFY BIT6 WAS CLEARED
;*****

```

TEST 20

```

2129 014230 012737 000020 001226 TST20: MOV #20,TSTNO
2130 014236 012737 014326 001216 MOV #TST21,NEXT
2131 014244 012737 014254 001220 MOV #1$,LOCK
2132 014252 104412 MSTCLR ;MASTER CLEAR M8200-YC
2133 014254 013701 001410 1$: MOV DMCTL,R1 ;PUT REGISTER ADDRESS IN R1
2134 014260 012705 000100 MOV #BIT6,R5 ;PUT DATA IN "EXPECTED"
2135 014264 010511 MOV R5,(R1) ;WRITE BIT 6
2136 014266 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER

```

```

2137 014270 020504          CMP      R5,R4          ; IS DATA CORRECT
2138 014272 001401          BEQ      2$             ; BR IF YES
2139 014274 104002          HLT      2             ; DATA ERROR
2140 014276 104401          SCOPE1          ; SW09 UP?
2141 014300 012737 014306 001220 2$: MOV      #3$,LOCK      ; NEW SCOPE1
2142 014306 042711 000100 3$: BIC      #BIT6,(R1)    ; CLEAR BIT 6
2143 014312 005005          CLR      R5            ; CLEAR "EXPECTED"
2144 014314 011104          MOV      (R1),R4       ; READ CONTROL OUT REGISTER
2145 014316 001402          BEQ      4$             ; BR IF ZERO
2146 014320 104002          HLT      2             ; DATA ERROR BIT6 NOT CLEARED
2147 014322 104401          SCOPE1          ; SW09 UP?
2148 014324 104400          SCOPE          ; SCOPE THIS TEST

```

```

***** TEST 21 *****
*CONTROL OUT REGISTER WRITE/READ TEST
*SET BIT7, VERIFY BIT7 WAS SET
*CLEAR BIT7, VERIFY BIT7 WAS CLEARED
*****

```

TEST 21

```

2159 014326 012737 000021 001226 TST21: MOV      #21,TSTNO
2160 014334 012737 014424 001216      MOV      #TST22,NEXT
2161 014342 012737 014352 001220      MOV      #1$,LOCK
2162 014350 104412          MSTCLR          ; MASTER CLEAR M8200-YC
2163 014352 013701 001410 1$: MOV      DMCTL,R1      ; PUT REGISTER ADDRESS IN R1
2164 014356 012705 000200      MOV      #BIT7,R5     ; PUT DATA IN "EXPECTED"
2165 014362 010511          MOV      R5,(R1)      ; WRITE BIT 7
2166 014364 011104          MOV      (R1),R4       ; READ CONTROL OUT REGISTER
2167 014366 020504          CMP      R5,R4         ; IS DATA CORRECT
2168 014370 001401          BEQ      2$             ; BR IF YES
2169 014372 104002          HLT      2             ; DATA ERROR
2170 014374 104401          SCOPE1          ; SW09 UP?
2171 014376 012737 014404 001220 2$: MOV      #3$,LOCK      ; NEW SCOPE1
2172 014404 042711 000200 3$: BIC      #BIT7,(R1)    ; CLEAR BIT 7
2173 014410 005005          CLR      R5            ; CLEAR "EXPECTED"
2174 014412 011104          MOV      (R1),R4       ; READ CONTROL OUT REGISTER
2175 014414 001402          BEQ      4$             ; BR IF ZERO
2176 014416 104002          HLT      2             ; DATA ERROR BIT7 NOT CLEARED
2177 014420 104401          SCOPE1          ; SW09 UP?
2178 014422 104400          SCOPE          ; SCOPE THIS TEST

```

```

***** TEST 22 *****
*CONTROL OUT REGISTER WRITE/READ TEST
*SET BIT12, VERIFY BIT12 WAS SET
*CLEAR BIT12, VERIFY BIT12 WAS CLEARED
*****

```

TEST 22

```

2189 014424 012737 000022 001226 TST22: MOV      #22,TSTNO
2190 014432 012737 014522 001216      MOV      #TST23,NEXT

```

```

2191 014440 012737 014450 001220      MOV      #1$,LOCK
2192 014446 104412      MSTCLR
2193 014450 013701 001410      1$:     MOV      DMCTL,R1      ;MASTER CLEAR M8200-YC
2194 014454 012705 010000      MOV      #BIT12,R5    ;PUT REGISTER ADDRESS IN R1
2195 014460 010511      MOV      R5,(R1)      ;PUT DATA IN "EXPECTED"
2196 014462 011104      MOV      (R1),R4      ;WRITE BIT 12
2197 014464 020504      CMP      R5,R4        ;READ CONTROL OUT REGISTER
2198 014466 001401      BEQ      2$           ;IS DATA CORRECT
2199 014470 104002      HLT      2            ;BR IF YES
2200 014472 104401      SCOPE1              ;DATA ERROR
2201 014474 012737 014502 001220      2$:     MOV      #3$,LOCK    ;SW09 UP?
2202 014502 042711 010000      3$:     BIC      #BIT12,(R1) ;NEW SCOPE1
2203 014506 005005      CLR      R5           ;CLEAR BIT 12
2204 014510 011104      MOV      (R1),R4      ;CLEAR "EXPECTED"
2205 014512 001402      BEQ      4$           ;READ CONTROL OUT REGISTER
2206 014514 104002      HLT      2            ;BR IF ZERO
2207 014516 104401      SCOPE1              ;DATA ERROR BIT12 NOT CLEARED
2208 014520 104400      4$:     SCOPE              ;SW09 UP?
2209                                     ;SCOPE THIS TEST
    
```

```

;***** TEST 23 *****
;CONTROL OUT REGISTER WRITE/READ TEST
;SET BIT13, VERIFY BIT13 WAS SET
;CLEAR BIT13, VERIFY BIT13 WAS CLEARED
;*****
    
```

TEST 23

```

2219 014522 012737 000023 001226 TST23:  MOV      #23,TSTNO
2220 014530 012737 014620 001216      MOV      #TST24,NEXT
2221 014536 012737 014546 001220      MOV      #1$,LOCK
2222 014544 104412      MSTCLR
2223 014546 013701 001410      1$:     MOV      DMCTL,R1      ;MASTER CLEAR M8200-YC
2224 014552 012705 020000      MOV      #BIT13,R5    ;PUT REGISTER ADDRESS IN R1
2225 014556 010511      MOV      R5,(R1)      ;PUT DATA IN "EXPECTED"
2226 014560 011104      MOV      (R1),R4      ;WRITE BIT 13
2227 014562 020504      CMP      R5,R4        ;READ CONTROL OUT REGISTER
2228 014564 001401      BEQ      2$           ;IS DATA CORRECT
2229 014566 104002      HLT      2            ;BR IF YES
2230 014570 104401      SCOPE1              ;DATA ERROR
2231 014572 012737 014600 001220      2$:     MOV      #3$,LOCK    ;SW09 UP?
2232 014600 042711 020000      3$:     BIC      #BIT13,(R1) ;NEW SCOPE1
2233 014604 005005      CLR      R5           ;CLEAR BIT 13
2234 014606 011104      MOV      (R1),R4      ;CLEAR "EXPECTED"
2235 014610 001402      BEQ      4$           ;READ CONTROL OUT REGISTER
2236 014612 104002      HLT      2            ;BR IF ZERO
2237 014614 104401      SCOPE1              ;DATA ERROR BIT13 NOT CLEARED
2238 014616 104400      4$:     SCOPE              ;SW09 UP?
2239                                     ;SCOPE THIS TEST
    
```

```

;***** TEST 24 *****
;PORT4 REGISTER WRITE/READ TEST
;FLOAT A ONE THROUGH PORT4 REGISTER
;FLOAT A ZERO THROUGH PORT4 REGISTER
    
```

2240
2241
2242
2243
2244

```

2245 ;:*****
2246 ;
2247 ; TEST 24
2248 ;-----
2249 014620 012737 000024 001226 TST24: MOV #24,TSTNO
2250 014626 012737 014744 001216 MOV #TST25,NEXT
2251 014634 012737 014654 001220 MOV #64$,LOCK
2252 014642 104412 MSTCLR ;MASTER CLEAR M8200-YC
2253 014644 013701 001412 MOV DMP04,R1 ;PUT REGISTER ADDRESS IN R1
2254 014650 012700 000001 MOV #1,R0 ;START WITH BIT0
2255 014654 64$: MOV R0,R5 ;PUT "EXPECTED" IN R5
2256 014654 010005 MOV R5,(R1) ;WRITE PORT4 REGISTER
2257 014656 010511 MOV (R1),R4 ;READ PORT4 REGISTER
2258 014660 011104 CMP R5,R4 ;COMPARE EXPECTED AND FOUND
2259 014662 020504 BEQ 65$ ;BR IF OK
2260 014664 001401 HLT 2 ;WRITE/READ ERROR
2261 014666 104002 65$: SCOPI ;LOOP TO 64$ IF SW09=1
2262 014670 104401 CLC ;CLEAR CARRY
2263 014672 000241 ROL R0 ;SHIFT TO NEXT BIT
2264 014674 006100 BNE 64$ ;BR IF NOT DONE YET?
2265 014676 001366 MOV #66$,LOCK ;NEW SCOPI
2266 014700 012737 014712 001220 MOV #1,R0 ;START WITH BIT0
2267 014706 012700 000001 66$: COM R0 ;CHANGE TO A FLOATING ZERO
2268 014712 66$: MOV R0,R5 ;PUT "EXPECTED" IN R5
2269 014712 005100 MOV R5,(R1) ;WRITE PORT4 REGISTER
2270 014714 010005 MOV (R1),R4 ;READ PORT4 REGISTER
2271 014716 010511 CMP R5,R4 ;COMPARE EXPECTED AND FOUND
2272 014720 011104 BEQ 67$ ;BR IF OK
2273 014722 020504 HLT 2 ;WRITE/READ ERROR
2274 014724 001401 67$: SCOPI ;LOOP TO 66$ IF SW09=1
2275 014726 014002 COM R0 ;CHANGE BACK TO A FLOATING ONE
2276 014730 104401 CLC ;CLEAR CARRY
2277 014732 005100 ROL R0 ;SHIFT TO NEXT BIT
2278 014734 000241 BNE 66$ ;BR IF NOT DONE YET?
2279 014736 006100 SCOPE ;SCOPE THIS TEST
2280 014740 001364
2281 014742 104400
2282
2283
2284 ;***** TEST 25 *****
2285 ;*PORT6 REGISTER WRITE/READ TEST
2286 ;*FLOAT A ONE THROUGH PORT6 REGISTER
2287 ;*FLOAT A ZERO THROUGH PORT6 REGISTER
2288 ;:*****
2289
2290 ; TEST 25
2291 ;-----
2292 014744 012737 000025 001226 TST25: MOV #25,TSTNO
2293 014752 012737 015070 001216 MOV #TST26,NEXT
2294 014760 012737 015000 001220 MOV #64$,LOCK
2295 014766 104412 MSTCLR ;MASTER CLEAR M8200-YC
2296 014770 013701 001414 MOV DMP06,R1 ;PUT REGISTER ADDRESS IN R1
2297 014774 012700 000001 MOV #1,R0 ;START WITH BIT0
2298 015000 64$:

```


M05

2353	015152	013701	001404			MOV	DMCSR,R1	;BASE MB200-YC ADDRESS TO R1
2354	015156	012700	000001			MOV	#1,RO	;RESTART PATTERN AT 1
2355	015162	012737	015170	001220		MOV	#3\$,LOCK	;NEW SCOPI
2356	015170	110005			3\$:	MOV8	RO,R5	;PUT DATA IN "EXPECTED"
2357	015172	111109				MOV8	(R1),R4	;READ DMC REGISTER INTO "FOUND"
2358	015174	020504				CMP	R5,R4	;IS DATA CORRECT
2359	015176	001401				BEQ	4\$;BR IF YES
2360	015200	104002				HLT	2	;DUAL ADDRESSING ERROR
2361	015202	104401			4\$:	SCOPI		;SW09=1?
2362	015204	105721				TSTB	(R1)+	;NEXT REGISTER
2363	015206	005200				INC	RO	;INCREMENT PATTERN
2364	015210	022700	000011			CMP	#11,RO	;LAST REGISTER?
2365	015214	001365				BNE	3\$;BR IF NO
2366	015216	104400				SCOPE		;SCOPE THIS TEST
2367								
2368								
2369								
2370								
2371								
2372								
2373								
2374								
2375								
2376								
2377	015220	012737	000027	001226	TST27:	MOV	#27,TSTNO	
2378	015226	012737	015360	001216		MOV	#TST30,NEXT	
2379	015234	012737	015252	001220		MOV	#1\$,LOCK	
2380								
2381	015242	104412				MSTCLR		;R1 CONTAINS BASE MB200-YC ADDRESS
2382	015244	012711	003000			MOV	#BIT9!BIT10,(R1)	;MASTER CLEAR MB200-YC
2383	015250	005005				CLR	R5	;SEL6 IS NOW THE IR
2384	015252	010561	000006		1\$:	MOV	R5,6(R1)	;PUT "EXPECTED" IN R5
2385	015256	016104	000006			MOV	6(R1),R4	;CLEAR THE IR
2386	015262	020504				CMP	R5,R4	;READ THE IR
2387	015264	001401				BEQ	2\$;IS IT CLEARED?
2388	015266	104023				HLT	23	;BR IF YES
2389	015270	104401			2\$:	SCOPI		;ERROR IR IS NOT CLEAR
2390	015272	012737	015304	001220		MOV	#3\$,LOCK	;LOOP TO 1\$ IF SW09=1
2391	015300	012705	177777			MOV	#-1,R5	;NEW SCOPI
2392	015304	010561	000006		3\$:	MOV	R5,6(R1)	;PUT "EXPECTED" IN R5
2393	015310	016104	000006			MOV	6(R1),R4	;WRITE ALL ONES TO THE IR
2394	015314	020504				CMP	R5,R4	;READ THE IR
2395	015316	001401				BEQ	4\$;IS IT ALL ONES?
2396	015320	104023				HLT	23	;BR IF YES
2397	015322	104401			4\$:	SCOPI		;ERROR IR IS NOT = ALL ONES
2398	015324	012737	015334	001220		MOV	#5\$,LOCK	;LOOP TO 3\$ IF SW09=1
2399	015332	005005				CLR	R5	;NEW SCOPI
2400	015334	000005			5\$:	RESET		;PUT "EXPECTED" IN R5
2401	015336	012711	003000			MOV	#BIT9!BIT10,(R1)	;BUS RESET
2402	015342	016104	000006			MOV	6(R1),R4	;SEL6 IS IR
2403	015346	020504				CMP	R5,R4	;READ THE IR
2404	015350	001401				BEQ	6\$;IS IT CLEARED?
2405	015352	104023				HLT	23	;BR IF YES
2406	015354	104401			6\$:	SCOPI		;ERROR IR IS NOT CLEARED
								;LOOP TO 5\$ IF SW09=1

```

***** TEST 27 *****
;MAINTENANCE INSTRUCTION REGISTER TEST
;VERIFY THAT THE MAINT IR CAN BE WRITTEN TO ALL ZEROS'
;AND ALL ONES'. VERIFY THAT IT IS CLEARED ON A BUS RESET.
*****
  
```

; TEST 27

```

-----
TST27: MOV #27,TSTNO
MOV #TST30,NEXT
MOV #1$,LOCK

;R1 CONTAINS BASE MB200-YC ADDRESS
;MASTER CLEAR MB200-YC
;SEL6 IS NOW THE IR
;PUT "EXPECTED" IN R5
;CLEAR THE IR
;READ THE IR
;IS IT CLEARED?
;BR IF YES
;ERROR IR IS NOT CLEAR
;LOOP TO 1$ IF SW09=1
;NEW SCOPI
;PUT "EXPECTED" IN R5
;WRITE ALL ONES TO THE IR
;READ THE IR
;IS IT ALL ONES?
;BR IF YES
;ERROR IR IS NOT = ALL ONES
;LOOP TO 3$ IF SW09=1
;NEW SCOPI
;PUT "EXPECTED" IN R5
;BUS RESET
;SEL6 IS IR
;READ THE IR
;IS IT CLEARED?
;BR IF YES
;ERROR IR IS NOT CLEARED
;LOOP TO 5$ IF SW09=1
  
```

```

2407 015356 104400
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418 015360 012737 000030 001226 TST30:
2419 015366 012737 015522 001216
2420 015374 012737 015412 001220
2421
2422 015402 104412
2423 015404 012711 003000
2424 015410 005005
2425 015412 010561 000006 1$:
2426 015416 016104 000006
2427 015422 020504
2428 015424 001401
2429 015426 104023
2430 015430 104401 2$:
2431 015432 012737 015444 001220
2432 015440 012705 177777
2433 015444 010561 000006 3$:
2434 015450 016104 000006
2435 015454 020504
2436 015456 001401
2437 015460 104023
2438 015462 104401 4$:
2439 015464 012737 015474 001220
2440 015472 005005
2441 015474 052711 040000 5$:
2442 015500 012711 003000
2443 015504 016104 000006
2444 015510 020504
2445 015512 001401
2446 015514 104023
2447 015516 104401 6$:
2448 015520 104400
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460

```

SCOPE ;SCOPE THIS TEST

```

***** TEST 30 *****
*MAINTENANCE INSTRUCTION REGISTER TEST
*VERIFY THAT THE MAINT IR CAN BE WRITTEN TO ALL ZEROS'
*AND ALL ONES'. VERIFY THAT IT IS CLEARED ON A MASTER RESET.
*****

```

TEST 30

```

-----
MOV #30,TSTNO
MOV #TST31,NEXT
MOV #1$,LOCK
;R1 CONTAINS BASE M8200-YC ADDRESS
MSTCLR ;MASTER CLEAR M8200-YC
MOV #BIT9:BIT10,(R1) ;SEL6 IS NOW THE IR
CLR R5 ;PUT "EXPECTED" IN R5
MOV R5,6(R1) ;CLEAR THE IR
MOV 6(R1),R4 ;READ THE IR
CMP R5,R4 ;IS IT CLEARED?
BEQ 2$ ;BR IF YES
HLT 23 ;ERROR IR IS NOT CLEAR
SCOPI ;LOOP TO 1$ IF SW09=1
MOV #3$,LOCK ;NEW SCOPI
MOV #-1,R5 ;PUT "EXPECTED" IN R5
MOV R5,6(R1) ;WRITE ALL ONES TO THE IR
MOV 6(R1),R4 ;READ THE IR
CMP R5,R4 ;IS IT ALL ONES?
BEQ 4$ ;BR IF YES
HLT 23 ;ERROR IR IS NOT = ALL ONES
SCOPI ;LOOP TO 3$ IF SW09=1
MOV #5$,LOCK ;NEW SCOPI
CLR R5 ;PUT "EXPECTED" IN R5
BIS #BIT14,(R1) ;MASTER CLEAR
MOV #BIT9:BIT10,(R1) ;SEL6 IS IR
MOV 6(R1),R4 ;READ THE IR
CMP R5,R4 ;IS IT CLEARED?
BEQ 6$ ;BR IF YES
HLT 23 ;ERROR, IR IS NOT CLEARED
SCOPI ;LOOP TO 5$ IF SW09=1
SCOPE ;SCOPE THIS TEST

```

```

***** TEST 31 *****
*MICRO PROCESSOR TEST
*LOAD DMP06 WITH A MICRO-PROCESSOR INSTRUCTION, CLOCK IT
*VERIFY INSTRUCTION EXECUTED PROPERLY
*INSTRUCTION SHOULD MOVE IBUS*4 TO IBUS*5, IBUS*4 IS ALL 1'S
*AND IBUS*5 IS ALL 0'S. RESULT SHOULD BE ALL 1'S IN SEL4
*****

```

TEST 31

```

2461 015522 012737 000031 001226 TST31: MOV #31,TSTNO
2462 015530 012737 015606 001216 MOV #TST32,NEXT
2463                                     ;R1 CONTAINS BASE M8200-YC ADDRESS
2464 015536 104412 MSTCLR ;MASTER CLEAR M8200-YC
2465 015540 012761 000377 000004 MOV #377,4(R1) ;PORT4 HI-BYTE=0'S LO-BYTE=1'S
2466 015546 012711 001000 MOV #BIT9,(R1) ;SET ROMI
2467 015552 012761 121105 000006 MOV #121105,6(R1) ;INSTRUCTION TO PORT6
2468 015560 052711 001400 BIS #BIT8:BIT9,(R1) ;CLK INSTRUCTION, MOVE IBUS*4 TO IBUS*5
2469 015564 000240 NOP
2470 015566 012705 177777 MOV #-1,R5 ;PUT "EXPECTED" IN R5
2471 015572 016104 000004 MOV 4(R1),R4 ;PUT "FOUND" INTO R4
2472 015576 020504 CMP R5,R4 ;IS DATA CORRECT
2473 015600 001401 BEQ 1$ ;BR IF YES
2474 015602 104003 HLT 3 ;ERROR
2475 015604 104400 1$: SCOPE ;SCOPE THIS TEST
2476
2477
2478 ;***** TEST 32 *****
2479 ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
2480 ;*FLOAT A 1 THROUGH IBUS* REGISTER 0
2481 ;*FLOAT A 0 THROUGH IBUS* REGISTER 0
2482 ;*****
2483
2484 ; TEST 32
2485 ;-----
2486 015606 012737 000032 001226 TST32: MOV #32,TSTNO
2487 015614 012737 016006 001216 MOV #TST33,NEXT
2488 015622 012737 015642 001220 MOV #64$,LOCK
2489                                     ;R1 CONTAINS BASE M8200-YC ADDRESS
2490 015630 104412 MSTCLR ;MASTER CLEAR M8200-YC
2491 015632 012702 000000 MOV #0,R2 ;SAVE REGISTER ADDRESS FOR TYPEOUT
2492 015636 012700 000001 MOV #1,R0 ;START WITH BIT 0
2493 015642 64$:
2494 015642 010061 000004 MOV R0,4(R1) ;PUT PATTERN INTO PORT4
2495 015646 042761 000030 000004 BIC #30,4(R1) ;CLEAR UNWANTED BITS
2496 015654 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2497 015656 121100 121100!0 MOV DATA TO IBUS* REGISTER 0
2498 015660 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2499 015662 121005 121005!<0*20> READ FROM IBUS* REGISTER 0
2500 015664 010005 MOV R0,R5 ;PUT EXPECTED IN R5
2501 015666 042705 000030 BIC #30,R5 ;CLEAR UNWANTED BITS
2502 015672 116104 000005 MOVB 5(R1),R4 ;PUT "FOUND" INTO R4
2503 015676 120504 CMPB R5,R4 ;DATA CORRECT?
2504 015700 001401 BEQ 65$ ;BR IF YES
2505 015702 104004 HLT 4 ;ERROR
2506 015704 104401 65$: SCOP1 ;SWD9=1?
2507 015706 000241 CLC ;CLEAR CARRY
2508 015710 106100 ROLB R0 ;SHIFT BIT IN R0
2509 015712 001353 BNE 64$ ;IF R0=0 THEN DONE
2510 015714 012737 015730 001220 MOV #67$,LOCK ;NEW SCOP1
2511 015722 012700 000001 MOV #1,R0 ;START WITH BIT 0
2512 015726 005100 69$: COM ;CHANGE TO FLOATING ZERO
2513 015730 67$:
2514 015730 010061 000004 MOV R0,4(R1) ;PUT PATTERN INTO PORT4
    
```

```

2515 015734 042761 000030 000004      BIC      #30,4(R1)      ;CLEAR UNWANTED BITS
2516 015742 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2517 015744 121100      121100!0      ;MOV DATA TO IBUS* REGISTER 0
2518 015746 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2519 015750 121005      121005!(<0*20> ;READ FROM IBUS* REGISTER 0
2520 015752 010005      MOV      R0,R5      ;PUT EXPECTED IN R5
2521 015754 042705 000030      BIC      #30,R5      ;CLEAR UNWANTED BITS
2522 015760 116104 000005      MOV      5(R1),R4    ;PUT "FOUND" INTO R4
2523 015764 120504      CMP      R5,R4      ;DATA CORRECT?
2524 015766 001401      BEQ      68$        ;BR IF YES
2525 015770 104004      HLT      4          ;ERROR
2526 015772 104401      SCOPE1     ;SW09=1?
2527 015774 005100      COM      R0        ;CHANGE TO FLOATING 1
2528 015776 000241      CLC      ;CLEAR CARRY
2529 016000 106100      ROL      R0        ;SHIFT BIT IN R0
2530 016002 001351      BNE      69$        ;IF R0=0 THEN DONE
2531 016004 104400      SCOPE      ;SCOPE THIS TEST

```

```

***** TEST 33 *****
;MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
;FLOAT A 1 THROUGH IBUS* REGISTER 2
;FLOAT A 0 THROUGH IBUS* REGISTER 2
*****

```

```

; TEST 33
-----

```

```

2542 016006 012737 000033 001226 TST33:  MOV      #33,TSTNO
2543 016014 012737 016206 001216      MOV      #TST34,NEXT
2544 016022 012737 016042 001220      MOV      #64$,LOCK
2545                                     ;R1 CONTAINS BASE M8200-YC ADDRESS
2546 016030 104412      MSTCLR     ;MASTER CLEAR M8200-YC
2547 016032 012702 000002      MOV      #2,R2      ;SAVE REGISTER ADDRESS FOR TYPEOUT
2548 016036 012700 000001      MOV      #1,R0      ;START WITH BIT 0
2549 016042                                     64$:
2550 016042 010061 000004      MOV      R0,4(R1)   ;PUT PATTERN INTO PORT4
2551 016046 042761 000070 000004      BIC      #70,4(R1)  ;CLEAR UNWANTED BITS
2552 016054 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2553 016056 121102      121100!2      ;MOV DATA TO IBUS* REGISTER 2
2554 016060 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2555 016062 121045      121005!(<2*20> ;READ FROM IBUS* REGISTER 2
2556 016064 010005      MOV      R0,R5      ;PUT EXPECTED IN R5
2557 016066 042705 000070      BIC      #70,R5      ;CLEAR UNWANTED BITS
2558 016072 116104 000005      MOV      5(R1),R4    ;PUT "FOUND" INTO R4
2559 016076 120504      CMP      R5,R4      ;DATA CORRECT?
2560 016100 001401      BEQ      65$        ;BR IF YES
2561 016102 104004      HLT      4          ;ERROR
2562 016104 104401      SCOPE1     ;SW09=1?
2563 016106 000241      CLC      ;CLEAR CARRY
2564 016110 106100      ROL      R0        ;SHIFT BIT IN R0
2565 016112 001353      BNE      64$        ;IF R0=0 THEN DONE
2566 016114 012737 016130 001220      MOV      #67$,LOCK ;NEW SCOPE1
2567 016122 012700 000001      MOV      #1,R0      ;START WITH BIT 0
2568 016126 005100      COM      R0        ;CHANGE TO FLOATING ZERO

```

```

2569 016130          67$: MOV      RO,4(R1)      ;PUT PATTERN INTO PORT4
2570 016130 010061 000004      BIC      #70,4(R1)    ;CLEAR UNWANTED BITS
2571 016134 042761 000070 000004 ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2572 016142 104414          121100!2  ;MOV DATA TO IBUS* REGISTER 2
2573 016144 121102          ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2574 016146 104414          121005!<2*20> ;READ FROM IBUS* REGISTER 2
2575 016150 121045          MOV      RO,R5       ;PUT EXPECTED IN R5
2576 016152 010005          BIC      #70,R5      ;CLEAR UNWANTED BITS
2577 016154 042705 000070      MOVVB   5(R1),R4     ;PUT "FOUND" INTO R4
2578 016160 116104 000005      CMPB   R5,R4       ;DATA CORRECT?
2579 016164 120504          BEQ     68$         ;BR IF YES
2580 016166 001401          HLT     4           ;ERROR
2581 016170 104004          68$: SCOP1        ;SW09=1?
2582 016172 104401          COM     RO         ;CHANGE TO FLOATING 1
2583 016174 005100          CLC                    ;CLEAR CARRY
2584 016176 000241          ROLB   RO         ;SHIFT BIT IN RO
2585 016200 106100          BNE    69$         ;IF RO=0 THEN DONE
2586 016202 001351          SCOPE                    ;SCOPE THIS TEST
2587 016204 104400
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598 016206 012737 000034 001226 TST34: MOV      #34,TSTNO
2599 016214 012737 016362 001216      MOV      #TST35,NEXT
2600 016222 012737 016242 001220      MOV      #64$,LOCK
2601
2602 016230 104412          MSTCLR                    ;R1 CONTAINS BASE M8200-YC ADDRESS
2603 016232 012702 000004      MOV      #4,R2         ;MASTER CLEAR M8200-YC
2604 016236 012700 000001      MOV      #1,RO        ;SAVE REGISTER ADDRESS FOR TYPEOUT
2605 016242          64$: MOV      RO,4(R1)    ;START WITH BIT 0
2606 016242 010061 000004      ROMCLK   ;PUT PATTERN INTO PORT4
2607 016246 104414          121100!4  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2608 016250 121104          ROMCLK   ;MOV DATA TO IBUS* REGISTER 4
2609 016252 104414          121005!<4*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2610 016254 121105          MOV      RO,R5       ;READ FROM IBUS* REGISTER 4
2611 016256 010005          MOVVB   5(R1),R4     ;PUT EXPECTED IN R5
2612 016260 116104 000005      CMPB   R5,R4       ;PUT "FOUND" INTO R4
2613 016264 120504          BEQ     65$         ;DATA CORRECT?
2614 016266 001401          HLT     4           ;BR IF YES
2615 016270 104004          65$: SCOP1        ;ERROR
2616 016272 104401          CLC                    ;SW09=1?
2617 016274 000241          ROLB   RO         ;CLEAR CARRY
2618 016276 106100          BNE    64$         ;SHIFT BIT IN RO
2619 016300 001360          MOV     #67$,LOCK   ;IF RO=0 THEN DONE
2620 016302 012737 016316 001220      MOV     #1,RO       ;NEW SCOPE
2621 016310 012700 000001          MOV     RO         ;START WITH BIT 0
2622 016314 005100          69$: COM     RO     ;CHANGE TO FLOATING ZERO

```

```

2623 016316          67$:  MOV      RO,4(R1)          ;PUT PATTERN INTO PORT4
2624 016316 010061 000004  ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2625 016322 104414          121100!4          ;MOV DATA TO IBUS* REGISTER 4
2626 016324 121104          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2627 016326 104414          121005!<4*20>      ;READ FROM IBUS* REGISTER 4
2628 016330 121105          MOV      RO,R5          ;PUT EXPECTED IN R5
2629 016332 010005          MOV     5(R1),R4        ;PUT "FOUND" INTO R4
2630 016334 116104 000005  MOV     R5,R4          ;DATA CORRECT?
2631 016340 120504          CMPB   R5,R4          ;BR IF YES
2632 016342 001401          BEQ    68$            ;ERROR
2633 016344 104004          HLT    4              ;SW09=1?
2634 016346 104401          68$:  SCOP1          ;CHANGE TO FLOATING 1
2635 016350 005100          COM    RO              ;CLEAR CARRY
2636 016352 000241          CLC                    ;SHIFT BIT IN RO
2637 016354 106100          ROLB   RO              ;IF RO=0 THEN DONE
2638 016356 001356          BNE   69$            ;SCOPE THIS TEST
2639 016360 104400          SCOPE
2640
2641
2642          ;***** TEST 35 *****
2643          ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
2644          ;*FLOAT A 1 THROUGH IBUS* REGISTER 5
2645          ;*FLOAT A 0 THROUGH IBUS* REGISTER 5
2646          ;*****
2647
2648          ; TEST 35
2649          ;-----
2650 016362 012737 000035 001226 TST35: MOV     #35,TSTNO
2651 016370 012737 016536 001216      MOV     #TST36,NEXT
2652 016376 012737 016416 001220      MOV     #64$,LOCK
2653
2654 016404 104412          MSTCLR          ;R1 CONTAINS BASE M8200-YC ADDRESS
2655 016406 012702 000005  MOV     #5,R2        ;MASTER CLEAR M8200-YC
2656 016412 012700 000001  MOV     #1,RO        ;SAVE REGISTER ADDRESS FOR TYPEOUT
2657 016416          64$:  MOV     #1,RO        ;START WITH BIT 0
2658 016416 010061 000004  MOV     RO,4(R1)      ;PUT PATTERN INTO PORT4
2659 016422 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2660 016424 121105          121100!5          ;MOV DATA TO IBUS* REGISTER 5
2661 016426 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2662 016430 121125          121005!<5*20>      ;READ FROM IBUS* REGISTER 5
2663 016432 010005          MOV     RO,R5          ;PUT EXPECTED IN R5
2664 016434 116104 000005  MOV     5(R1),R4        ;PUT "FOUND" INTO R4
2665 016440 120504          CMPB   R5,R4          ;DATA CORRECT?
2666 016442 001401          BEQ    65$            ;BR IF YES
2667 016444 104004          HLT    4              ;ERROR
2668 016446 104401          65$:  SCOP1          ;SW09=1?
2669 016450 000241          CLC                    ;CLEAR CARRY
2670 016452 106100          ROLB   RO              ;SHIFT BIT IN RO
2671 016454 001360          BNE   64$            ;IF RO=0 THEN DONE
2672 016456 012737 016472 001220  MOV     #67$,LOCK      ;NEW SCOP1
2673 016464 012700 000001  MOV     #1,RO          ;START WITH BIT 0
2674 016470 005100          69$:  COM    RO          ;CHANGE TO FLOATING ZERO
2675 016472          67$:
2676 016472 010061 000004  MOV     RO,4(R1)      ;PUT PATTERN INTO PORT4
    
```

```

2677 016476 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2678 016500 121105 121100!5 ;MOV DATA TO IBUS* REGISTER 5
2679 016502 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2680 016504 121125 121005!<5*20> ;READ FROM IBUS* REGISTER 5
2681 016506 010005 MOV RO,R5 ;PUT EXPECTED IN R5
2682 016510 116104 000005 MOVB 5(R1),R4 ;PUT "FOUND" INTO R4
2683 016514 120504 CMPB R5,R4 ;DATA CORRECT?
2684 016516 001401 BEQ 68$ ;BR IF YES
2685 016520 104004 HLT 4 ;ERROR
2686 016522 104401 68$: SCOP1 ;SW09=1?
2687 016524 005100 COM RO ;CHANGE TO FLOATING 1
2688 016526 000241 CLC ;CLEAR CARRY
2689 016530 106100 ROLB RO ;SHIFT BIT IN RO
2690 016532 001356 BNE 69$ ;IF RO=0 THEN DONE
2691 016534 104400 SCOPE ;SCOPE THIS TEST
2692
2693
2694
2695 ;***** TEST 36 *****
2696 ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
2697 ;*FLOAT A 1 THROUGH IBUS* REGISTER 10
2698 ;*FLOAT A 0 THROUGH IBUS* REGISTER 10
2699 ;*THE NPR R0 BIT (BIT 0) IS MASKED DURING THIS TEST
2700 ;*****
2701 ; TEST 36
2702 ;-----
2703 016536 012737 000036 001226 TST36: MOV #36,TSTNO
2704 016544 012737 016736 001216 MOV #TST37,NEXT
2705 016552 012737 016572 001220 MOV #64$,LOCK
2706
2707 016560 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
2708 016562 012702 000010 MOV #10,R2 ;MASTER CLEAR M8200-YC
2709 016566 012700 000001 MOV #1,R0 ;SAVE REGISTER ADDRESS FOR TYPEOUT
2710 016572 64$: ;START WITH BIT 0
2711 016572 010061 000004 MOV RO,4(R1) ;PUT PATTERN INTO PORT4
2712 016576 042761 000141 000004 BIC #141,4(R1) ;CLEAR UNWANTED BITS
2713 016604 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2714 016606 121110 121100!10 ;MOV DATA TO IBUS* REGISTER 10
2715 016610 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2716 016612 121205 121005!<10*20> ;READ FROM IBUS* REGISTER 10
2717 016614 010005 MOV RO,R5 ;PUT EXPECTED IN R5
2718 016616 042705 000141 BIC #141,R5 ;CLEAR UNWANTED BITS
2719 016622 116104 000005 MOVB 5(R1),R4 ;PUT "FOUND" INTO R4
2720 016626 120504 CMPB R5,R4 ;DATA CORRECT?
2721 016630 001401 BEQ 65$ ;BR IF YES
2722 016632 104004 HLT 4 ;ERROR
2723 016634 104401 65$: SCOP1 ;SW09=1?
2724 016636 000241 CLC ;CLEAR CARRY
2725 016640 106100 ROLB RO ;SHIFT BIT IN RO
2726 016642 001353 BNE 64$ ;IF RO=0 THEN DONE
2727 016644 012737 016660 001220 MOV #67$,LOCK ;NEW SCOPE
2728 016652 012700 000001 MOV #1,R0 ;START WITH BIT 0
2729 016656 005100 69$: COM RO ;CHANGE TO FLOATING ZERO
2730 016660 67$:
    
```

2731	016660	010061	000004		MOV	RO,4(R1)	:PUT PATTERN INTO PORT4
2732	016664	042761	000141	000004	BIC	#141,4(R1)	:CLEAR UNWANTED BITS
2733	016672	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2734	016674	121110			121100!10		:MOV DATA TO IBUS* REGISTER 10
2735	016676	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2736	016700	121205			121005!<10*20>		:READ FROM IBUS* REGISTER 10
2737	016702	010005			MOV	RO,R5	:PUT EXPECTED IN R5
2738	016704	042705	000141		BIC	#141,R5	:CLEAR UNWANTED BITS
2739	016710	116104	000005		MOV	5(R1),R4	:PUT "FOUND" INTO R4
2740	016714	120504			CMPB	R5,R4	:DATA CORRECT?
2741	016716	001401			BEQ	68\$:BR IF YES
2742	016720	104004			HLT	4	:ERROR
2743	016722	104401		68\$:	SCOP1		:SW09=1?
2744	016724	005100			COM	RO	:CHANGE TO FLOATING 1
2745	016726	000241			CLC		:CLEAR CARRY
2746	016730	106100			ROLB	RO	:SHIFT BIT IN RO
2747	016732	001351			BNE	69\$:IF RO=0 THEN DONE
2748	016734	104400			SCOPE		:SCOPE THIS TEST

2749
 2750
 2751 :***** TEST 37 *****
 2752 :*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
 2753 :*FLOAT A 1 THROUGH IBUS* REGISTER 11
 2754 :*FLOAT A 0 THROUGH IBUS* REGISTER 11
 2755 :*THE BR RQ BIT, PGM CLOCK BIT, FORCE POWER FAIL BIT
 2756 :*(BITS 7,4,1) ARE ALL MASKED DURING THIS TEST
 2757 :*****
 2758
 2759

2760					:	TEST 37	
2761	016736	012737	000037	001226	TST37:	MOV	#37,TSTNO
2762	016744	012737	017156	001216		MOV	#TST40,NEXT
2763	016752	012737	016772	001220		MOV	#64\$,LOCK
2764							
2765	016760	104412				MSTCLR	
2766	016762	012702	000011			MOV	#11,R2
2767	016766	012700	000001			MOV	#1,RO
2768	016772				64\$:		
2769	016772	010061	000004			MOV	RO,4(R1)
2770	016776	042761	000262	000004		BIC	#262,4(R1)
2771	017004	104414				ROMCLK	
2772	017006	121111				121100!11	
2773	017010	104414				ROMCLK	
2774	017012	121225				121005!<11*20>	
2775	017014	010005				MOV	RO,R5
2776	017016	042705	000262			BIC	#262,R5
2777	017022	052705	000020			BIS	#20,R5
2778	017026	116104	000005			MOV	5(R1),R4
2779	017032	052704	000020			BIS	#20,R4
2780	017036	120504				CMPB	R5,R4
2781	017040	001401				BEQ	65\$
2782	017042	104004				HLT	4
2783	017044	104401		65\$:		SCOP1	
2784	017046	000241				CLC	

:R1 CONTAINS BASE M8200-YC ADDRESS
 :MASTER CLEAR M8200-YC
 :SAVE REGISTER ADDRESS FOR TYPEOUT
 :START WITH BIT 0
 :PUT PATTERN INTO PORT4
 :CLEAR UNWANTED BITS
 :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 :MOV DATA TO IBUS* REGISTER 11
 :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 :READ FROM IBUS* REGISTER 11
 :PUT EXPECTED IN R5
 :CLEAR UNWANTED BITS
 :ADD THESE BITS
 :PUT "FOUND" INTO R4
 :ADD THIS BIT
 :DATA CORRECT?
 :BR IF YES
 :ERROR
 :SW09=1?
 :CLEAR CARRY

```

2785 017050 106100          ROLB   R0          :SHIFT BIT IN R0
2786 017052 001347          BNE    64$         :IF R0=0 THEN DONE
2787 017054 012737 017070 001220  MOV    #67$,LOCK  :NEW SCOPE
2788 017062 012700 000001      MOV    #1,R0      :START WITH BIT 0
2789 017066 005100          COM    R0         :CHANGE TO FLOATING ZERO
2790 017070          69$:
2791 017070 010061 000004      MOV    R0,4(R1)   :PUT PATTERN INTO PORT4
2792 017074 042761 000262 000004  BIC    #262,4(R1) :CLEAR UNWANTED BITS
2793 017102 104414          ROMCLK          :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2794 017104 121111          ROMCLK 121100!11 :MOV DATA TO IBUS* REGISTER 11
2795 017106 104414          ROMCLK          :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2796 017110 121225          ROMCLK 121005!<11*20> :READ FROM IBUS* REGISTER 11
2797 017112 010005          MOV    R0,R5     :PUT EXPECTED IN R5
2798 017114 042705 000262      BIC    #262,R5    :CLEAR UNWANTED BITS
2799 017120 052705 000020      BIS    #20,R5     :ADD THESE BITS
2800 017124 116104 000005      MOVB  5(R1),R4   :PUT "FOUND" INTO R4
2801 017130 052704 000020      BIS    #20,R4    :ADD THIS BIT
2802 017134 120504          CMPB  R5,R4      :DATA CORRECT?
2803 017136 001401          BEQ   68$        :BR IF YES
2804 017140 104004          HLT   4          :ERROR
2805 017142 104401          68$:
2806 017144 005100          COM    R0         :SW09=1?
2807 017146 000241          CLC          :CHANGE TO FLOATING 1
2808 017150 106100          ROLB   R0         :CLEAR CARRY
2809 017152 001345          BNE    69$         :SHIFT BIT IN R0
2810 017154 104400          SCOPE          :IF R0=0 THEN DONE
                :SCOPE THIS TEST

```

```

:***** TEST 40 *****
:MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
:FLOAT A 1 THROUGH IBUS REGISTER 0
:FLOAT A 0 THROUGH IBUS REGISTER 0
:*****

```

TEST 40

```

2820          :-----
2821 017156 012737 000040 001226 TST40: MOV    #40,TSTNO
2822 017164 012737 017332 001216  MOV    #TST41,NEXT
2823 017172 012737 017212 001220  MOV    #64$,LOCK
2824          :R1 CONTAINS BASE M8200-YC ADDRESS
2825 017200 104412          MSTCLR          :MASTER CLEAR M8200-YC
2826 017202 012702 000000      MOV    #0,R2     :SAVE REGISTER ADDRESS FOR TYPEOUT
2827 017206 012700 000001      MOV    #1,R0     :START WITH BIT 0
2828 017212          64$:
2829 017212 010061 000004      MOV    R0,4(R1)   :PUT PATTERN INTO PORT4
2830 017216 104414          ROMCLK          :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2831 017220 122100          ROMCLK 122100!0   :MOV DATA TO IBUS REGISTER 0
2832 017222 104414          ROMCLK          :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2833 017224 021005          ROMCLK 21005!<0*20> :READ FROM IBUS REGISTER 0
2834 017226 010005          MOV    R0,R5     :PUT EXPECTED IN R5
2835 017230 116104 000005      MOVB  5(R1),R4   :PUT "FOUND" INTO R4
2836 017234 120504          CMPB  R5,R4      :DATA CORRECT?
2837 017236 001401          BEQ   65$        :BR IF YES
2838 017240 104005          HLT   5          :ERROR

```

```

2839 017242 104401          65$: SCOP1          ;SW09=1?
2840 017244 000241          CLC              ;CLEAR CARRY
2841 017246 106100          ROLB            RO ;SHIFT BIT IN RO
2842 017250 001360          BNE            64$ ;IF RO=0 THEN DONE
2843 017252 012737 017266 001220 MOV            #67$, LOCK ;NEW SCOP1
2844 017260 012700 000001 MOV            #1, RO ;START WITH BIT 0
2845 017264 005100          COM            RO ;CHANGE TO FLOATING ZERO
2846 017266          67$:
2847 017266 010061 000004 MOV            RO, 4(R1) ;PUT PATTERN INTO PORT4
2848 017272 104414          ROMCLK         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2849 017274 122100          122100!0       ;MOV DATA TO IBUS REGISTER 0
2850 017276 104414          ROMCLK         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2851 017300 021005          21005!<0*20> ;READ FROM IBUS REGISTER 0
2852 017302 010005          MOV            RO, R5 ;PUT EXPECTED IN R5
2853 017304 116104 000005 MOVB          5(R1), R4 ;PUT "FOUND" INTO R4
2854 017310 120504          CMPB          R5, R4 ;DATA CORRECT?
2855 017312 001401          BEQ            68$ ;BR IF YES
2856 017314 104005          HLT            5 ;ERROR
2857 017316 104401          68$: SCOP1          ;SW09=1?
2858 017320 005100          COM            RO ;CHANGE TO FLOATING 1
2859 017322 000241          CLC              ;CLEAR CARRY
2860 017324 106100          ROLB            RO ;SHIFT BIT IN RO
2861 017326 001356          BNE            69$ ;IF RO=0 THEN DONE
2862 017330 104400          SCOPE          ;SCOPE THIS TEST

```

```

2865 ;***** TEST 41 *****
2866 ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
2867 ;*FLOAT A 1 THROUGH IBUS REGISTER 1
2868 ;*FLOAT A 0 THROUGH IBUS REGISTER 1
2869 ;*****

```

```

2870 ;
2871 ; TEST 41
2872 ;-----
2873 017332 012737 000041 001226 TST41: MOV            #41, TSTNO
2874 017340 012737 017506 001216 MOV            #TST42, NEXT
2875 017346 012737 017366 001220 MOV            #64$, LOCK
2876          MSTCLR          ;R1 CONTAINS BASE M8200-YC ADDRESS
2877 017354 104412          MOV            #1, R2 ;MASTER CLEAR M8200-YC
2878 017356 012702 000001 MOV            #1, RO ;SAVE REGISTER ADDRESS FOR TYPEOUT
2879 017362 012700 000001 MOV            #1, RO ;START WITH BIT 0
2880 017366          64$:
2881 017366 010061 000004 MOV            RO, 4(R1) ;PUT PATTERN INTO PORT4
2882 017372 104414          ROMCLK         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2883 017374 122101          122100!1       ;MOV DATA TO IBUS REGISTER 1
2884 017376 104414          ROMCLK         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2885 017400 021025          21005!<1*20> ;READ FROM IBUS REGISTER 1
2886 017402 010005          MOV            RO, R5 ;PUT EXPECTED IN R5
2887 017404 116104 000005 MOVB          5(R1), R4 ;PUT "FOUND" INTO R4
2888 017410 120504          CMPB          R5, R4 ;DATA CORRECT?
2889 017412 001401          BEQ            65$ ;BR IF YES
2890 017414 104005          HLT            5 ;ERROR
2891 017416 104401          65$: SCOP1          ;SW09=1?
2892 017420 000241          CLC              ;CLEAR CARRY

```

```

2893 017422 106100          ROLB   RO          ;SHIFT BIT IN RO
2894 017424 001360          BNE    #64$,LOCK  ;IF RO=0 THEN DONE
2895 017426 012737 017442 001220  MOV    #67$,LOCK  ;NEW SCOPE
2896 017434 012700 000001          MOV    #1,RO      ;START WITH BIT 0
2897 017440 005100          COM    RO         ;CHANGE TO FLOATING ZERO
2898 017442          69$:
2899 017442 010061 000004          MOV    RO,4(R1)   ;PUT PATTERN INTO PORT4
2900 017446 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2901 017450 122101          122100!1        ;MOV DATA TO IBUS REGISTER 1
2902 017452 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2903 017454 021025          21005!<1*20>   ;READ FROM IBUS REGISTER 1
2904 017456 010005          MOV    RO,R5     ;PUT EXPECTED IN R5
2905 017460 116104 000005          MOVB  5(R1),R4   ;PUT "FOUND" INTO R4
2906 017464 120504          CMPB  R5,R4     ;DATA CORRECT?
2907 017466 001401          BEQ   #68$      ;BR IF YES
2908 017470 104005          HLT   5         ;ERROR
2909 017472 104401          68$:          SCOPE1         ;SW09=1?
2910 017474 005100          COM    RO         ;CHANGE TO FLOATING 1
2911 017476 000241          CLC                    ;CLEAR CARRY
2912 017500 106100          ROLB  RO         ;SHIFT BIT IN RO
2913 017502 001356          BNE   #69$      ;IF RO=0 THEN DONE
2914 017504 104400          SCOPE          ;SCOPE THIS TEST

```

```

;***** TEST 42 *****
;MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
;FLOAT A 1 THROUGH IBUS REGISTER 2
;FLOAT A 0 THROUGH IBUS REGISTER 2
;*****

```

TEST 42

```

2924          ;-----
2925 017506 012737 000042 001226  TST42:  MOV    #42,TSTNO
2926 017514 012737 017662 001216          MOV    #TST43,NEXT
2927 017522 012737 017542 001220          MOV    #64$,LOCK
2928          ;R1 CONTAINS BASE M8200-YC ADDRESS
2929 017530 104412          MSTCLR          ;MASTER CLEAR M8200-YC
2930 017532 012702 000002          MOV    #2,R2     ;SAVE REGISTER ADDRESS FOR TYPEOUT
2931 017536 012700 000001          MOV    #1,RO     ;START WITH BIT 0
2932 017542          64$:
2933 017542 010061 000004          MOV    RO,4(R1)   ;PUT PATTERN INTO PORT4
2934 017546 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2935 017550 122102          122100!2        ;MOV DATA TO IBUS REGISTER 2
2936 017552 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2937 017554 021045          21005!<2*20>   ;READ FROM IBUS REGISTER 2
2938 017556 010005          MOV    RO,R5     ;PUT EXPECTED IN R5
2939 017560 116104 000005          MOVB  5(R1),R4   ;PUT "FOUND" INTO R4
2940 017564 120504          CMPB  R5,R4     ;DATA CORRECT?
2941 017566 001401          BEQ   #65$      ;BR IF YES
2942 017570 104005          HLT   5         ;ERROR
2943 017572 104401          65$:          SCOPE1         ;SW09=1?
2944 017574 000241          CLC                    ;CLEAR CARRY
2945 017576 106100          ROLB  RO         ;SHIFT BIT IN RO
2946 017600 001360          BNE   #64$      ;IF RO=0 THEN DONE

```

```

2947 017602 012737 017616 001220      MOV      #67$,LOCK      ;NEW SCOPE1
2948 017610 012700 000001              MOV      #1,R0          ;START WITH BIT 0
2949 017614 005100              COM      R0              ;CHANGE TO FLOATING ZERO
2950 017616              69$:
2951 017616 010061 000004              MOV      R0,4(R1)      ;PUT PATTERN INTO PORT4
2952 017622 104414              ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2953 017624 122102 122100!2          ;MOV DATA TO IBUS REGISTER 2
2954 017626 104414              ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2955 017630 021045 21005!<2*20>    ;READ FROM IBUS REGISTER 2
2956 017632 010005              MOV      R0,R5          ;PUT EXPECTED IN R5
2957 017634 116104 000005              MOVB    5(R1),R4       ;PUT "FOUND" INTO R4
2958 017640 120504              CMPB    R5,R4          ;DATA CORRECT?
2959 017642 001401              BEQ     68$            ;BR IF YES
2960 017644 104005              HLT     5              ;ERROR
2961 017646 104401              68$: SCOPE1           ;SW09=1?
2962 017650 005100              COM      R0              ;CHANGE TO FLOATING 1
2963 017652 000241              CLC                      ;CLEAR CARRY
2964 017654 106100              ROLB    R0              ;SHIFT BIT IN R0
2965 017656 001356              BNE     69$            ;IF R0=0 THEN DONE
2966 017660 104400              SCOPE                    ;SCOPE THIS TEST

;***** TEST 43 *****
;MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
;FLOAT A 1 THROUGH IBUS REGISTER 3
;FLOAT A 0 THROUGH IBUS REGISTER 3
;*****

; TEST 43
-----
2977 017662 012737 000043 001226 TST43: MOV      #43,TSTNO
2978 017670 012737 020036 001216      MOV      #TST44,NEXT
2979 017676 012737 017716 001220      MOV      #64$,LOCK
2980              ;R1 CONTAINS BASE M8200-YC ADDRESS
2981 017704 104412              MSTCLR          ;MASTER CLEAR M8200-YC
2982 017706 012702 000003              MOV      #3,R2          ;SAVE REGISTER ADDRESS FOR TYPEOUT
2983 017712 012700 000001              MOV      #1,R0          ;START WITH BIT 0
2984 017716              64$:
2985 017716 010061 000004              MOV      R0,4(R1)      ;PUT PATTERN INTO PORT4
2986 017722 104414              ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2987 017724 122103 122100!3          ;MOV DATA TO IBUS REGISTER 3
2988 017726 104414              ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2989 017730 021065 21005!<3*20>    ;READ FROM IBUS REGISTER 3
2990 017732 010005              MOV      R0,R5          ;PUT EXPECTED IN R5
2991 017734 116104 000005              MOVB    5(R1),R4       ;PUT "FOUND" INTO R4
2992 017740 120504              CMPB    R5,R4          ;DATA CORRECT?
2993 017742 001401              BEQ     65$            ;BR IF YES
2994 017744 104005              HLT     5              ;ERROR
2995 017746 104401              65$: SCOPE1           ;SW09=1?
2996 017750 000241              CLC                      ;CLEAR CARRY
2997 017752 106100              ROLB    R0              ;SHIFT BIT IN R0
2998 017754 001360              BNE     64$            ;IF R0=0 THEN DONE
2999 017756 012737 017772 001220      MOV      #67$,LOCK      ;NEW SCOPE1
3000 017764 012700 000001              MOV      #1,R0          ;START WITH BIT 0
    
```

```

3001 017770 005100          69$: COM      R0          ;CHANGE TO FLOATING ZERO
3002 017772                67$:          ;
3003 017772 010061 000004    MOV      R0,4(R1)      ;PUT PATTERN INTO PORT4
3004 017776 104414          ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3005 020000 122103          122100!3    ;MOV DATA TO IBUS REGISTER 3
3006 020002 104414          ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3007 020004 021065          21005!<3*20> ;READ FROM IBUS REGISTER 3
3008 020006 010005          MOV      R0,R5        ;PUT EXPECTED IN R5
3009 020010 116104 000005    MOVB    5(R1),R4     ;PUT "FOUND" INTO R4
3010 020014 120504          CMPB    R5,R4        ;DATA CORRECT?
3011 020016 001401          BEQ     68$          ;BR IF YES
3012 020020 104005          HLT     5            ;ERROR
3013 020022 104401          68$: SCOP1         ;SW09=1?
3014 020024 005100          COM      R0          ;CHANGE TO FLOATING 1
3015 020026 000241          CLC                    ;CLEAR CARRY
3016 020030 106100          ROLB    R0          ;SHIFT BIT IN R0
3017 020032 001356          BNE     69$          ;IF R0=0 THEN DONE
3018 020034 104400          SCOPE                    ;SCOPE THIS TEST
3019
3020
3021          ;***** TEST 44 *****
3022          ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3023          ;*FLOAT A 1 THROUGH IBUS REGISTER 4
3024          ;*FLOAT A 0 THROUGH IBUS REGISTER 4
3025          ;*****
3026
3027          ; TEST 44
3028          ;-----
3029 020036 012737 000044 001226 TST44: MOV      #44,TSTNO
3030 020044 012737 020212 001216    MOV      #TST45,NEXT
3031 020052 012737 020072 001220    MOV      #64$,LOCK
3032
3033 020060 104412          MSTCLR                    ;R1 CONTAINS BASE M8200-YC ADDRESS
3034 020062 012702 000004    MOV      #4,R2          ;MASTER CLEAR M8200-YC
3035 020066 012700 000001    MOV      #1,R0          ;SAVE REGISTER ADDRESS FOR TYPEOUT
3036 020072                64$:          ;START WITH BIT 0
3037 020072 010061 000004    MOV      R0,4(R1)      ;PUT PATTERN INTO PORT4
3038 020076 104414          ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3039 020100 122104          122100!4    ;MOV DATA TO IBUS REGISTER 4
3040 020102 104414          ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3041 020104 021105          21005!<4*20> ;READ FROM IBUS REGISTER 4
3042 020106 010005          MOV      R0,R5        ;PUT EXPECTED IN R5
3043 020110 116104 000005    MOVB    5(R1),R4     ;PUT "FOUND" INTO R4
3044 020114 120504          CMPB    R5,R4        ;DATA CORRECT?
3045 020116 001401          BEQ     65$          ;BR IF YES
3046 020120 104005          HLT     5            ;ERROR
3047 020122 104401          65$: SCOP1         ;SW09=1?
3048 020124 000241          CLC                    ;CLEAR CARRY
3049 020126 106100          ROLB    R0          ;SHIFT BIT IN R0
3050 020130 001360          BNE     64$          ;IF R0=0 THEN DONE
3051 020132 012737 020146 001220    MOV      #67$,LOCK
3052 020140 012700 000001    MOV      #1,R0
3053 020144 005100          69$: COM      R0          ;CHANGE TO FLOATING ZERO
3054 020146                67$:          ;

```

3055	020146	010061	000004		MOV	RO,4(R1)		: PUT PATTERN INTO PORT4
3056	020152	104414			ROMCLK			: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3057	020154	122104			122100!4			: MOV DATA TO IBUS REGISTER 4
3058	020156	104414			ROMCLK			: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3059	020160	021105			21005!<4*20>			: READ FROM IBUS REGISTER 4
3060	020162	010005			MOV	RO,R5		: PUT EXPECTED IN R5
3061	020164	116104	000005		MOVB	5(R1),R4		: PUT "FOUND" INTO R4
3062	020170	120504			CMPB	R5,R4		: DATA CORRECT?
3063	020172	001401			BEQ	68\$: BR IF YES
3064	020174	104005			HLT	5		: ERROR
3065	020176	104401		68\$:	SCOPI			: SW09=1?
3066	020200	005100			COM	RO		: CHANGE TO FLOATING 1
3067	020202	000241			CLC			: CLEAR CARRY
3068	020204	106100			ROLB	RO		: SHIFT BIT IN RO
3069	020206	001356			BNE	69\$: IF RO=0 THEN DONE
3070	020210	104400			SCOPE			: SCOPE THIS TEST
3071								
3072								
3073								
3074								
3075								
3076								
3077								
3078								
3079								
3080								
3081	020212	012737	000045	001226	TST45:	MOV	#45,TSTNO	
3082	020220	012737	020366	001216		MOV	#TST46,NEXT	
3083	020226	012737	020246	001220		MOV	#64\$,LOCK	
3084								
3085	020234	104412				MSTCLR		: R1 CONTAINS BASE MB200-YC ADDRESS
3086	020236	012702	000005			MOV	#5,R2	: MASTER CLEAR MB200-YC
3087	020242	012700	000001			MOV	#1,RO	: SAVE REGISTER ADDRESS FOR TYPEOUT
3088	020246				64\$:			: START WITH BIT 0
3089	020246	010061	000004			MOV	RO,4(R1)	: PUT PATTERN INTO PORT4
3090	020252	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3091	020254	122105				122100!5		: MOV DATA TO IBUS REGISTER 5
3092	020256	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3093	020260	021125				21005!<5*20>		: READ FROM IBUS REGISTER 5
3094	020262	010005				MOV	RO,R5	: PUT EXPECTED IN R5
3095	020264	116104	000005			MOVB	5(R1),R4	: PUT "FOUND" INTO R4
3096	020270	120504				CMPB	R5,R4	: DATA CORRECT?
3097	020272	001401				BEQ	65\$: BR IF YES
3098	020274	104005				HLT	5	: ERROR
3099	020276	104401			65\$:	SCOPI		: SW09=1?
3100	020300	000241				CLC		: CLEAR CARRY
3101	020302	106100				ROLB	RO	: SHIFT BIT IN RO
3102	020304	001360				BNE	64\$: IF RO=0 THEN DONE
3103	020306	012737	020322	001220		MOV	#67\$,LOCK	: NEW SCOPI
3104	020314	012700	000001			MOV	#1,RO	: START WITH BIT 0
3105	020320	005100			69\$:	COM	RO	: CHANGE TO FLOATING ZERO
3106	020322				67\$:			
3107	020322	010061	000004			MOV	RO,4(R1)	: PUT PATTERN INTO PORT4
3108	020326	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```

:***** TEST 45 *****
:MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
:FLOAT A 1 THROUGH IBUS REGISTER 5
:FLOAT A 0 THROUGH IBUS REGISTER 5
:*****

```

: TEST 45

```

:-----
:MOV #45,TSTNO
:MOV #TST46,NEXT
:MOV #64$,LOCK

```

```

:R1 CONTAINS BASE MB200-YC ADDRESS
:MASTER CLEAR MB200-YC
:SAVE REGISTER ADDRESS FOR TYPEOUT
:START WITH BIT 0

```



```

3163 020510 021145          21005!<6*20>          ;READ FROM IBUS REGISTER 6
3164 020512 010005          MOV          RO,R5          ;PUT EXPECTED IN R5
3165 020514 116104 000005  MOVB         5(R1),R4      ;PUT "FOUND" INTO R4
3166 020520 120504          CMPB         R5,R4          ;DATA CORRECT?
3167 020522 001401          BEQ          68$           ;BR IF YES
3168 020524 104005          HLT          5             ;ERROR
3169 020526 104401          SCOPI                   ;SW09=1?
3170 020530 005100          COM          RO            ;CHANGE TO FLOATING 1
3171 020532 000241          CLC                               ;CLEAR CARRY
3172 020534 106100          ROLB         RO            ;SHIFT BIT IN RO
3173 020536 001356          BNE          69$           ;IF RO=0 THEN DONE
3174 020540 104400          SCOPE                   ;SCOPE THIS TEST
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185 020542 012737 000047 001226 TST47:  MOV          #47,ISTNO
3186 020550 012737 020716 001216  MOV          #TST50,NEXT
3187 020556 012737 020576 001220  MOV          #64$,LOCK
3188
3189 020564 104412          MSTCLR                   ;R1 CONTAINS BASE M8200-YC ADDRESS
3190 020566 012702 000007  MOV          #7,R2          ;MASTER CLEAR M8200-YC
3191 020572 012700 000001  MOV          #1,RO         ;SAVE REGISTER ADDRESS FOR TYPEOUT
3192
3193 020576 010061 000004          64$:  MOV          RO,4(R1)       ;START WITH BIT 0
3194 020602 104414          ROMCLK                   ;PUT PATTERN INTO PORT4
3195 020604 122107          122100!7                 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3196 020606 104414          ROMCLK                   ;MOV DATA TO IBUS REGISTER 7
3197 020610 021165          21005!<7*20>             ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3198 020612 010005          MOV          RO,R5          ;READ FROM IBUS REGISTER 7
3199 020614 116104 000005  MOVB         5(R1),R4      ;PUT EXPECTED IN R5
3200 020620 120504          CMPB         R5,R4          ;PUT "FOUND" INTO R4
3201 020622 001401          BEQ          65$           ;DATA CORRECT?
3202 020624 104005          HLT          5             ;BR IF YES
3203 020626 104401          SCOPI                   ;ERROR
3204 020630 000241          CLC                               ;SW09=1?
3205 020632 106100          ROLB         RO            ;CLEAR CARRY
3206 020634 001360          BNE          64$           ;SHIFT BIT IN RO
3207 020636 012737 020652 001220  MOV          #67$,LOCK     ;IF RO=0 THEN DONE
3208 020644 012700 000001  MOV          #1,RO         ;NEW SCOPI
3209 020650 005100          69$:  COM          RO            ;START WITH BIT 0
3210 020652          67$:  MOV          RO,4(R1)       ;CHANGE TO FLOATING ZERO
3211 020652 010061 000004          MOV          RO,4(R1)       ;PUT PATTERN INTO PORT4
3212 020656 104414          ROMCLK                   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3213 020660 122107          122100!7                 ;MOV DATA TO IBUS REGISTER 7
3214 020662 104414          ROMCLK                   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3215 020664 021165          21005!<7*20>             ;READ FROM IBUS REGISTER 7
3216 020666 010005          MOV          RO,R5          ;PUT EXPECTED IN R5
    
```

```

3217 020670 116104 000005          MOVB    5(R1),R4          ;PUT "FOUND" INTO R4
3218 020674 120504          CMPB    R5,R4           ;DATA CORRECT?
3219 020676 001401          BEQ     68$             ;BR IF YES
3220 020700 104005          HLT     5               ;ERROR
3221 020702 104401          68$:   SCOP1           ;SW09=1?
3222 020704 005100          COM     R0              ;CHANGE TO FLOATING 1
3223 020706 000241          CLC                    ;CLEAR CARRY
3224 020710 106100          ROLB   R0              ;SHIFT BIT IN R0
3225 020712 001356          BNE    69$             ;IF R0=0 THEN DONE
3226 020714 104400          SCOPE                    ;SCOPE THIS TEST
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237 020716 012737 000050 001226 TST50: MOV     #50,TSTNO
3238 020724 012737 021144 001216 MOV     #TST51,NEXT
3239 020732 012737 020750 001220 MOV     #1$,LOCK
3240
3241 020740 104412          MSTCLR                    ;R1 CONTAINS BASE M8200-YC ADDRESS
3242 020742 012700 000001          MOV     #1,R0            ;MASTER CLEAR M8200-YC
3243 020746 005002          CLR     R2               ;START WITH A ONE
3244 020750 010203          1$:   MOV     R2,R3        ;R2 CONTAINS ADDRESS OF REGISTER
3245 020752 010061 000004          MOV     R0,4(R1)         ;R3=REGISTER ADDRESS
3246 020756 042737 000017 020772 BIC     #17,5$           ;WRITE DATA TO PORT4
3247 020764 050337 020772          BIS     R3,5$           ;CLEAR ADDRESS FIELD OF INSTRUCTION
3248 020770 104414          ROMCLK                    ;ADD ADDRESS TO INSTRUCTION
3249 020772 122100          5$:   122100             ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3250 020774 006303          ASL     R3               ;MOVE DATA TO IBUS REGISTER
3251 020776 006303          ASL     R3               ;SHIFT ADDRESS
3252 021000 006303          ASL     R3               ;4 TIMES TO GET
3253 021002 006303          ASL     R3               ;IT TO BITS 4-7
3254 021004 042737 000360 021020 BIC     #360,6$         ;OF NEXT INSTRUCTION
3255 021012 050337 021020          BIS     R3,6$           ;CLEAR ADDRESS FIELD
3256 021016 104414          ROMCLK                    ;ADD ADDRESS TO INSTRUCTION
3257 021020 021005          6$:   21005             ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3258 021022 010005          MOV     R0,R5           ;READ FROM IBUS REGISTER
3259 021024 116104 000005          MOVB   5(R1),R4         ;PUT "EXPECTED" IN R5
3260 021030 120504          CMPB   R5,R4           ;PUT "FOUND" IN R4
3261 021032 001401          BEQ    2$              ;IS DATA CORRECT?
3262 021034 104005          HLT    5               ;BR IF YES
3263 021036 104401          2$:   SCOP1           ;DATA ERROR
3264 021040 005200          INC     R0              ;SW09=1?
3265 021042 005202          INC     R2              ;INCREMENT PATTERN
3266 021044 022702 000010          CMP     #7+1,R2 ;LAST ADDRESS DONE? ;INCREMENT REGISTER ADDRESS
3267 021050 001337          BNE    1$              ;BR IF NO
3268 021052 012737 021070 001220 MOV     #3$,LOCK        ;NEW SCOP1
3269 021060 012700 000001          MOV     #1,R0           ;RESTART PATTERN TO 1
3270 021064 005002          CLR     R2              ;RESTART AT ADDRESS 0

```

```

;***** TEST 50 *****
;MICRO PROCESSOR IBUS DUAL ADDRESS TEST
;WRITE ALL IBUS REGISTERS WITH INCREMENTING PATTERN
;READ ALL IBUS REGISTERS TO VERIFY CORRECT ADDRESSING
;*****

```

```

; TEST 50

```

```

3271 021066 005003          CLR      R3          ;RESTART AT ADDRESS 0
3272 021070 042737 000360 021104 3$: BIC      #360,7$    ;CLEAR ADDRESS FIELD OF INSTRUCTION
3273 021076 050337 021104          BIS      R3,7$      ;ADD ADDRESS TO INSTRUCTION
3274 021102 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3275 021104 021005          21005          ;READ FROM IBUS REGISTER
3276 021106 010005          MOV      R0,R5     ;PUT "EXPECTED" IN R5
3277 021110 116104 000005  MOVB     5(R1),R4   ;PUT "FOUND" IN R4
3278 021114 120504          CMPB     R5,R4     ;DATA CORRECT?
3279 021116 001401          BEQ      4$       ;BR IF YES
3280 021120 104005          HLT      5        ;DUAL ADDRESSING ERROR
3281 021122 104401          4$: SCOP1          ;SW09=1?
3282 021124 005200          INC      R0       ;INCREMENT PATTERN
3283 021126 005202          INC      R2       ;NEXT ADDRESS
3284 021130 062703 000020  ADD      #20,R3    ;ADD 1 TO ADDRESS IN R3(SHIFTED 4 TIMES)
3285 021134 022702 000010  CMP      #7+1,R2  ;LAST ADDRESS DONE?
3286 021140 001353          BNE     3$       ;BR IF NO
3287 021142 104400          SCOPE          ;SCOPE THIS TEST
3288
3289
3290          ;***** TEST 51 *****
3291          ;*MICRO PROCESSOR BR REGISTER TEST
3292          ;*FLOAT A 1 THROUGH THE BR
3293          ;*FLOAT A 0 THROUGH THE BR
3294          ;*****
3295
3296          ; TEST 51
3297
3298 021144 012737 000051 001226 TST51: MOV      #51,TSTNO
3299 021152 012737 021314 001216  MOV      #TST52,NEXT
3300 021160 012737 021174 001220  MOV      #64$,LOCK
3301
3302 021166 104412          MSTCLR          ;R1 CONTAINS BASE M8200-YC ADDRESS
3303 021170 012700 000001  MOV      #1,R0   ;MASTER CLEAR M8200-YC
3304 021174          64$:          ;START PATTERN WITH BIT0
3305 021174 010061 000004  MOV      R0,4(R1) ;WRITE PATTERN IN PORT4
3306 021200 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3307 021202 120500          120500          ;MOVE DATA TO THE BR REGISTER
3308 021204 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3309 021206 061225          061225          ;MOVE BR TO PORT 5
3310 021210 010005          MOV      R0,R5     ;PUT "EXPECTED" IN R5
3311 021212 116104 000005  MOVB     5(R1),R4   ;PUT "FOUND" IN R4
3312 021216 120504          CMPB     R5,R4     ;DATA CORRECT?
3313 021220 001401          BEQ      65$     ;BR IF YES
3314 021222 104006          HLT      6        ;DATA ERROR
3315 021224 104401          65$: SCOP1
3316 021226 000241          CLC            ;CLEAR CARRY
3317 021230 106100          ROLB     R0       ;SHIFT BIT IN R0
3318 021232 001360          BNE     64$     ;DONE IF R0=0
3319 021234 012737 021250 001220  MOV      #67$,LOCK ;NEW SCOPE
3320 021242 012700 000001  MOV      #1,R0   ;START PATTERN WITH BIT0
3321 021246 005100          69$: COM      R0  ;CHANGE TO FLOATING ZERO
3322 021250          67$:
3323 021250 010061 000004  MOV      R0,4(R1) ;WRITE PATTERN IN PORT4
3324 021254 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
  
```

```

3325 021256 120500          120500          ; MOVE DATA TO THE BR REGISTER
3326 021260 104414          ROMCLK          ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3327 021262 061225          061225          ; MOVE BR TO PORT 5
3328 021264 010005          MOV            RO,R5          ; PUT "EXPECTED" IN R5
3329 021266 116104 000005  MOVB          5(R1),R4        ; PUT "FOUND" IN R4
3330 021272 120504          CMPB          R5,R4          ; DATA CORRECT?
3331 021274 001401          BEQ           68$           ; BR IF YES
3332 021276 104006          HLT           6            ; DATA ERROR
3333 021300 104401          68$: SCOP1          ;
3334 021302 005100          COM           RO            ; CHANGE BACK TO A ONE
3335 021304 000241          CLC          ; CLEAR CARRY
3336 021306 106100          ROLB          RO           ; SHIFT BIT IN RO
3337 021310 001356          BNE           69$           ; DONE IF RO=0
3338 021312 104400          SCOPE          ; SCOPE THIS TEST

```

```

3339
3340
3341 ***** TEST 52 *****
3342 *SCRATCH PAD TEST
3343 *FLOAT A 1 THROUGH EACH SCRATCH PAD LOCATION
3344 *FLOAT A 0 THROUGH EACH SCRATCH PAD LOCATION
3345 :*****
3346
3347
3348

```

```

3349 021314 012737 000052 001226 TST52: MOV #52,TSTNO
3350 021322 012737 021562 001216 MOV #TST53,NEXT
3351 021330 012737 021346 001220 MOV #64$,LOCK
3352
3353 021336 104412          MSTCLR
3354 021340 005002          CLR           R2
3355 021342 012700 000001  MOV #1,RO
3356 021346 042737 000017 021366 64$: BIC #17,65$
3357 021354 050237 021366  BIC R2,65$
3358 021360 010061 000004  MOV R0,4(R1)
3359 021364 104414          ROMCLK
3360 021366 123100          123100          65$:
3361 021370 042737 000017 021404  BIC #17,66$
3362 021376 050237 021404  BIS R2,66$
3363 021402 104414          ROMCLK
3364 021404 040600          040600          66$:
3365 021406 104414          ROMCLK
3366 021410 061225          061225          ; MOVE BR TO PORT 5
3367 021412 010005          MOV            RO,R5          ; PUT "EXPECTED" IN R5
3368 021414 116104 000005  MOVB          5(R1),R4        ; PUT "FOUND" IN R4
3369 021420 120504          CMPB          R5,R4          ; DATA CORRECT
3370 021422 001401          BEQ           67$           ; BR IF YES
3371 021424 104007          HLT           7            ; DATA ERROR
3372 021426 104401          67$: SCOP1          ; SW09=1?
3373 021430 000241          CLC          ; CLEAR CARRY
3374 021432 106100          ROLB          RO           ; SHIFT BIT IN RO
3375 021434 001344          BNE           64$           ; DONE IF RO=0
3376 021436 012737 021452 001220  MOV #69$,LOCK
3377 021444 012700 000001  MOV #1,RO
3378 021450 005100          73$: COM           RO

```

```

; R1 CONTAINS BASE MB200-YC ADDRESS
; MASTER CLEAR MB200-YC
; START AT ADDRESS ZERO
; START WITH BIT0
; CLEAR ADDRESS FIELD OF INSTRUCTION
; ADD ADDRESS TO INSTRUCTION
; WRITE PATTERN TO PORT4
; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
; WRITE SCRATCH PAD(ADDRESS IN R2)
; CLEAR ADDRESS FIELD OF INSTRUCTION
; ADD ADDRESS TO INSTRUCTION
; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
; MOV SP TO BR
; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
; MOVE BR TO PORT5
; PUT "EXPECTED" IN R5
; PUT "FOUND" IN R4
; DATA CORRECT
; BR IF YES
; DATA ERROR
; SW09=1?
; CLEAR CARRY
; SHIFT BIT IN RO
; DONE IF RO=0
; NEW SCOP1
; START WITH BIT0
; CHANGE TO FLOATING ZERO

```

```

3379 021452 042737 000017 021472 69$: BIC #17,70$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
3380 021460 050237 021472 ;ADD ADDRESS TO INSTRUCTION
3381 021464 010061 000004 MOV R0,4(R1) ;WRITE PATTERN TO PORT4
3382 021470 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3383 021472 123100 123100 ;WRITE SCRATCH PAD(ADDRESS IN R2)
3384 021474 042737 000017 021510 70$: BIC #17,71$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
3385 021502 050237 021510 ;ADD ADDRESS TO INSTRUCTION
3386 021506 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3387 021510 040600 040600 71$: MOV SP TO BR
3388 021512 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3389 021514 061225 061225 ;MOVE BR TO PORT5
3390 021516 010005 MOV R0,R5 ;PUT "EXPECTED" IN R5
3391 021520 116104 000005 MOV R5,R4 ;PUT "FOUND" IN R4
3392 021524 120504 CMP R5,R4 ;DATA CORRECT
3393 021526 001401 BEQ 72$ ;BR IF YES
3394 021530 104007 HLT 7 ;DATA ERROR
3395 021532 104401 72$: SCOPE1 ;SW09=1?
3396 021534 005100 COM R0 ;CHANGE BACK TO A ONE
3397 021536 000241 CLC ;CLEAR CARRY
3398 021540 106100 ROL R0 ;SHIFT BIT IN R0
3399 021542 001342 BNE 73$ ;DONE IF R0=0
3400 021544 012700 000001 MOV #1,R0 ;RESTART AT BIT 0
3401 021550 005202 INC R2 ;NEXT SP ADDRESS
3402 021552 022702 000020 CMP #20,R2 ;LAST ADDRESS?
3403 021556 001273 BNE 64$ ;BR IF NO
3404 021560 104400 SCOPE ;SCOPE THIS TEST
    
```

```

3405
3406
3407 ;***** TEST 53 *****
3408 ;SCRATCH PAD DUAL ADDRESSING TEST
3409 ;WRITE AN INCREMENTING PATTERN IN ALL SP LOCATIONS
3410 ;READ ALL SP LOCATIONS TO VERIFY CORRECT ADDRESSING
3411 ;*****
    
```

```

3412 ; TEST 53
3413 ;-----
3414
3415 021562 012737 000053 001226 TST53: MOV #53,TSTNO
3416 021570 012737 022004 001216 MOV #TST54,NEXT
3417 021576 012737 021614 001220 MOV #1$,LOCK
3418
3419 021604 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
3420 021606 012700 000001 MOV #1,R0 ;MASTER CLEAR M8200-YC
3421 021612 005003 CLR R3 ;START WITH A 1
3422 021614 010302 1$: MOV R3,R2 ;ADDRESS 0
3423 021616 042737 000017 021636 BIC #17,2$ ;MOVE ADDRESS TO R2
3424 021624 050237 021636 BIS R2,2$ ;CLEAR ADDRESS FIELD
3425 021630 010061 000004 MOV R0,4(R1) ;ADD ADDRESS TO INSTRUCTION
3426 021634 104414 ROMCLK ;WRITE PATTERN TO PORT4
3427 021636 123100 123100 2$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3428 021640 042737 000017 021654 BIC #17,3$ ;WRITE SP(ADDRESS IN R2)
3429 021646 050237 021654 BIS R2,3$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
3430 021652 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
3431 021654 060600 60600 3$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3432 021656 104414 ROMCLK ;MOV SP TO BR
    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
    
```

```

3433 021660 061225          61225          :MOV BR TO PORTS
3434 021662 010005          MOV      R0,R5          :PUT "EXPECTED" IN R5
3435 021664 116104 000005  MOVB     5(R1),R4       :PUT "FOUND" IN R4
3436 021670 120504          CMPB     R5,R4         :DATA CORRECT?
3437 021672 001401          BEQ      4$           :BR IF YES
3438 021674 104007          HLT      7           :DATA ERROR
3439 021676 104401          4$: SCOP1           :SW09=0
3440 021700 005200          INC      R0          :INCREMENT PATTERN
3441 021702 005203          INC      R3          :NEXT ADDRESS
3442 021704 022703 000020  CMP      #20,R3       :LAST ADDRESS DONE?
3443 021710 001341          BNE     1$           :BR IF NO
3444 021712 012737 021726 001220  MOV      #5$,LOCK     :NEW SCOP1
3445 021720 012700 000001  MOV      #1,R0        :RESTART PATTERN AT 1
3446 021724 005003          CLR      R3          :RESTART AT ADDRESS ZERO
3447 021726 010302          MOV      R3,R2       :PUT ADDRESS IN R2
3448 021730 042737 000017 021744  BIC     #17,6$        :CLEAR ADDRESS FIELD OF INSTRUCTION
3449 021736 050237 021744  BIS     R2,6$        :ADD ADDRESS TO INSTRUCTION
3450 021742 104414          ROMCLK  60600        :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3451 021744 060600          6$: ROMCLK          :MOV SP TO BR
3452 021746 104414          ROMCLK          :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3453 021750 061225          61225          :MOV BR TO PORTS
3454 021752 010005          MOV      R0,R5          :PUT "EXPECTED" IN R5
3455 021754 116104 000005  MOVB     5(R1),R4       :PUT "FOUND" IN R4
3456 021760 120504          CMPB     R5,R4         :DATA CORRECT?
3457 021762 001401          BEQ      7$           :BR IF YES
3458 021764 104007          HLT      7           :SP ADDRESSING ERROR
3459 021766 104401          7$: SCOP1           :SW09=1?
3460 021770 005200          INC      R0          :INCREMENT PATTERN
3461 021772 005203          INC      R3          :NEXT ADDRESS
3462 021774 022703 000020  CMP      #20,R3       :LAST ADDRESS DONE?
3463 022000 001352          BNE     5$           :BR IF NO
3464 022002 104400          SCOPE          :SCOPE THIS TEST

```

```

:***** TEST 54 *****
:*INTERRUPT TEST
:*TEST THAT DEVICE CAN INTERRUPT TO VECTOR A
:*****

```

: TEST 54

```

3473 :-----
3474 022004 012737 000054 001226 TST54: MOV      #54,TSTNO
3475 022012 012737 022100 001216  MOV      #TST55,NEXT
3476 :
3477 022020 000005          RESET
3478 022022 005011          CLR      (R1)
3479 022024 004537 034652  JSR      R5,SETVEC
3480 022030 022072          3$
3481 022032 022070          2$
3482 022034          340          :.BYTE 340,340
3483 022036 012737 000340 177776 1$: MOV      #340,PS
3484 022044 012761 000200 000004  MOV      #200,4(R1)
3485 022052 104414          ROMCLK
3486 022054 121111          121111
:R1 CONTAINS BASE M8200-YC ADDRESS
:BUS RESET
:CLEAR RUN
:SET UP VECTORS
:XX0
:XX4
:LEVEL 7
:PS = LEVEL 7
:WRITE PORT4
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:SET BR RQ IN IBUS* REG 11

```

```

3487 022056 005037 177776          CLR      PS          ;ALLOW INTERRUPT
3488 022062 000240          NOP
3489 022064 104010          HLT      10          ;NO INTERRUPT
3490 022066 000403          BR       4$
3491 022070 104011          2$: HLT      11          ;WRONG VECTOR
3492 022072 012706 001200        3$: MOV      #STACK,SP ;RESET STACK
3493 022076 104400          4$: SCOPE          ;SCOPE THIS TEST
3494
3495
3496
3497
3498
3499
3500
3501
3502
3503 022100 012737 000055 001226 TST55: MOV      #55,TSTNO
3504 022106 012737 022172 001216        MOV      #TST56,NEXT
3505
3506 022114 104412          MSTCLR
3507 022116 004537 034652          JSR      R5,SETVEC ;R1 CONTAINS BASE M8200-YC ADDRESS
3508 022122 022162          2$:          ;MASTER CLEAR M8200-YC
3509 022124 022164          3$:          ;SET UP VECTORS
3510 022126 340          .BYTE 340,340 ;XX0
3511 022130 012737 000340 177776 1$: MOV      #340,PS ;XX4
3512 022136 012761 000300 000004        MOV      #300,4(R1) ;LEVEL 7
3513 022144 104414          ROMCLK ;PS = LEVEL 7
3514 022146 121111          121111 ;WRITE PORT4
3515 022150 005037 177776          CLR      PS          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3516 022154 000240          NOP          ;ALLOW INTERRUPT
3517 022156 104010          HLT      10          ;NO INTERRUPT
3518 022160 000403          BR       4$
3519 022162 104011          2$: HLT      11          ;WRONG VECTOR
3520 022164 012706 001200        3$: MOV      #STACK,SP ;RESET STACK
3521 022170 104400          4$: SCOPE          ;SCOPE THIS TEST
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532 022172 012737 000056 001226 TST56: MOV      #56,TSTNO
3533 022200 012737 022312 001216        MOV      #TST57,NEXT
3534
3535 022206 104412          MSTCLR
3536 022210 012702 000340          MOV      #340,R2 ;R1 CONTAINS BASE M8200-YC ADDRESS
3537 022214 010237 177776          MOV      R2,PS      ;MASTER CLEAR M8200-YC
3538 022220 013700 001366          MOV      STAT1,R0   ;PUT LEVEL 7 IN R2
3539 022224 006200          ASR      R0          ;SET PRIORITY TO 7
3540 022226 006200          ASR      R0          ;GET BR LEVEL OF M8200-YC
                          ;SHIFT R0 4 TIMES
                          ;TO GET PROPER LEVEL

```

```

***** TEST 55 *****
*INTERRUPT TEST
*TEST THAT DEVICE CAN INTERRUPT TO VECTOR B
*****

```

```

TEST 55
-----

```

```

***** TEST 56 *****
*PRIORITY INTERRUPT TEST
*SET PS TO ALL BR LEVELS EQUAL OR GREATER THAN
*THE M8200-YC LEVEL, VERIFY THAT M8200-YC DOES NOT INTERRUPT
*****

```

```

TEST 56
-----

```

```

3541 022230 006200 ASR R0
3542 022232 006200 ASR R0
3543 022234 042700 177437 BIC #177437,R0 ;CLEAR UNWANTED BITS
3544 022240 004537 034652 JSR R5,SETVEC ;SET UP VECTORS
3545 022244 022306 2$ ;A VECTOR
3546 022246 022306 2$ ;B VECTOR
3547 022250 340 340 .BYTE 340,340 ;PRIORITY 7
3548 022252 012761 000200 000004 4$: MOV #200,4(R1) ;LOAD PORT4
3549 022260 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3550 022262 121111 ;SET BR REQUEST
3551 022264 010237 177776 5$: MOV R2,PS ;PUT LEVEL IN R2 IN PS
3552 022270 000240 NOP
3553 022272 020002 CMP R0,R2 ;IS PRESENT PS LEVEL = TO DMC LEVEL
3554 022274 001403 BEQ 1$ ;BR IF YES
3555 022276 162702 000040 SUB #40,R2 ;NO GET NEXT LOWER LEVEL IN R2
3556 022302 000770 BR 5$ ;AND CONTINUE WITH TEST
3557 022304 104400 1$: SCOPE ;SCOPE THIS TEST
3558 022306 104020 2$: HLT 20 ;ERROR UNEXPECTED INTERRUPT
3559 022310 000002 RTI

;***** TEST 57 *****
;PRIORITY INTERRUPT TESTS
;SET PS TO ALL BR LEVELS LESS THAN THE M8200-YC LEVEL
;VERIFY THAT THE M8200-YC WILL INTERRUPT
;*****

; TEST 57
;-----
3570 022312 012737 000057 001226 TST57: MOV #57,TSTNO
3571 022320 012737 022456 001216 MOV #TST60,NEXT ;R1 CONTAINS BASE M8200-YC ADDRESS
3572 022322 000000 MSTCLR ;MASTER CLEAR M8200-YC
3573 022326 104412 MOV #340,R2 ;PUT LEVEL 7 IN R2
3574 022330 012702 000340 MOV R2,PS ;SET PRIORITY TO 7
3575 022334 010237 177776 MOV STAT1,R0 ;GET BR LEVEL OF M8200-YC
3576 022340 013700 001366 ASR R0 ;SHIFT R0 4 TIMES
3577 022344 006200 ASR R0 ;TO GET PROPER LEVEL
3578 022346 006200 ASR R0
3579 022350 006200 ASR R0
3580 022352 006200 ASR R0
3581 022354 042700 177437 BIC #177437,R0 ;CLEAR UNWANTED BITS
3582 022360 010002 MOV R0,R2 ;PUT DMC LEVEL IN R2
3583 022362 162702 000040 SUB #40,R2 ;GET NEXT LOWER LEVEL IN R2
3584 022366 004537 034652 JSR R5,SETVEC ;SET UP VECTORS
3585 022372 022440 2$ ;A VECTOR
3586 022374 022446 3$ ;B VECTOR
3587 022376 340 340 .BYTE 340,340 ;PRIORITY 7
3588 022400 012761 000200 000004 4$: MOV #200,4(R1) ;LOAD PORT4
3589 022406 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3590 022410 121111 ;SET BR REQUEST
3591 022412 010237 177776 5$: MOV R2,PS ;PUT LEVEL IN R2 IN PS
3592 022416 000240 NOP
3593 022420 104010 HLT 10 ;ERROR, NO INTERRUPT
3594 022422 022702 000140 6$: CMP #140,R2 ;IS IT DOWN TO LEVEL 3 YET?
    
```

```

3595 022426 001403          BEQ      1$          ;YES,DMC DID NOT INTERUPT, ERROR
3596 022430 162702 000040  SUB      #40,R2      ;PUT NEXT LOWER LEVEL IN R2
3597 022434 000761          BR       4$          ;CONTINUE TEST
3598 022436 104400          1$:     SCOPE      ;SCOPE THIS TEST
3599 022440 012716 022422  2$:     MOV       #6$, (SP) ;SET UP FOR RTI
3600 022444 000002          RTI
3601 022446 104011          3$:     HLT      11      ;ERROR, WRONG VECTOR
3602 022450 012716 022422  MOV      #6$, (SP)  ;SET UP FOR RTI
3603 022454 000002          RTI
3604
3605
3606          ;***** TEST 60 *****
3607          ;*NPR TEST
3608          ;*TEST OF DATO, 1 WORD FROM UPROC TO 11 MEMORY
3609          ;*****
3610
3611          ; TEST 60
3612          ;-----
3613 022456 012737 000060 001226 TST60: MOV      #60,TSTNO
3614 022464 012737 022562 001216  MOV      #TST61,NEXT
3615
3616 022472 000005          RESET
3617 022474 005011          CLR      (R1)
3618 022476 005061 000004  CLR      4(R1)
3619 022502 004537 034674  JSR      R5,NPRSET
3620 022506 000000          0
3621 022510 177777          -1
3622 022512 022560          3$
3623 022514 022556          2$
3624 022516 005037 022556  CLR      2$
3625 022522 012761 000021 000004  MOV      #21,4(R1)
3626 022530 104414          ROMCLK
3627 022532 121110          121110
3628 022534 000240          NOP
3629 022536 012705 177777  MOV      #-1,R5
3630 022542 013704 022556  MOV      2$,R4
3631 022546 020504          CMP      R5,R4
3632 022550 001401          BEQ      4$
3633 022552 104012          HLT     12
3634 022554 104400          4$:     SCOPE
3635 022556 000000          2$:     0
3636 022560 000000          3$:     0
3637
3638
3639          ;***** TEST 61 *****
3640          ;*NPR TEST
3641          ;*TEST OF DATI, 1 WORD FROM 11 MEMORY TO UPROC
3642          ;*****
3643
3644          ; TEST 61
3645          ;-----
3646 022562 012737 000061 001226 TST61: MOV      #61,TSTNO
3647 022570 012737 022676 001216  MOV      #TST62,NEXT
3648
;R1 CONTAINS BASE M8200-YC ADDRESS

```

```

3649 022576 104412 MSTCLR ;MASTER CLEAR M8200-YC
3650 022600 005061 000004 CLR 4(R1) ;CLR PORT4
3651 022604 004537 034674 JSR R5,NPRSET ;SET UP IBUS REG 0-7
3652 022610 000000 0 ;IN DATA
3653 022612 177777 -1 ;OUT DATA
3654 022614 022674 3$ ;IN BA
3655 022616 022672 2$ ;OUT BA
3656 022620 012737 177777 022674 MOV #-1,3$ ;PUT DATA IN 3$
3657 022626 012761 000001 000004 MOV #1,4(R1) ;WRITE PORT4
3658 022634 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3659 022636 121110 121110 ;SET NPR BITS IN IBUS* REG 11
3660 022640 000240 NOP
3661 022642 012705 177777 MOV #-1,R5 ;PUT "EXPECTED" IN R5
3662 022646 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3663 022650 021004 021004 ;MOVE IN DATA LOW BYTE TO PORT4
3664 022652 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3665 022654 021025 021025 ;MOVE IN DATA HIGH BYTE TO PORT5
3666 022656 016104 000004 MOV 4(R1),R4 ;PUT "FOUND" IN R4
3667 022662 020504 CMP R5,R4 ;DATA CORRECT?
3668 022664 001401 BEQ 4$ ;BR IF YES
3669 022666 104012 HLT 12 ;ERROR NPR FAILED
3670 022670 104400 4$: SCOPE ;SCOPE THIS TEST
3671 022672 000000 2$: 0 ;OUT BA
3672 022674 000000 3$: 0 ;IN BA

```

```

;***** TEST 62 *****
;NPR TEST
;TEST OF DATOB, 1 BYTE FROM UPROC TO 11 MEMORY
;*****

```

TEST 62

```

3681 ;
3682 022676 012737 000062 001226 TST62: MOV #62,TSTNO
3683 022704 012737 023000 001216 MOV #TST63,NEXT
3684 ;
3685 022712 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
3686 022714 005061 000004 CLR 4(R1) ;MASTER CLEAR M8200-YC
3687 022720 004537 034674 JSR R5,NPRSET ;CLR PORT4
3688 022724 000000 0 ;SET UP IBUS REG 0-7
3689 022726 177777 -1 ;IN DATA
3690 022730 022776 3$ ;OUT DATA
3691 022732 022775 2$+1 ;OUT BA
3692 022734 005037 022774 CLR 2$ ;CLEAR 2$
3693 022740 012761 000221 000004 MOV #221,4(R1) ;WRITE PORT4
3694 022746 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3695 022750 121110 121110 ;SET NPR BITS IN IBUS* REG 11
3696 022752 000240 NOP
3697 022754 012705 177400 MOV #177400,R5 ;PUT "EXPECTED" IN R5
3698 022760 013704 022774 MOV 2$,R4 ;PUT "FOUND" IN R4
3699 022764 020504 CMP R5,R4 ;DATA CORRECT?
3700 022766 001401 BEQ 4$ ;BR IF YES
3701 022770 104012 HLT 12 ;ERROR NPR FAILED
3702 022772 104400 4$: SCOPE ;SCOPE THIS TEST

```

3703 022774 000000 2\$: 0 ; OUT BA
3704 022776 000000 3\$: 0 ; IN BA

***** TEST 63 *****
*TEST OF EA BITS 16 AND 17
*DO A DATO TO AN ADDRESS USING OUT BA BITS 16 AND 17
*VERIFY CORRECT RESULTS

TEST 63

3715 023000 012737 000063 001226 TST63:
3716 023006 012737 023136 001216
3718 023014 104412
3719 023016 013737 001412 023044
3720 023024 013737 001412 023042
3721 023032 004537 034674
3722 023036 000000
3723 023040 125252
3724 023042 000000 2\$: 0
3725 023044 000000 1\$: 0
3726 023046 012761 000014 000004
3727 023054 104414
3728 023056 121111
3729 023060 012761 000021 000004
3730 023066 012761 121110 000006
3731 023074 012711 003000
3732 023100 052711 000400
3733 023104 000240
3734 023106 012705 121110
3735 023112 104414
3736 023114 021044
3737 023116 104414
3738 023120 021065
3739 023122 016104 000004
3740 023126 020504
3741 023130 001401
3742 023132 104012
3743 023134 104400 3\$:

MOV #63,TSTNO
MOV #TST64,NEXT
MSTCLR
MOV DMP04,1\$
MOV DMP04,2\$
JSR R5,NPASET
0
125252
0
0
MOV #14,4(R1)
ROMCLK
121111
MOV #21,4(R1)
MOV #121110,6(R1)
MOV #BIT9:BIT10,(R1)
BIS #BIT8,(R1)
NOP
MOV #121110,R5
ROMCLK
021044
ROMCLK
021065
MOV 4(R1),R4
CMP R5,R4
BEQ 3\$
HLT 12
SCOPE

:R1 CONTAINS BASE M8200-YC ADDRESS
:MASTER CLEAR M8200-YC
:USE SEL4 FOR ADDRESS
:USE SEL4 FOR ADDRESS
:LOAD BA AND DATA
:IN DATA
:OUT DATA
:IN BA
:OUT BA
:LOAD SEL 4 WITH OUT BA16 AND 17
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:SET OUTBA 16 AND 17
:LOAD SEL4
:PUT INSTRUCTION IN SEL6
:SET CROMI AND CROMO!!
:CLOCK IT!
:WAIT FOR NPR
:PUT "EXPECTED" IN R5
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:MOVE OUT DATA LB TO SEL4
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:MOVE OUT DATA HB TO SEL5
:PUT "FOUND" IN R4
:CORRECT RESULTS ?
:BR IF YES
:ERROR BA 16 AND 17 FAILED
:SCOPE THIS TEST

***** TEST 64 *****
*TEST OF EA BITS 16 AND 17
*DO A DATI USING IN BA BITS 16 AND 17
*VERIFY CORRECT RESULTS

TEST 64

3754 023136 012737 000064 001226 TST64:
3755 023144 012737 023262 001216
3756

MOV #64,TSTNO
MOV #TST65,NEXT

:R1 CONTAINS BASE M8200-YC ADDRESS

3757	023152	104412			MSTCLR		: MASTER CLEAR M8200-YC
3758	023154	013737	001412	023202	MOV	DMP04,1\$: USE SEL4 FOR ADDRESS
3759	023162	013737	001412	023200	MOV	DMP04,2\$: USE SEL4 FOR ADDRESS
3760	023170	004537	034674		JSR	RS,NPRSET	: LOAD BA AND DATA
3761	023174	000000			0		: IN DATA
3762	023176	125252			0	125252	: OUT DATA
3763	023200	000000			0		: IN BA
3764	023202	000000			0		: OUT BA
3765	023204	012761	000015	000004	MOV	#15,4(R1)	: LOAD SEL4
3766	023212	012761	121110	000006	MOV	#121110,6(R1)	: PUT INSTRUCTION IN SEL6
3767	023220	012711	003000		MOV	#BIT9!BIT10,(R1)	: SET CROMI AND CROMO!!
3768	023224	052711	000400		BIS	#BIT8,(R1)	: CLOCK IT!
3769	023230	000240			NOP		: WAIT FOR NPR
3770	023232	012705	121110		MOV	#121110,R5	: PUT "EXPECTED" IN R5
3771	023236	104414			ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3772	023240	021004			021004		: MOVE IN DATA LB TO SEL4
3773	023242	104414			ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3774	023244	021025			021025		: MOVE IN DATA HB TO SEL5
3775	023246	016104	000004		MOV	4(R1),R4	: PUT "FOUND" IN R4
3776	023252	020504			CMP	R5,R4	: CORRECT RESULTS ?
3777	023254	001401			BEQ	3\$: BR IF YES
3778	023256	104012			HLT	12	: ERROR BA 16 AND 17 FAILED
3779	023260	104400			3\$: SCOPE		: SCOPE THIS TEST

***** TEST 65 *****
 :#NPR NON-EXISTENT MEMORY TEST
 :#DO A DATO TO A NON-EXISTENT ADDRESS
 :#VERIFY THAT THE NON-EXISTENT BIT SET IN IBUS REG 11
 :*****

: TEST 65

3780							
3781							
3782							
3783							
3784							
3785							
3786							
3787							
3788							
3789							
3790	023262	012737	000065	001226	TST65:	MOV	#65,TSTNO
3791	023270	012737	023372	001216		MOV	#TST66,NEXT
3792							
3793	023276	104412			MSTCLR		: R1 CONTAINS BASE M8200-YC ADDRESS
3794	023300	004537	034674		JSR	RS,NPRSET	: MASTER CLEAR M8200-YC
3795	023304	000000			0		: LOAD IBUS REGISTERS 0-7
3796	023306	000000			0		: IN DATA
3797	023310	177320			0	177320	: OUT DATA
3798	023312	177320			0	177320	: IN BA
3799	023314	012761	000014	000004	MOV	#14,4(R1)	: OUT BA
3800	023322	104414			ROMCLK		: SET OUT BA BITS 16+17 IN PORT4
3801	023324	121111			121111		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3802	023326	012761	000021	000004	MOV	#21,4(R1)	: SET OUTBA 16 AND 17
3803	023334	104414			ROMCLK		: SET NPR REQUEST BITS IN PORT4
3804	023336	121110			121110		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3805	023340	000240			NOP		: MOV IBUS* 4 TO IBUS* 10
3806	023342	104414			ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3807	023344	121225			121225		: MOV IBUS*11 TO IBUS*5
3808	023346	012705	000001		MOV	#1,R5	: PUT "EXPECTED" IN R5
3809	023352	116104	000005		MOVB	5(R1),R4	: PUT "FOUND" IN R4
3810	023356	042704	177776		BIC	#177776,R4	: CLEAR UNWANTED BITS

```

3811 023362 020504      CMP      R5,R4      ;DATA CORRECT?
3812 023364 001401      BEQ      1$         ;BR IF YES
3813 023366 104012      HLT      12         ;ERROR NON-EXISTENT MEM BIT FAILED TO SET
3814 023370 104400      1$:      SCOPE      ;SCOPE THIS TEST
3815
3816
3817

```

```

;***** TEST 66 *****
;NPR NON-EXISTENT MEMORY TEST
;DO A DATI FROM A NON-EXISTENT ADDRESS
;VERIFY THAT THE NON-EXISTENT BIT SET IN IBUS REG 11
;*****

```

TEST 66

```

3824 ;-----
3825 023372 012737 000066 001226 TST66: MOV      #66,TSTNO
3826 023400 012737 023500 001216      MOV      #TST67,NEXT
3827 ;R1 CONTAINS BASE M8200-YC ADDRESS
3828 023406 104412      MSTCLR   ;MASTER CLEAR M8200-YC
3829 023410 004537 034674      JSR      R5,NPRSET ;LOAD IBUS REGISTERS 0-7
3830 023414 000000      0        ;IN DATA
3831 023416 000000      0        ;OUT DATA
3832 023420 177320      177320   ;IN BA
3833 023422 177320      177320   ;OUT BA
3834 023424 005061 000004      CLR      4(R1)
3835 023430 104414      ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3836 023432 121111      121111   ;CLEAR NON-EXISTENT BIT
3837 023434 012761 000015 000004      MOV      #15,4(R1) ;SET NPR REQUEST BITS IN PORT4
3838 023442 104414      ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3839 023444 121110      121110   ;MOV IBUS* 4 TO IBUS* 10
3840 023446 000240      NOP
3841 023450 104414      ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3842 023452 121225      121225   ;MOV IBUS*11 TO IBUS*5
3843 023454 012705 000001      MOV      #1,R5     ;PUT "EXPECTED" IN R5
3844 023460 116104 000005      MOVB    5(R1),R4   ;PUT "FOUND" IN R4
3845 023464 042704 177776      BIC     #17776,R4  ;CLEAR UNWANTED BITS
3846 023470 020504      CMP      R5,R4     ;DATA CORRECT?
3847 023472 001401      BEQ      1$         ;BR IF YES
3848 023474 104012      HLT      12         ;ERROR NON-EXISTENT MEM BIT FAILED TO SET
3849 023476 104400      1$:      SCOPE      ;SCOPE THIS TEST
3850
3851
3852

```

```

;***** TEST 67 *****
;NPR TEST
;USING DATO, NPR A BINARY COUNT (0-377 )
;FROM MICRO-PROCESSOR TO ALL AVAILABLE MEMORY
;*****

```

TEST 67

```

3859 ;-----
3860 023500 012737 000067 001226 TST67: MOV      #67,TSTNO
3861 023506 012737 000003 001222      MOV      #3,ICOUNT
3862 023514 012737 023676 001216      MOV      #TST70,NEXT
3863 ;R1 CONTAINS BASE M8200-YC ADDRESS
3864 023522 104412      MSTCLR   ;MASTER CLEAR M8200-YC

```

```

3865 023524 005037 023674          CLR      5$          ;START FLAG AT 0
3866 023530 005000                   CLR      RO          ;DATA
3867 023532 012702 037000          MOV      #CORMAX,R2 ;ADDRESS
3868 023536                   1$:      MOV      RO,2$ ;LOAD DATA
3869 023536 010037 023566                   MOV      R2,4$      ;LOAD BA
3870 023542 010237 023572                   BIT      #BIT0,R2   ;IS BA ODD?
3871 023546 032702 000001                   BEQ      .+6        ;BR IF NO
3872 023552 001402                   SWAB     2$         ;IF ODD PUT DATA IN HI-BYTE
3873 023554 000337 023566                   JSR      R5,NPRSET ;LOAD NPR REGISTERS
3874 023560 004537 034674                   0        ;IN DATA
3875 023564 000000                   2$:      0          ;OUT DATA
3876 023566 000000                   0        ;IN BA
3877 023570 000000                   4$:      0          ;OUT BA
3878 023572 000000                   CLRB     (R2)       ;CLEAR MEMORY LOCATION
3879 023574 105012                   MOV      #221,4(R1);LOAD PORT4
3880 023576 012761 000221 000004 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=53
3881 023604 104414                   121110 ;DO THE NPR
3882 023606 121110                   NOP
3883 023610 000240                   MOV      RO,R5     ;PUT "EXPECTED" IN R5
3884 023612 010005                   MOVB     (R2),R4   ;PUT "FOUND" IN R4
3885 023614 111204                   CMPB     R5,R4     ;IS DATA CORRECT?
3886 023616 120504                   BEQ      3$        ;BR IF YES
3887 023620 001401                   HLT      21        ;ERROR, DATA INCORRECT
3888 023622 104021                   3$:      SCOP1
3889 023624 104401                   INC      RO        ;NEXT CHARACTER
3890 023626 005200                   BIC      #177400,RO;USE ONLY LOW BYTE
3891 023630 042700 177400                   TST      5$        ;HAS MAX MEMORY BEEN REACHED YET?
3892 023634 005737 023674                   BEQ      6$        ;BR IF NO
3893 023640 001402                   TST      RO        ;DONE PATTERN?
3894 023642 005700                   BEQ      7$        ;BR IF YES
3895 023644 001412                   INC      R2        ;INC BA
3896 023646 005202                   CMP      MEMLIM,R2;REACHED MEMORY LIMIT YET?
3897 023650 023702 001304                   BNE     1$         ;BR IF NOT
3898 023654 001330                   MOV      #CORMAX,R2;RESTART BA AT FIRST ADDRESS
3899 023656 012702 037000                   MOV      #-1,5$   ;SET FLAG TO END TEST AT END OF DATA PATTERN
3900 023662 012737 177777 023674 BR 1$ ;CONTINUE
3901 023670 000722                   SCOPE
3902 023672 104400                   7$:      0        ;SCOPE THIS TEST
3903 023674 000000                   5$:      0        ;THIS LOCATION IS A FLAG, IT STARTS AT 0,
;AND IS SET TO -1 WHEN LAST MEMORY ADDRESS
;IS USED, TEST IS THEN ENDED WHEN PATTERN IS FINISHED

;***** TEST 70 *****
;MAIN MEMORY TEST
;FLOAT A 1 THROUGH ALL MAIN MEMORY LOCATIONS
;*****

; TEST 70
-----
3915 023676 012737 000070 001226 TST70: MOV      #70,TSTNO
3916 023704 012737 024024 001216 MOV      #TST71,NEXT
3917 023712 012737 023730 001220 MOV      #65$,LOCK
3918

;R1 CONTAINS BASE M8200-YC ADDRESS

```

3919	023720	104412				MSTCLR			:MASTER CLEAR M8200-YC
3920	023722	005002				CLR	R2		:START WITH ADDRESS 0
3921	023724	012700	000001		1\$:	MOV	#1,R0		:START WITH BIT 0
3922	023730	042737	000377	023744	65\$:	BIC	#377,66\$:CLEAR ADDRESS FIELD OF INSTRUCTION
3923	023736	050237	023744			BIS	R2,66\$:ADD ADDRESS TO INSTRUCTION
3924	023742	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3925	023744	010000			66\$:	010000			:LOAD MAR WITH ADDRESS IN R2
3926	023746	010061	000004			MOV	RO,4(R1)		:WRITE PATTERN IN PORT4
3927	023752	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3928	023754	122500				122500			:MOVE PORT4 TO MEMORY
3929	023756	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3930	023760	040620				040620			:MOVE MEMORY TO BR
3931	023762	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3932	023764	061225				61225			:MOVE BR TO PORT5
3933	023766	010005				MOV	RO,R5		:PUT "EXPECTED" IN R5
3934	023770	116104	000005			MOVB	5(R1),R4		:PUT "FOUND" IN R4
3935	023774	120504				CMPB	R5,R4		:DATA CORRECT?
3936	023776	001401				BEQ	67\$:BR IF YES
3937	024000	104013				HLT	13		:DATA ERROR
3938	024002	104401			67\$:	SCOPI			:SW09=1?
3939	024004	000241				CLC			:CLEAR CARRY
3940	024006	106100				ROLB	RO		:SHIFT BIT IN RO
3941	024010	001347				BNE	65\$:DONE IF RO=0
3942	024012	005202				INC	R2		:NEXT ADDRESS
3943	024014	022702	000400			CMP	#400,R2		:LAST ADDRESS
3944	024020	001341				BNE	1\$:BR IF NO
3945	024022	104400				SCOPE			:SCOPE THIS TEST
3946									
3947									
3948									
3949									
3950									
3951									
3952									
3953									
3954									
3955	024024	012737	000071	001226	TST71:	MOV	#71,TSTNO		
3956	024032	012737	024156	001216		MOV	#TST72,NEXT		
3957	024040	012737	024060	001220		MOV	#65\$,LOCK		
3958									
3959	024046	104412				MSTCLR			:R1 CONTAINS BASE M8200-YC ADDRESS
3960	024050	005002				CLR	R2		:MASTER CLEAR M8200-YC
3961	024052	012700	000001		1\$:	MOV	#1,R0		:START WITH ADDRESS 0
3962	024056	005100			64\$:	COM	RO		:START WITH BIT 0
3963	024060	042737	000377	024074	65\$:	BIC	#377,66\$:CHANGE TO FLOATING 0
3964	024066	050237	024074			BIS	R2,66\$:CLEAR ADDRESS FIELD OF INSTRUCTION
3965	024072	104414				ROMCLK			:ADD ADDRESS TO INSTRUCTION
3966	024074	010000			66\$:	010000			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3967	024076	010061	000004			MOV	RO,4(R1)		:LOAD MAR WITH ADDRESS IN R2
3968	024102	104414				ROMCLK			:WRITE PATTERN IN PORT4
3969	024104	122500				122500			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3970	024106	104414				ROMCLK			:MOVE PORT4 TO MEMORY
3971	024110	040620				040620			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3972	024112	104414				ROMCLK			:MOVE MEMORY TO BR
									:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```

:***** TEST 71 *****
:MAIN MEMORY TEST
:FLOAT A 0 THROUGH ALL MAIN MEMORY LOCATIONS
:*****

```

TEST 71

4027	024306	050237	024314		BIS	R2,5\$;ADD ADDRESS TO INSTRUCTION
4028	024312	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4029	024314	010000		5\$:	010000		;LOAD THE MAR
4030	024316	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4031	024320	040620			040620		;MOVE MEMORY TO THE BR
4032	024322	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4033	024324	061225			61225		;MOV BR TO PORT5
4034	024326	010205			MOV	R2,R5	;PUT "EXPECTED" IN R5
4035	024330	116104	000005		MOV	5(R1),R4	;PUT "FOUND" IN R4
4036	024334	120504			CMPB	R5,R4	;DATA CORRECT?
4037	024336	001401			BEQ	6\$;BR IF YES
4038	024340	104013			HLT	13	;ADDRESSING ERROR
4039	024342	104401		6\$:	SCOPI		;SW09=1?
4040	024344	005202			INC	R2	;NEXT ADDRESS
4041	024346	022702	000400		CMP	#400,R2	;IS IT THE LAST
4042	024352	001352			BNE	4\$;BR IF NO
4043	024354	104400			SCOPE		;SCOPE THIS TEST

4044
 4045
 4046
 4047
 4048
 4049
 4050
 4051
 4052
 4053

```

;***** TEST 73 *****
;MAR TEST
;PERFORM DUAL ADDRESSING TEST
;USING MAR AUTO-INC FEATURE
;*****
    
```

```

; TEST 73
;-----
4054 024356 012737 000073 001226 TST73: MOV #73,TSTNO
4055 024364 012737 024512 001216 MOV #TST74,NEXT
4056
4057 024372 104412 MSTCLR
4058 024374 005002 CLR R2
4059 024376 104414 ROMCLK
4060 024400 010000 010000 ;R1 CONTAINS BASE M8200-YC ADDRESS
4061 024402 032737 100000 001366 BIT #BIT15,STAT1 ;MASTER CLEAR M8200-YC
4062 024410 001402 .+6 ;START WITH A ZERO
4063 024412 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4064 024414 004000 4000 ;LOAD MAR
4065 024416 010261 000004 1$: MOV R2,4(R1) ;DMC?
4066 024422 104414 ROMCLK ;BR IF YES
4067 024424 136500 136500 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4068 024426 005202 INC R2 ;MAR HI + 0 (KMC ONLY)
4069 024430 022702 000400 CMP #400,R2 ;WRITE DATA TO PORT4
4070 024434 001370 BNE 1$ ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4071 024436 005002 CLR R2 ;LOAD MEM AUTO-INC MAR
4072 024440 104414 ROMCLK ;INCREMENT DATA
4073 024442 010000 010000 ;DONE YET?
4074 024444 032737 100000 001366 BIT #BIT15,STAT1 ;BR IF NO
4075 024452 001402 BEQ .+6 ;RESTART WITH A ZERO
4076 024454 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4077 024456 004000 4000 ;LOAD MAR
4078 024460 104414 ROMCLK ;MAR HI + 0 (KMC ONLY)
4079 024460 055224 2$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4080 024462 055224 ;MOVE MEM TO PORT4
    
```

```

4081 024464 010205           MOV     R2,R5           ;PUT "EXPECTED" IN R5
4082 024466 016104 000004   MOV     4(R1),R4       ;PUT "FOUND" IN R4
4083 024472 120504           CMPB   R5,R4          ;DATA CORRECT?
4084 024474 001401           BEQ    3$             ;BR IF YES
4085 024476 104014           HLT    14             ;MAR ERROR
4086 024500 005202           3$:   INC     R2         ;NEXT ADDRESS
4087 024502 022702 000400   CMP    #400,R2       ;DONE YET?
4088 024506 001364           BNE   2$             ;BR IF NO
4089 024510 104400           SCOPE                ;SCOPE THIS TEST
4090
4091
4092
4093
4094
4095
4096
4097
4098
4099 024512 012737 000074 001226 TST74: MOV     #74,TSTNO
4100 024520 012737 024624 001216   MOV     #TST75,NEXT
4101 024526 012737 024552 001220   MOV     #1$,LOCK
4102
4103 024534 104412           MSTCLR                ;R1 CONTAINS BASE M8200-YC ADDRESS
4104 024536 004737 034736   JSR    PC,MEMLD      ;MASTER CLEAR M8200-YC
4105 024542 024614           TDATA                ;LOAD MAINMEM DATA
4106 024544 004737 034772   JSR    PC,SPLD       ;POINTER TO DATA
4107 024550 024614           TDATA                ;LOAD SP DATA
4108
4109
4110
4111
4112
4113
4114
4115
4116
4117
4118
4119
4120
4121
4122
4123
4124
4125
4126
4127
4128
4129
4130
4131
4132
4133
4134

;***** TEST 74 *****
;*ALU C BIT TEST
;*TEST THAT AN ADD OF 377 AND 377 WILL SET THE C BIT
;*****

; TEST 74
-----
;
1$:   ROMCLK 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
      ROMCLK 054400!<0*20> ;MAR←0
      ROMCLK 040401!<1*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
      ROMCLK 61224 ;ADD 377 AND 377, TO SET C BIT
      MOV    #1,R5 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
      MOV    4(R1),R4 ;ADD 0 AND 0 AND THE C BIT
      CMPB  R5,R4 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
      BEQ   2$ ;PUT RESULTS IN PORT4
      HLT   15 ;PUT "EXPECTED" IN R5
2$:   SCOPE1 ;PUT "FOUND" IN R4
      SCOPE ;DATA CORRECT?
      .BYTE -1,0,0,0,0,0,0,0 ;BR IF YES
      ;ERROR C BIT NOT SET
      ;SW09=1?
      ;SCOPE THIS TEST

.EVEN

;***** TEST 75 *****
;*ALU TEST
;*TEST OF ALU FUNCTION SEL B WITH C BIT CLEARED
;*ALU FUNCTION (B) CODE=11
;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
  
```

```

4135 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4136 ;:*****
4137
4138 ; TEST 75
4139 -----
4140 024624 012737 000075 001226 TST75: MOV #75,TSTNO ;R1 CONTAINS BASE M8200-YC ADDRESS
4141 024632 012737 025000 001216 MOV #TST76,NEXT ;MASTER CLEAR M8200-YC
4142 024640 012737 024672 001220 MOV #1$,LOCK ;MEM + SP ADDRESS
4143 ;
4144 024646 104412 MSTCLR ;LOAD 8 WORDS OF MAIN MEMORY
4145 024650 005000 CLR R0 ;POINTER TO CORRECT DATA
4146 024652 012702 024770 MOV #5$,R2 ;LOAD 8 WORDS OF SP
4147 024656 004737 034736 JSR PC,MEMLD ;POINTER TO DATA
4148 024662 035062 MEMDAT ;LOAD 8 WORDS OF SP
4149 024664 004737 034772 JSR PC,SPLD ;POINTER TO DATA
4150 024670 035072 SPDAT ;CLEAR C BIT!
4151 024672 004737 035036 1$: JSR PC,CLRC ;CLEAR ADDRESS FIELD OF INSTRUCTION
4152 024676 042737 000017 024712 BIC #17,2$ ;ADD ADDRESS TO INSTRUCTION
4153 024704 050037 024712 BIS R0,2$ ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4154 024710 104414 ROMCLK ;LOAD MAR
4155 024712 010000 010000 2$: BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
4156 024714 042737 000017 024730 BIS R0,3$ ;ADD ADDRESS TO INSTRUCTION
4157 024722 050037 024730 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4158 024726 104414 040400! <11*20> 3$: BR + SEL 8
4159 024730 040620 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4160 024732 104414 61224 MOVB (R2),R5 ;MOVE BR TO PORT4
4161 024734 061224 MOVB 4(R1),R4 ;PUT "EXPECTED" IN R5
4162 024736 111205 000004 CMPB R5,R4 ;PUT "FOUND" IN R4
4163 024740 116104 BEQ 4$ ;DATA CORRECT?
4164 024744 120504 HLT 15 ;BR IF YES
4165 024746 001401 4$: SCOP1 ;ALU ERROR
4166 024750 104015 INC R2 ;SW09=1?
4167 024752 104401 INC R0 ;NEXT DATA
4168 024754 005202 INC R0 ;NEXT ADDRESS
4169 024756 005200 000010 CMP #10,R0 ;DONE YET?
4170 024760 022700 BNE 1$ ;BR IF NO
4171 024764 001342 SCOPE ;SCOPE THIS TEST
4172 024766 104400 .BYTE 0,-1,0,-1,125,252,125,252
4173 024770 000 377 000 5$:
4174 024773 377 125 252
4175 024776 125 252
4176 .EVEN
4177
4178 ;***** TEST 76 *****
4179 ;*ALU TEST
4180 ;*TEST OF ALU FUNCTION SEL A WITH C BIT CLEARED
4181 ;*ALU FUNCTION (A) CODE=10
4182 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4183 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4184 ;:*****
4185 ;
4186 ; TEST 76
4187 -----
4188

```

```

4189 025000 012737 000076 001226 TST76: MOV #76,ISTNO
4190 025006 012737 025154 001216 MOV #TST77,NEXT
4191 025014 012737 025046 001220 MOV #1$,LOCK
4192
4193 025022 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
4194 025024 005000 CLR ;MASTER CLEAR M8200-YC
4195 025026 012702 025144 CLR RO ;MEM + SP ADDRESS
4196 025032 004737 034736 MOV #5$,R2 ;POINTER TO CORRECT DATA
4197 025036 035062 JSR PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
4198 025040 004737 034772 MEMDAT ;POINTER TO DATA
4199 025044 035072 JSR PC,SPLD ;LOAD 8 WORDS OF SP
4200 025046 004737 035036 SPDAT ;POINTER TO DATA
4201 025052 042737 000017 025066 1$: JSR PC,CLRC ;CLEAR C BIT!
4202 025060 050037 025066 BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
4203 025064 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
4204 025066 010000 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4205 025070 042737 000017 025104 2$: BIC #17,3$ ;LOAD MAR
4206 025076 050037 025104 BIS RO,3$ ;CLEAR ADDRESS OF INSTRUCTION
4207 025102 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
4208 025104 040600 040400! <10*20> 3$: ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4209 025106 104414 ROMCLK ;BR + SEL A
4210 025110 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4211 025112 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
4212 025114 116104 000004 MOVB 4(R1),R4 ;PUT "EXPECTED" IN R5
4213 025120 120504 CMPB R5,R4 ;PUT "FOUND" IN R4
4214 025122 001401 BEQ 4$ ;DATA CORRECT?
4215 025124 104015 HLT 15 ;BR IF YES
4216 025126 104401 4$: SCOPE1 ;ALU ERROR
4217 025130 005202 INC R2 ;SW09=1?
4218 025132 005200 INC RO ;NEXT DATA
4219 025134 022700 000010 CMP #10,RO ;NEXT ADDRESS
4220 025140 001342 BNE 1$ ;DONE YET?
4221 025142 104400 SCOPE ;BR IF NO
4222 025144 000 000 377 5$: .BYTE 0,0,-1,-1,125,125,252,252 ;SCOPE THIS TEST
4223 025147 377 125
4224 025152 252 252
4225 .EVEN
4226
4227
4228 ;***** TEST 77 *****
4229 ;*ALU TEST
4230 ;*TEST OF ALU FUNCTION A OR NOTB WITH C BIT CLEARED
4231 ;*ALU FUNCTION (A OR NOTB) CODE=12
4232 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4233 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4234 ;*****
4235
4236 ; TEST 77
4237 ;-----
4238 025154 012737 000077 001226 TST77: MOV #77,TSTNO
4239 025162 012737 025330 001216 MOV #TST100,NEXT
4240 025170 012737 025222 001220 MOV #1$,LOCK
4241
4242 025176 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
;MASTER CLEAR M8200-YC

```

4243	025200	005000				CLR	RO	: MEM + SP ADDRESS
4244	025202	012702	025320			MOV	#5\$,R2	: POINTER TO CORRECT DATA
4245	025206	004737	034736			JSR	PC,MEMLD	: LOAD 8 WORDS OF MAIN MEMORY
4246	025212	035062				MEMDAT		: POINTER TO DATA
4247	025214	004737	034772			JSR	PC,SPLD	: LOAD 8 WORDS OF SP
4248	025220	035072				SPDAT		: POINTER TO DATA
4249	025222	004737	035036			JSR	PC,CLRC	: CLEAR C BIT!
4250	025226	042737	000017	025242	1\$:	BIC	#17,2\$: CLEAR ADDRESS FIELD OF INSTRUCTION
4251	025234	050037	025242			BIS	RO,2\$: ADD ADDRESS TO INSTRUCTION
4252	025240	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4253	025242	010000				010000		: LOAD MAR
4254	025244	042737	000017	025260	2\$:	BIC	#17,3\$: CLEAR ADDRESS OF INSTRUCTION
4255	025252	050037	025260			BIS	RO,3\$: ADD ADDRESS TO INSTRUCTION
4256	025256	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4257	025260	040640			3\$:	040400!	<12*20>	: BR + A OR NOTB
4258	025262	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4259	025264	061224				61224		: MOVE BR TO PORT4
4260	025266	111205				MOV#	(R2),R5	: PUT "EXPECTED" IN R5
4261	025270	116104	000004			MOV#	4(R1),R4	: PUT "FOUND" IN R4
4262	025274	120504				CMP#	R5,R4	: DATA CORRECT?
4263	025276	001401				BEQ	4\$: BR IF YES
4264	025300	104015				HLT	15	: ALU ERROR
4265	025302	104401			4\$:	SCOPI		: SW09=1?
4266	025304	005202				INC	R2	: NEXT DATA
4267	025306	005200				INC	RO	: NEXT ADDRESS
4268	025310	022700	000010			CMP	#10,RO	: DONE YET?
4269	025314	001342				BNE	1\$: BR IF NO
4270	025316	104400				SCOPE		: SCOPE THIS TEST
4271	025320	377	000	377	5\$:	.BYTE	-1,0,-1,-1,-1,125,252,-1	
4272	025323	377	377	125				
4273	025326	252	377					

.EVEN

```

***** TEST 100 *****
: *ALU TEST
: *TEST OF ALU FUNCTION A AND B WITH C BIT CLEARED
: *ALU FUNCTION (A AND B) CODE=13
: *LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
: *PERFORM THE FUNCTION, VERIFY THE RESULTS
: *****

```

: TEST 100

4287	025330	012737	000100	001226	TST100:	MOV	#100,TSTNO	
4288	025336	012737	025504	001216		MOV	#TST101,NEXT	
4289	025344	012737	025376	001220		MOV	#1\$,LOCK	
4290								: R1 CONTAINS BASE M8200-YC ADDRESS
4291	025352	104412				MSTCLR		: MASTER CLEAR M8200-YC
4292	025354	005000				CLR	RO	: MEM + SP ADDRESS
4293	025356	012702	025474			MOV	#5\$,R2	: POINTER TO CORRECT DATA
4294	025362	004737	034736			JSR	PC,MEMLD	: LOAD 8 WORDS OF MAIN MEMORY
4295	025366	035062				MEMDAT		: POINTER TO DATA
4296	025370	004737	034772			JSR	PC,SPLD	: LOAD 8 WORDS OF SP

4297	025374	035072				SPDAT		: POINTER TO DATA
4298	025376	004737	035036		1\$:	JSR	PC, CLRC	: CLEAR C BIT!
4299	025402	042737	000017	025416		BIC	#17, 2\$: CLEAR ADDRESS FIELD OF INSTRUCTION
4300	025410	050037	025416			BIS	RO, 2\$: ADD ADDRESS TO INSTRUCTION
4301	025414	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4302	025416	010000			2\$:	010000		: LOAD MAR
4303	025420	042737	000017	025434		BIC	#17, 3\$: CLEAR ADDRESS OF INSTRUCTION
4304	025426	050037	025434			BIS	RO, 3\$: ADD ADDRESS TO INSTRUCTION
4305	025432	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4306	025434	040660			3\$:	040400! <13*20>		: BR + A AND B
4307	025436	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4308	025440	061224				61224		: MOVE BR TO PORT4
4309	025442	111205				MOV B	(R2), R5	: PUT "EXPECTED" IN R5
4310	025444	116104	000004			MOV B	4(R1), R4	: PUT "FOUND" IN R4
4311	025450	120504				CMP B	R5, R4	: DATA CORRECT?
4312	025452	001401				BEQ	4\$: BR IF YES
4313	025454	104015				HLT	1\$: ALU ERROR
4314	025456	104401			4\$:	SCOPI		: SW09=1?
4315	025460	005202				INC	R2	: NEXT DATA
4316	025462	005200				INC	RO	: NEXT ADDRESS
4317	025464	022700	000010			CMP	#10, RO	: DONE YET?
4318	025470	001342				BNE	1\$: BR IF NO
4319	025472	104400				SCOPE		: SCOPE THIS TEST
4320	025474	000	000	000	5\$:	.BYTE	0,0,0,-1,125,0,0,252	
4321	025477	377	125	000				
4322	025502	000	252					
4323								.EVEN
4324								
4325								
4326								: ***** TEST 101 *****
4327								: *ALU TEST
4328								: *TEST OF ALU FUNCTION A OR B WITH C BIT CLEARED
4329								: *ALU FUNCTION (A OR B) CODE=14
4330								: *LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4331								: *PERFORM THE FUNCTION, VERIFY THE RESULTS
4332								: *****
4333								
4334								: TEST 101
4335								
4336	025504	012737	000101	001226	TST101:	MOV	#101, TSTNO	
4337	025512	012737	025660	001216		MOV	#TST102, NEXT	
4338	025520	012737	025552	001220		MOV	#1\$, LOCK	
4339								: R1 CONTAINS BASE MB200-YC ADDRESS
4340	025526	104412				MSTCLR		: MASTER CLEAR MB200-YC
4341	025530	005000				CLR	RO	: MEM + SP ADDRESS
4342	025532	012702	025650			MOV	#5\$, R2	: POINTER TO CORRECT DATA
4343	025536	004737	034736			JSR	PC, MEMLD	: LOAD 8 WORDS OF MAIN MEMORY
4344	025542	035062				MEMDAT		: POINTER TO DATA
4345	025544	004737	034772			JSR	PC, SPLD	: LOAD 8 WORDS OF SP
4346	025550	035072				SPDAT		: POINTER TO DATA
4347	025552	004737	035036		1\$:	JSR	PC, CLRC	: CLEAR C BIT!
4348	025556	042737	000017	025572		BIC	#17, 2\$: CLEAR ADDRESS FIELD OF INSTRUCTION
4349	025564	050037	025572			BIS	RO, 2\$: ADD ADDRESS TO INSTRUCTION
4350	025570	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

4405	025766	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4406	025770	061224				61224			:MOVE BR TO PORT4
4407	025772	111205				MOVB	(R2),R5		:PUT "EXPECTED" IN R5
4408	025774	116104	000004			MOVB	4(R1),R4		:PUT "FOUND" IN R4
4409	026000	120504				CMPB	R5,R4		:DATA CORRECT?
4410	026002	001401				BEQ	4\$:BR IF YES
4411	026004	104015				HLT	1\$:ALU ERROR
4412	026006	104401			4\$:	SCOPI			:SW09=1?
4413	026010	005202				INC	R2		:NEXT DATA
4414	026012	005200				INC	R0		:NEXT ADDRESS
4415	026014	022700	000010			CMP	#10,R0		:DONE YET?
4416	026020	001342				BNE	1\$:BR IF NO
4417	026022	104400				SCOPE			:SCOPE THIS TEST
4418	026024	000	377	377	5\$:	.BYTE	0,-1,-1,0,0,-1,-1,0		
4419	026027	000	000	377					
4420	026032	377	000						

.EVEN

```

:***** TEST 103 *****
:*ALU TEST
:*TEST OF ALU FUNCTION ADD WITH C BIT CLEARED
:*ALU FUNCTION (A PLUS B) CODE=00
:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
:*PERFORM THE FUNCTION, VERIFY THE RESULTS
:*****
    
```

: TEST 103

4434	026034	012737	000103	001226	TST103:	MOV	#103,TSTNO		
4435	026042	012737	026210	001216		MOV	#TST104,NEXT		
4436	026050	012737	026102	001220		MOV	#1\$,LOCK		
4437									:R1 CONTAINS BASE M8200-YC ADDRESS
4438	026056	104412				MSTCLR			:MASTER CLEAR M8200-YC
4439	026060	005000				CLR	R0		:MEM + SP ADDRESS
4440	026062	012702	026200			MOV	#5\$,R2		:POINTER TO CORRECT DATA
4441	026066	004737	034736			JSR	PC,MEMLD		:LOAD 8 WORDS OF MAIN MEMORY
4442	026072	035062				MEMDAT			:POINTER TO DATA
4443	026074	004737	034772			JSR	PC,SPLD		:LOAD 8 WORDS OF SP
4444	026100	035072				SPDAT			:POINTER TO DATA
4445	026102	004737	035036		1\$:	JSR	PC,CLRC		:CLEAR C BIT!
4446	026106	042737	000017	026122		BIC	#17,2\$:CLEAR ADDRESS FIELD OF INSTRUCTION
4447	026114	050037	026122			BIS	R0,2\$:ADD ADDRESS TO INSTRUCTION
4448	026120	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4449	026122	010000			2\$:	010000			:LOAD MAR
4450	026124	042737	000017	026140		BIC	#17,3\$:CLEAR ADDRESS OF INSTRUCTION
4451	026132	050037	026140			BIS	R0,3\$:ADD ADDRESS TO INSTRUCTION
4452	026136	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4453	026140	040400			3\$:	040400!	<00*20>		:BR + ADD
4454	026142	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4455	026144	061224				61224			:MOVE BR TO PORT4
4456	026146	111205				MOVB	(R2),R5		:PUT "EXPECTED" IN R5
4457	026150	116104	000004			MOVB	4(R1),R4		:PUT "FOUND" IN R4
4458	026154	120504				CMPB	R5,R4		:DATA CORRECT?

```

4459 026156 001401      BEQ      4$      ;BR IF YES
4460 026160 104015      HLT      15      ;ALU ERROR
4461 026162 104401      4$: SCOP1      ;SW09=1?
4462 026164 005202      INC      R2      ;NEXT DATA
4463 026166 005200      INC      R0      ;NEXT ADDRESS
4464 026170 022700 000010  CMP      #10,R0  ;DONE YET?
4465 026174 001342      BNE      1$      ;BR IF NO
4466 026176 104400      SCOPE      ;SCOPE THIS TEST
4467 026200      000      377      377 5$: .BYTE 0,-1,-1,376,252,-1,-1,124
4468 026203      376      377
4469 026206      377      124
4470      .EVEN
    
```

```

***** TEST 104 *****
*ALU TEST
*TEST OF ALU FUNCTION 2A W/C WITH C BIT CLEARED
*ALU FUNCTION (A PLUS A PLUS C) CODE=6
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS
*****
    
```

TEST 104

```

4483 026210 012737 000104 001226 TST104: MOV      #104,TSTNO
4484 026216 012737 026364 001216      MOV      #TST105,NEXT
4485 026224 012737 026256 001220      MOV      #1$,LOCK
4486
4487 026232 104412      MSTCLR      ;R1 CONTAINS BASE M8200-YC ADDRESS
4488 026234 005000      CLR      RO      ;MASTER CLEAR M8200-YC
4489 026236 012702 026354      MOV      #5$,R2   ;MEM + SP ADDRESS
4490 026242 004737 034736      JSR      PC,MEMLD ;POINTER TO CORRECT DATA
4491 026246 035062      MEMDAT      ;LOAD 8 WORDS OF MAIN MEMORY
4492 026250 004737 034772      JSR      PC,SPLD  ;POINTER TO DATA
4493 026254 035072      SPDAT      ;LOAD 8 WORDS OF SP
4494 026256 004737 035036      JSR      PC,CLRC  ;POINTER TO DATA
4495 026262 042737 000017 026276 1$: BIC      #17,2$  ;CLEAR C BIT!
4496 026270 050037 026276      BIS      RO,2$   ;CLEAR ADDRESS FIELD OF INSTRUCTION
4497 026274 104414      ROMCLK      ;ADD ADDRESS TO INSTRUCTION
4498 026276 010000      010000      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4499 026300 042737 000017 026314 2$: BIC      #17,3$  ;LOAD MAR
4500 026306 050037 026314      BIS      RO,3$   ;CLEAR ADDRESS OF INSTRUCTION
4501 026312 104414      ROMCLK      ;ADD ADDRESS TO INSTRUCTION
4502 026314 040540      040400! <6*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4503 026316 104414      ROMCLK      ;BR + 2A W/C
4504 026320 061224      61224      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4505 026322 111205      MOVB      (R2),R5 ;MOVE BR TO PORT4
4506 026324 116104 000004      MOVB      4(R1),R4 ;PUT "EXPECTED" IN R5
4507 026330 120504      CMPB      R5,R4  ;PUT "FOUND" IN R4
4508 026332 001401      BEQ      4$      ;DATA CORRECT?
4509 026334 104015      HLT      15      ;BR IF YES
4510 026336 104401      4$: SCOP1      ;ALU ERROR
4511 026340 005202      INC      R2      ;SW09=1?
4512 026342 005200      INC      RO      ;NEXT DATA
    
```

4513 026344 022700 000010
 4514 026350 001342
 4515 026352 104400
 4516 026354 000 000 376 5\$:
 4517 026357 376 252 252
 4518 026362 124 124
 4519
 4520
 4521
 4522
 4523
 4524
 4525
 4526
 4527
 4528
 4529
 4530
 4531

CMP #10,R0 ;DONE YET?
 BNE 1\$;BR IF NO
 SCOPE ;SCOPE THIS TEST
 .BYTE 0,0,376,376,252,252,124,124
 .EVEN

***** TEST 105 *****
 :ALU TEST
 :TEST OF ALU FUNCTION SUB WITH C BIT CLEARED
 :ALU FUNCTION (A-B) CODE=16
 :LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
 :PERFORM THE FUNCTION, VERIFY THE RESULTS
 :*****

TEST 105

4532 026364 012737 000105 001226 TST105:
 4533 026372 012737 026540 001216
 4534 026400 012737 026432 001220
 4535
 4536 026406 104412
 4537 026410 005000
 4538 026412 012702 026530
 4539 026416 004737 034736
 4540 026422 035062
 4541 026424 004737 034772
 4542 026430 035072
 4543 026432 004737 035036 1\$:
 4544 026436 042737 000017 026452
 4545 026444 050037 026452
 4546 026450 104414
 4547 026452 010000 2\$:
 4548 026454 042737 000017 026470
 4549 026462 050037 026470
 4550 026466 104414
 4551 026470 040740 3\$:
 4552 026472 104414
 4553 026474 061224
 4554 026476 111205
 4555 026500 116104 000004
 4556 026504 120504
 4557 026506 001401
 4558 026510 104015
 4559 026512 104401 4\$:
 4560 026514 005202
 4561 026516 005200
 4562 026520 022700 000010
 4563 026524 001342
 4564 026526 104400
 4565 026530 000 001 377 5\$:
 4566 026533 000 000 253

MOV #105,TSTNO
 MOV #TST106,NEXT
 MOV #1\$,LOCK
 MSTCLR
 CLR RO
 MOV #5\$,R2
 JSR PC,MEMLD
 MEMDAT
 JSR PC,SPLD
 SPDAT
 JSR PC,CLRC
 BIC #17,2\$
 BIS RO,2\$
 ROMCLK
 010000
 BIC #17,3\$
 BIS RO,3\$
 ROMCLK
 040400! <16*20>
 ROMCLK
 61224
 MOVB (R2),R5
 MOVB 4(R1),R4
 CMPB R5,R4
 BEQ 4\$
 HLT 15
 SCOPE1
 INC R2
 INC RO
 CMP #10,R0
 BNE 1\$
 SCOPE
 .BYTE 0,1,-1,0,0,253,125,0
 ;R1 CONTAINS BASE M8200-YC ADDRESS
 ;MASTER CLEAR M8200-YC
 ;MEM + SP ADDRESS
 ;POINTER TO CORRECT DATA
 ;LOAD 8 WORDS OF MAIN MEMORY
 ;POINTER TO DATA
 ;LOAD 8 WORDS OF SP
 ;POINTER TO DATA
 ;CLEAR C BIT!
 ;CLEAR ADDRESS FIELD OF INSTRUCTION
 ;ADD ADDRESS TO INSTRUCTION
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;LOAD MAR
 ;CLEAR ADDRESS OF INSTRUCTION
 ;ADD ADDRESS TO INSTRUCTION
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;BR + SUB
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;MOVE BR TO PORT4
 ;PUT "EXPECTED" IN R5
 ;PUT "FOUND" IN R4
 ;DATA CORRECT?
 ;BR IF YES
 ;ALU ERROR
 ;SW09=1?
 ;NEXT DATA
 ;NEXT ADDRESS
 ;DONE YET?
 ;BR IF NO
 ;SCOPE THIS TEST

```

4567 026536 125 000
4568
4569 .EVEN
4570
4571
4572
4573
4574
4575
4576
4577
4578
4579
4580
4581 026540 012737 000106 001226 TST106:
4582 026546 012737 026714 001216
4583 026554 012737 026606 001220
4584
4585 026562 104412
4586 026564 005000
4587 026566 012702 026704
4588 026572 004737 034736
4589 026576 035062
4590 026600 004737 034772
4591 026604 035072
4592 026606 004737 035036 1$:
4593 026612 042737 000017 026626
4594 026620 050037 026626
4595 026624 104414
4596 026626 010000 2$:
4597 026630 042737 000017 026644
4598 026636 050037 026644
4599 026642 104414
4600 026644 040420 3$:
4601 026646 104414
4602 026650 061224
4603 026652 111205
4604 026654 116104 000004
4605 026660 120504
4606 026662 001401
4607 026664 104015
4608 026666 104401 4$:
4609 026670 005202
4610 026672 005200
4611 026674 022700 000010
4612 026700 001342
4613 026702 104400
4614 026704 000 377 377 5$:
4615 026707 376 252 377
4616 026712 377 124
4617 .EVEN
4618
4619
4620

```

```

***** TEST 106 *****
*ALU TEST
*TEST OF ALU FUNCTION ADD W/C WITH C BIT CLEARED
*ALU FUNCTION (A PLUS B PLUS C) CODE=01
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS
*****

```

TEST 106

```

-----
MOV #106,TSTNO
MOV #TST107,NEXT
MOV #1$,LOCK
MSTCLR
CLR RO
MOV #5$,R2
JSR PC,MEMLD
MEMDAT
JSR PC,SPLD
SPDAT
JSR PC,CLRC
BIC #17,2$
BIS RO,2$
ROMCLK
010000
BIC #17,3$
BIS RO,3$
ROMCLK
040400! <01*20>
MOV (R2),R5
MOV 4(R1),R4
CMPB R5,R4
BEQ 4$
HLT 15
SCOPI
INC R2
INC RO
CMP #10,RO
BNE 1$
SCOPE
.BYTE 0,-1,-1,376,252,-1,-1,124

```

R1 CONTAINS BASE M8200-YC ADDRESS
MASTER CLEAR M8200-YC
MEM + SP ADDRESS
POINTER TO CORRECT DATA
LOAD 8 WORDS OF MAIN MEMORY
POINTER TO DATA
LOAD 8 WORDS OF SP
POINTER TO DATA
CLEAR C BIT!
CLEAR ADDRESS FIELD OF INSTRUCTION
ADD ADDRESS TO INSTRUCTION
NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
LOAD MAR
CLEAR ADDRESS OF INSTRUCTION
ADD ADDRESS TO INSTRUCTION
NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
BR + ADD W/C
NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
MOVE BR TO PORT4
PUT "EXPECTED" IN R5
PUT "FOUND" IN R4
DATA CORRECT?
BR IF YES
ALU ERROR
SW09=1?
NEXT DATA
NEXT ADDRESS
DONE YET?
BR IF NO
SCOPE THIS TEST

***** TEST 107 *****

```

4621 : *ALU TEST
4622 : *TEST OF ALU FUNCTION SUB W/C WITH C BIT CLEARED
4623 : *ALU FUNCTION (A-B-C) CODE=2
4624 : *LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4625 : *PERFORM THE FUNCTION, VERIFY THE RESULTS
4626 : *****

```

```

4627 : TEST 107
4628 : -----
4629 :
4630 026714 012737 000107 001226 TST107: MOV #107,TSTNO
4631 026722 012737 027070 001216 MOV #TST110,NEXT
4632 026730 012737 026762 001220 MOV #1$,LOCK
4633 :
4634 026736 104412 MSTCLR : R1 CONTAINS BASE M8200-YC ADDRESS
4635 026740 005000 CLR RO : MASTER CLEAR M8200-YC
4636 026742 012702 027060 MOV #5$,R2 : MEM + SP ADDRESS
4637 026746 004737 034736 JSR PC,MEMLD : POINTER TO CORRECT DATA
4638 026752 035062 MEMDAT : LOAD 8 WORDS OF MAIN MEMORY
4639 026754 004737 034772 JSR PC,SPLD : POINTER TO DATA
4640 026760 035072 SPDAT : LOAD 8 WORDS OF SP
4641 026762 004737 035036 1$: JSR PC,CLRC : POINTER TO DATA
4642 026766 042737 000017 027002 BIC #17,2$ : CLEAR C BIT!
4643 026774 050037 027002 BIS RO,2$ : CLEAR ADDRESS FIELD OF INSTRUCTION
4644 027000 104414 ROMCLK : ADD ADDRESS TO INSTRUCTION
4645 027002 010000 2$: 010000 : NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4646 027004 042737 000017 027020 BIC #17,3$ : LOAD MAR
4647 027012 050037 027020 BIS RO,3$ : CLEAR ADDRESS OF INSTRUCTION
4648 027016 104414 ROMCLK : ADD ADDRESS TO INSTRUCTION
4649 027020 040400! <2*20> 3$: 040400! <2*20> : NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4650 027022 104414 ROMCLK : BR + SUB W/C
4651 027024 061224 61224 : MOVE BR TO PORT4
4652 027026 111205 MOVB (R2),R5 : PUT "EXPECTED" IN R5
4653 027030 116104 000004 MOVB 4(R1),R4 : PUT "FOUND" IN R4
4654 027034 120504 CMPB R5,R4 : DATA CORRECT?
4655 027036 001401 BEQ 4$ : BR IF YES
4656 027040 104015 HLT 1$ : ALU ERROR
4657 027042 104401 4$: SCOPE1 : SW09=1?
4658 027044 005202 INC R2 : NEXT DATA
4659 027046 005200 INC RO : NEXT ADDRESS
4660 027050 022700 000010 CMP #10,RO : DONE YET?
4661 027054 001342 BNE 1$ : BR IF NO
4662 027056 104400 SCOPE : SCOPE THIS TEST
4663 027060 377 000 376 5$: .BYTE -1,0,376,-1,-1,252,124,-1
4664 027063 377 252
4665 027066 124 377
4666 :
4667 : .EVEN
4668 :
4669 : ***** TEST 110 *****
4670 : *ALU TEST
4671 : *TEST OF ALU FUNCTION INC A WITH C BIT CLEARED
4672 : *ALU FUNCTION (A PLUS 1) CODE=3
4673 : *LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4674 : *PERFORM THE FUNCTION, VERIFY THE RESULTS

```

```

4675 ;:*****
4676 ;
4677 ; TEST 110
4678 ;-----
4679 027070 012737 000110 001226 TST110: MOV #110,TSTNO
4680 027076 012737 027244 001216 MOV #TST111,NEXT
4681 027104 012737 027136 001220 MOV #1$,LOCK
4682 ;
4683 027112 104412 MSTCLR ; R1 CONTAINS BASE M8200-YC ADDRESS
4684 027114 005000 CLR R0 ; MASTER CLEAR M8200-YC
4685 027116 012702 027234 MOV #5$,R2 ; MEM + SP ADDRESS
4686 027122 004737 034736 JSR PC,MEMLD ; POINTER TO CORRECT DATA
4687 027126 035062 MEMDAT ; LOAD 8 WORDS OF MAIN MEMORY
4688 027130 004737 034772 JSR PC,SPLD ; POINTER TO DATA
4689 027134 035072 SPDAT ; LOAD 8 WORDS OF SP
4690 027136 004737 035036 1$: JSR PC,CLRC ; POINTER TO DATA
4691 027142 042737 000017 027156 BIC #17,2$ ; CLEAR C BIT!
4692 027150 050037 027156 BIS R0,2$ ; CLEAR ADDRESS FIELD OF INSTRUCTION
4693 027154 104414 ROMCLK ; ADD ADDRESS TO INSTRUCTION
4694 027156 010000 010000 2$: ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4695 027160 042737 000017 027174 BIC #17,3$ ; LOAD MAR
4696 027166 050037 027174 BIS R0,3$ ; CLEAR ADDRESS OF INSTRUCTION
4697 027172 104414 ROMCLK ; ADD ADDRESS TO INSTRUCTION
4698 027174 040460 040400! <3*20> 3$: ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4699 027176 104414 ROMCLK ; BR + INC A
4700 027200 061224 61224 ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4701 027202 111205 MOVB (R2),R5 ; MOVE BR TO PORT4
4702 027204 116104 000004 MOVB 4(R1),R4 ; PUT "EXPECTED" IN R5
4703 027210 120504 CMPB R5,R4 ; PUT "FOUND" IN R4
4704 027212 001401 BEQ 4$ ; DATA CORRECT?
4705 027214 104015 HLT 1$ ; BR IF YES
4706 027216 104401 4$: SCOP1 ; ALU ERROR
4707 027220 005202 INC R2 ; SW09=1?
4708 027222 005200 INC R0 ; NEXT DATA
4709 027224 022700 000010 CMP #10,R0 ; NEXT ADDRESS
4710 027230 001342 BNE 1$ ; DONE YET?
4711 027232 104400 SCOPE ; BR IF NO
4712 027234 001 001 000 5$: .BYTE 1,1,0,0,126,126,253,253 ; SCOPE THIS TEST
4713 027237 000 126 126
4714 027242 253 253
4715 .EVEN
4716
4717
4718 ;***** TEST 111 *****
4719 ;*ALU TEST
4720 ;*TEST OF ALU FUNCTION 2A WITH C BIT CLEARED
4721 ;*ALU FUNCTION (A PLUS A) CODE=5
4722 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4723 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4724 ;:*****
4725 ;
4726 ; TEST 111
4727 ;-----
4728 027244 012737 000111 001226 TST111: MOV #111,TSTNO

```

4729	027252	012737	027420	001216		MOV	#TST112,NEXT	
4730	027260	012737	027312	001220		MOV	#1\$,LOCK	
4731								:R1 CONTAINS BASE M8200-YC ADDRESS
4732	027266	104412				MSTCLR		:MASTER CLEAR M8200-YC
4733	027270	005000				CLR	RO	:MEM + SP ADDRESS
4734	027272	012702	027410			MOV	#5\$,R2	:POINTER TO CORRECT DATA
4735	027276	004737	034736			JSR	PC,MEMLD	:LOAD 8 WORDS OF MAIN MEMORY
4736	027302	035062				MEMDAT		:POINTER TO DATA
4737	027304	004737	034772			JSR	PC,SPLD	:LOAD 8 WORDS OF SP
4738	027310	035072				SPDAT		:POINTER TO DATA
4739	027312	004737	035036		1\$:	JSR	PC,CLRC	:CLEAR C BIT!
4740	027316	042737	000017	027332		BIC	#17,2\$:CLEAR ADDRESS FIELD OF INSTRUCTION
4741	027324	050037	027332			BIS	RO,2\$:ADD ADDRESS TO INSTRUCTION
4742	027330	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4743	027332	010000			2\$:	010000		:LOAD MAR
4744	027334	042737	000017	027350		BIC	#17,3\$:CLEAR ADDRESS OF INSTRUCTION
4745	027342	050037	027350			BIS	RO,3\$:ADD ADDRESS TO INSTRUCTION
4746	027346	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4747	027350	040520			3\$:	040400! <5*20>		:BR + 2A
4748	027352	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4749	027354	061224				61224		:MOVE BR TO PORT4
4750	027356	111205				MOVB	(R2),R5	:PUT "EXPECTED" IN R5
4751	027360	116104	000004			MOVB	4(R1),R4	:PUT "FOUND" IN R4
4752	027364	120504				CMPB	R5,R4	:DATA CORRECT?
4753	027366	001401				BEQ	4\$:BR IF YES
4754	027370	104015				HLT	15	:ALU ERROR
4755	027372	104401			4\$:	SCOPI		:SW09=1?
4756	027374	005202				INC	R2	:NEXT DATA
4757	027376	005200				INC	RO	:NEXT ADDRESS
4758	027400	022700	000010			CMP	#10,RO	:DONE YET?
4759	027404	001342				BNE	1\$:BR IF NO
4760	027406	104400				SCOPE		:SCOPE THIS TEST
4761	027410	000	000	376	5\$:	.BYTE	0,0,376,376,252,252,124,124	
4762	027413	376	252	252				
4763	027416	124	124					
4764								.EVEN
4765								
4766								
4767								:***** TEST 112 *****
4768								:*ALU TEST
4769								:*TEST OF ALU FUNCTION A PLUS C WITH C BIT CLEARED
4770								:*ALU FUNCTION (A PLUS C) CODE=4
4771								:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4772								:*PERFORM THE FUNCTION, VERIFY THE RESULTS
4773								:*****
4774								
4775								: TEST 112
4776								
4777	027420	012737	000112	001226		TST112:	MOV	#112,TSTNO
4778	027426	012737	027574	001216			MOV	#TST113,NEXT
4779	027434	012737	027466	001220			MOV	#1\$,LOCK
4780								:R1 CONTAINS BASE M8200-YC ADDRESS
4781	027442	104412				MSTCLR		:MASTER CLEAR M8200-YC
4782	027444	005000				CLR	RO	:MEM + SP ADDRESS

4783	027446	012702	027564			MOV	#5\$,R2		: POINTER TO CORRECT DATA
4784	027452	004737	034736			JSR	PC,MEMLD		: LOAD 8 WORDS OF MAIN MEMORY
4785	027456	035062				MEMDAT			: POINTER TO DATA
4786	027460	004737	034772			JSR	PC,SPLD		: LOAD 8 WORDS OF SP
4787	027464	035072				SPDAT			: POINTER TO DATA
4788	027466	004737	035036		1\$:	JSR	PC,CLRC		: CLEAR C BIT!
4789	027472	042737	000017	027506		BIC	#17,2\$: CLEAR ADDRESS FIELD OF INSTRUCTION
4790	027500	050037	027506			BIS	RO,2\$: ADD ADDRESS TO INSTRUCTION
4791	027504	104414				ROMCLK			: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4792	027506	010000			2\$:	010000			: LOAD MAR
4793	027510	042737	000017	027524		BIC	#17,3\$: CLEAR ADDRESS OF INSTRUCTION
4794	027516	050037	027524			BIS	RO,3\$: ADD ADDRESS TO INSTRUCTION
4795	027522	104414				ROMCLK			: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4796	027524	040500			3\$:	040400! <4*20>			: BR + A PLUS C
4797	027526	104414				ROMCLK			: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4798	027530	061224				61224			: MOVE BR TO PORT4
4799	027532	111205				MOV#	(R2),R5		: PUT "EXPECTED" IN R5
4800	027534	116104	000004			MOV#	4(R1),R4		: PUT "FOUND" IN R4
4801	027540	120504				CMP#	R5,R4		: DATA CORRECT?
4802	027542	001401				BEQ	4\$: BR IF YES
4803	027544	104015				HLT	1\$: ALU ERROR
4804	027546	104401			4\$:	SCOPE1			: SMO9=1?
4805	027550	005202				INC	R2		: NEXT DATA
4806	027552	005200				INC	RO		: NEXT ADDRESS
4807	027554	022700	000010			CMP	#10,RO		: DONE YET?
4808	027560	001342				BNE	1\$: BR IF NO
4809	027562	104400				SCOPE			: SCOPE THIS TEST
4810	027564	000	000	377	5\$:	.BYTE	0,0,-1,-1,125,125,252,252		
4811	027567	377	125	125					
4812	027572	252	252						

.EVEN

***** TEST 113 *****
 : *ALU TEST
 : *TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT CLEARED
 : *ALU FUNCTION (A-B-1) CODE=17
 : *LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
 : *PERFORM THE FUNCTION, VERIFY THE RESULTS
 : *****

: TEST 113

4825									
4826	027574	012737	000113	001226	TST113:	MOV	#113,TSTNO		
4827	027602	012737	027750	001216		MOV	#TST114,NEXT		
4828	027610	012737	027642	001220		MOV	#1\$,LOCK		
4829									: R1 CONTAINS BASE M8200-YC ADDRESS
4830	027616	104412				MSTCLR			: MASTER CLEAR M8200-YC
4831	027620	005000				CLR	RO		: MEM + SP ADDRESS
4832	027622	012702	027740			MOV	#5\$,R2		: POINTER TO CORRECT DATA
4833	027626	004737	034736			JSR	PC,MEMLD		: LOAD 8 WORDS OF MAIN MEMORY
4834	027632	035062				MEMDAT			: POINTER TO DATA
4835	027634	004737	034772			JSR	PC,SPLD		: LOAD 8 WORDS OF SP
4836	027640	035072				SPDAT			: POINTER TO DATA

```

4837 027642 004737 035036 1$: JSR PC,CLRC ;CLEAR C BIT!
4838 027646 042737 000017 027662 BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
4839 027654 050037 027662 BIS RO,2$ ;ADD ADDRESS TO INSTRUCTION
4840 027660 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4841 027662 010000 010000 ;LOAD MAR
4842 027664 042737 000017 027700 BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
4843 027672 050037 027700 BIS RO,3$ ;ADD ADDRESS TO INSTRUCTION
4844 027676 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4845 027700 040760 040400! <17*20> ;BR + 2'S COMP SUB
4846 027702 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4847 027704 061224 61224 ;MOVE BR TO PORT4
4848 027706 111205 MOVB (R2),R5 ;PUT "EXPECTED" IN R5
4849 027710 116104 000004 MOVB 4(R1),R4 ;PUT "FOUND" IN R4
4850 027714 120504 CMPB R5,R4 ;DATA CORRECT?
4851 027716 001401 4$ BEQ 4$ ;BR IF YES
4852 027720 104015 HLT 15 ;ALU ERROR
4853 027722 104401 4$: SCOP1 ;SW09=1?
4854 027724 005202 INC R2 ;NEXT DATA
4855 027726 005200 INC RO ;NEXT ADDRESS
4856 027730 022700 000010 CMP #10,RO ;DONE YET?
4857 027734 001342 BNE 1$ ;BR IF NO
4858 027736 104400 SCOPE ;SCOPE THIS TEST
4859 027740 377 000 376 5$: .BYTE -1,0,376,-1,-1,252,124,-1
4860 027743 377 377 252
4861 027746 124 377
4862 .EVEN

```

```

***** TEST 114 *****
*ALU TEST
*TEST OF ALU FUNCTION DEC A WITH C BIT CLEARED
*ALU FUNCTION (A-1) CODE=7
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS
*****

```

TEST 114

```

4875 027750 012737 000114 001226 TST114: MOV #114,TSTNO
4876 027756 012737 030124 001216 MOV #TST115,NEXT
4877 027764 012737 030016 001220 MOV #1$,LOCK
4878
4879 027772 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
4880 027774 005000 CLR RO ;MASTER CLEAR M8200-YC
4881 027776 012702 030114 MOV #5$,R2 ;MEM + SP ADDRESS
4882 030002 004737 034736 JSR PC,MEMLD ;POINTER TO CORRECT DATA
4883 030006 035062 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
4884 030010 004737 034772 JSR PC,SPLD ;POINTER TO DATA
4885 030014 035072 SPDAT ;LOAD 8 WORDS OF SP
4886 030016 004737 035036 1$: JSR PC,CLRC ;POINTER TO DATA
4887 030022 042737 000017 030036 BIC #17,2$ ;CLEAR C BIT!
4888 030030 050037 030036 BIS RO,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
4889 030034 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
4890 030036 010000 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD MAR

```

```

4891 030040 042737 000017 030054 BIC #17,3$ :CLEAR ADDRESS OF INSTRUCTION
4892 030046 050037 030054 BIS RO,3$ :ADD ADDRESS TO INSTRUCTION
4893 030052 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4894 030054 040560 3$: 040400!<7*20> BR + DEC A
4895 030056 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4896 030060 061224 MOVB (R2),R5 :MOVE BR TO PORT4
4897 030062 111205 MOVB 4(R1),R4 :PUT "EXPECTED" IN R5
4898 030064 116104 000004 :PUT "FOUND" IN R4
4899 030070 120504 CMPB R5,R4 :DATA CORRECT?
4900 030072 001401 BEQ 4$ :BR IF YES
4901 030074 104015 HLT 15 :ALU ERROR
4902 030076 104401 4$: SCOPI :SW09=1?
4903 030100 005202 INC R2 :NEXT DATA
4904 030102 005200 INC RO :NEXT ADDRESS
4905 030104 022700 000010 CMP #10,RO :DONE YET?
4906 030110 001342 BNE 1$ :BR IF NO
4907 030112 104400 SCOPE :SCOPE THIS TEST
4908 030114 377 377 5$: .BYTE -1,-1,376,376,124,124,251,251
4909 030117 376 124
4910 030122 251 251
4911 .EVEN

```

```

***** TEST 115 *****
*ALU TEST
*TEST OF ALU FUNCTION SEL B WITH C BIT SET
*ALU FUNCTION (B) CODE=11
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS
*****

```

TEST 115

```

4923 ; TEST 115
4924 030124 012737 000115 001226 TST115: MOV #115,TSTNO
4925 030132 012737 030300 001216 MOV #TST116,NEXT
4926 030140 012737 030172 001220 MOV #1$,LOCK
4927 :R1 CONTAINS BASE M8200-YC ADDRESS
4928 030146 104412 MSTCLR :MASTER CLEAR M8200-YC
4929 030150 005000 CLR RO :MEM + SP ADDRESS
4930 030152 012702 030270 MOV #5$,R2 :POINTER TO CORRECT DATA
4931 030156 004737 034736 JSR PC,MEMLD :LOAD 8 WORDS OF MAIN MEMORY
4932 030162 035062 MEMDAT :POINTER TO DATA
4933 030164 004737 034772 JSR PC,SPLD :LOAD 8 WORDS OF SP
4934 030170 035072 SPDAT :POINTER TO DATA
4935 030172 004737 035050 1$: JSR PC,SETC :SET C BIT!
4936 030176 042737 000017 030212 BIC #17,2$ :CLEAR ADDRESS FIELD OF INSTRUCTION
4937 030204 050037 030212 BIS RO,2$ :ADD ADDRESS TO INSTRUCTION
4938 030210 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4939 030212 010000 2$: 010000 :LOAD MAR
4940 030214 042737 000017 030230 BIC #17,3$ :CLEAR ADDRESS OF INSTRUCTION
4941 030222 050037 030230 BIS RO,3$ :ADD ADDRESS TO INSTRUCTION
4942 030226 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4943 030230 040620 3$: 040400!<11*20> :BR + SEL B
4944 030232 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```

```

4945 030234 061224 61224 :MOVE BR TO PORT4
4946 030236 111205 :PUT "EXPECTED" IN R5
4947 030240 116104 000004 :PUT "FOUND" IN R4
4948 030244 120504 :DATA CORRECT?
4949 030246 001401 :BR IF YES
4950 030250 104015 :ALU ERROR
4951 030252 104401 4$: :SW09=1?
4952 030254 005202 :NEXT DATA
4953 030256 005200 :NEXT ADDRESS
4954 030260 022700 000010 :DONE YET?
4955 030264 001342 :BR IF NO
4956 030266 104400 :SCOPE THIS TEST
4957 030270 000 377 000 5$: .BYTE 0,-1,0,-1,125,252,125,252
4958 030273 377 125 252
4959 030276 125 252
4960 :.EVEN
    
```

```

***** TEST 116 *****
*ALU TEST
*TEST OF ALU FUNCTION SEL A WITH C BIT SET
*ALU FUNCTION (A) CODE=10
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS
*****
    
```

```

; TEST 116
-----
4973 030300 012737 000116 001226 TST116: MOV #116,TSTNO
4974 030306 012737 030454 001216 MOV #TST117,NEXT
4975 030314 012737 030346 001220 MOV #1$,LOCK
4976
4977 030322 104412 MSTCLR :R1 CONTAINS BASE M8200-YC ADDRESS
4978 030324 005000 CLR RO :MASTER CLEAR M8200-YC
4979 030326 012702 030444 MOV #5$,R2 :MEM + SP ADDRESS
4980 030332 004737 034736 JSR PC,MEMLD :POINTER TO CORRECT DATA
4981 030336 035062 MEMDAT :LOAD 8 WORDS OF MAIN MEMORY
4982 030340 004737 034772 JSR PC,SPLD :POINTER TO DATA
4983 030344 035072 SPDAT :LOAD 8 WORDS OF SP
4984 030346 004737 035050 1$: JSR PC,SETC :POINTER TO DATA
4985 030352 042737 000017 030366 BIC #17,2$ :SET C BIT!
4986 030360 050037 030366 BIS RO,2$ :CLEAR ADDRESS FIELD OF INSTRUCTION
4987 030364 104414 ROMCLK :ADD ADDRESS TO INSTRUCTION
4988 030366 010000 010000 2$: ROMCLK PC=5304
4989 030370 042737 000017 030404 BIC #17,3$ :LOAD MAR
4990 030376 050037 030404 BIS RO,3$ :CLEAR ADDRESS OF INSTRUCTION
4991 030402 104414 ROMCLK :ADD ADDRESS TO INSTRUCTION
4992 030404 040600 040400! <10*20> 3$: ROMCLK PC=5304
4993 030406 104414 ROMCLK :BR + SEL A
4994 030410 061224 61224 :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4995 030412 111205 :MOVE BR TO PORT4
4996 030414 116104 000004 :PUT "EXPECTED" IN R5
4997 030420 120504 :PUT "FOUND" IN R4
4998 030422 001401 :DATA CORRECT?
:BEQ 4$ :BR IF YES
    
```

```

4999 030424 104015          HLT      15          ;ALU ERROR
5000 030426 104401          4$: SCOP1          ;SW09=1?
5001 030430 005202          INC      R2          ;NEXT DATA
5002 030432 005200          INC      R0          ;NEXT ADDRESS
5003 030434 022700 000010    CMP      #10,R0      ;DONE YET?
5004 030440 001342          BNE     1$          ;BR IF NO
5005 030442 104400          SCOPE          ;SCOPE THIS TEST
5006 030444      000      000      377 5$: .BYTE 0,0,-1,-1,125,125,252,252
5007 030447      377      125      125
5008 030452      252      252
5009                                     .EVEN

```

```

5010
5011
5012
5013
5014
5015
5016
5017
5018
5019
5020

```

```

:***** TEST 117 *****
:ALU TEST
:TEST OF ALU FUNCTION A OR NOTB WITH C BIT SET
:ALU FUNCTION (A OR NOTB) CODE=12
:LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
:PERFORM THE FUNCTION, VERIFY THE RESULTS
:*****

```

```

5021
5022
5023
5024
5025
5026
5027
5028
5029
5030
5031
5032
5033
5034
5035
5036
5037
5038
5039
5040
5041
5042
5043
5044
5045
5046
5047
5048
5049
5050
5051
5052

```

```

: TEST 117
:-----
5022 030454 012737 000117 001226 TST117: MOV      #117,TSTNO
5023 030462 012737 030630 001216      MOV      #TST120,NEXT
5024 030470 012737 030522 001220      MOV      #1$,LOCK
5026 030476 104412          MSTCLR          ;R1 CONTAINS BASE M8200-YC ADDRESS
5027 030500 005000          CLR      R0      ;MASTER CLEAR M8200-YC
5028 030502 012702 030620      MOV      #5$,R2  ;MEM + SP ADDRESS
5029 030506 004737 034736      JSR     PC,MEMLD ;POINTER TO CORRECT DATA
5030 030512 035062          MEMDAT          ;LOAD 8 WORDS OF MAIN MEMORY
5031 030514 004737 034772      JSR     PC,SPLD  ;POINTER TO DATA
5032 030520 035072          SPDAT          ;LOAD 8 WORDS OF SP
5033 030522 004737 035050      JSR     PC,SETC  ;POINTER TO DATA
5034 030526 042737 000017 030542 1$: BIC     #17,2$   ;SET C BIT!
5035 030534 050037 030542      BIS     R0,2$   ;CLEAR ADDRESS FIELD OF INSTRUCTION
5036 030540 104414          ROMCLK          ;ADD ADDRESS TO INSTRUCTION
5037 030542 010000          2$: ROMCLK 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5038 030544 042737 000017 030560      BIC     #17,3$   ;LOAD MAR
5039 030552 050037 030560      BIS     R0,3$   ;CLEAR ADDRESS OF INSTRUCTION
5040 030556 104414          ROMCLK          ;ADD ADDRESS TO INSTRUCTION
5041 030560 040640          3$: ROMCLK 040400! <12*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5042 030562 104414          ROMCLK          ;BR + A OR NOTB
5043 030564 061224          61224          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5044 030566 111205          MOVB     (R2),R5 ;MOVE BR TO PORT4
5045 030570 116104 000004      MOVB     4(R1),R4 ;PUT "EXPECTED" IN R5
5046 030574 120504          CMPB     R5,R4   ;PUT "FOUND" IN R4
5047 030576 001401          BEQ     4$      ;DATA CORRECT?
5048 030600 104015          HLT     15      ;BR IF YES
5049 030602 104401          4$: SCOP1          ;ALU ERROR
5050 030604 005202          INC      R2      ;SW09=1?
5051 030606 005200          INC      R0      ;NEXT DATA
5052 030610 022700 000010    CMP      #10,R0   ;NEXT ADDRESS
                    ;DONE YET?

```

```

5053 030614 001342 BNE 1$ ;BR IF NO
5054 030616 104400 SCOPE ;SCOPE THIS TEST
5055 030620 377 000 377 5$: .BYTE -1,0,-1,-1,-1,125,252,-1
5056 030623 377 377 125
5057 030626 252 377
5058 .EVEN
5059
5060

```

```

***** TEST 120 *****
*ALU TEST
*TEST OF ALU FUNCTION A AND B WITH C BIT SET
*ALU FUNCTION (A AND B) CODE=13
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS
*****

```

```

5061
5062
5063
5064
5065
5066
5067
5068
5069
5070 ; TEST 120
5071 030630 012737 000120 001226 TST120: MOV #120,TSTNO
5072 030636 012737 031004 001216 MOV #TST121,NEXT
5073 030644 012737 030676 001220 MOV #1$,LOCK
5074
5075 030652 104412 MSTCLR ;R1 CONTAINS BASE MB200-YC ADDRESS
5076 030654 005000 CLR RO ;MASTER CLEAR MB200-YC
5077 030656 012702 030774 MOV #5$,R2 ;MEM + SP ADDRESS
5078 030662 004737 034736 JSR PC,MEMLD ;POINTER TO CORRECT DATA
5079 030666 035062 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
5080 030670 004737 034772 JSR PC,SPLD ;POINTER TO DATA
5081 030674 035072 SPDAT ;LOAD 8 WORDS OF SP
5082 030676 004737 035050 1$: JSR PC,SETC ;POINTER TO DATA
5083 030702 042737 000017 030716 BIC #17,2$ ;SET C BIT!
5084 030710 050037 030716 BIS RO,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
5085 030714 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5086 030716 010000 010000 2$: ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5087 030720 042737 000017 030734 BIC #17,3$ ;LOAD MAR
5088 030726 050037 030734 BIS RO,3$ ;CLEAR ADDRESS OF INSTRUCTION
5089 030732 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5090 030734 040660 040400! <13*20> 3$: ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5091 030736 104414 ROMCLK ;BR + A AND B
5092 030740 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5093 030742 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
5094 030744 116104 000004 MOVB 4(R1),R4 ;PUT "EXPECTED" IN R5
5095 030750 120504 CMPB R5,R4 ;PUT "FOUND" IN R4
5096 030752 001401 BEQ 4$ ;DATA CORRECT?
5097 030754 104015 HLT 15 ;BR IF YES
5098 030756 104401 4$: SCOP1 ;ALU ERROR
5099 030760 005202 INC R2 ;SW09=1?
5100 030762 005200 INC RO ;NEXT DATA
5101 030764 022700 000010 CMP #10,RO ;NEXT ADDRESS
5102 030770 001342 BNE 1$ ;DONE YET?
5103 030772 104400 SCOPE ;BR IF NO
5104 030774 000 000 000 5$: .BYTE 0,0,0,-1,125,0,0,252 ;SCOPE THIS TEST
5105 030777 377 125 000
5106 031002 000 252

```

5107 .EVEN

***** TEST 121 *****
*ALU TEST
*TEST OF ALU FUNCTION A OR B WITH C BIT SET
*ALU FUNCTION (A OR B) CODE=14
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS

TEST 121

5120 031004 012737 000121 001226 TST121:
5121 031012 012737 031160 001216
5122 031020 012737 031052 001220
5124 031026 104412
5125 031030 005000
5126 031032 012702 031150
5127 031036 004737 034736
5128 031042 035062
5129 031044 004737 034772
5130 031050 035072
5131 031052 004737 035050 1\$:
5132 031056 042737 000017 031072
5133 031064 050037 031072
5134 031070 104414
5135 031072 010000 2\$:
5136 031074 042737 000017 031110
5137 031102 050037 031110
5138 031106 104414
5139 031110 040700 3\$:
5140 031112 104414
5141 031114 061224
5142 031116 111205
5143 031120 116104 000004
5144 031124 120504
5145 031126 001401
5146 031130 104015
5147 031132 104401 4\$:
5148 031134 005202
5149 031136 005200
5150 031140 022700 000010
5151 031144 001342
5152 031146 104400
5153 031150 000 377
5154 031153 377 125 377
5155 031156 377 252

MOV #121,TSTNO
MOV #TST122,NEXT
MOV #1\$,LOCK
MSTCLR
CLR R0
MOV #5\$,R2
JSR PC,MEMLD
MEMDAT
JSR PC,SPLD
SPDAT
JSR PC,SETC
BIC #17,2\$
BIS R0,2\$
ROMCLK
010000
BIC #17,3\$
BIS R0,3\$
ROMCLK
040400! <14*20>
ROMCLK
61224
MOVB (R2),R5
MOVB 4(R1),R4
CMPB R5,R4
BEQ 4\$
HLT 15
SCOPE
INC R2
INC R0
CMP #10,R0
BNE 1\$
SCOPE
.BYTE 0,-1,-1,-1,125,-1,-1,252

:R1 CONTAINS BASE MB200-YC ADDRESS
:MASTER CLEAR MB200-YC
:MEM + SP ADDRESS
:POINTER TO CORRECT DATA
:LOAD 8 WORDS OF MAIN MEMORY
:POINTER TO DATA
:LOAD 8 WORDS OF SP
:POINTER TO DATA
:SET C BIT!
:CLEAR ADDRESS FIELD OF INSTRUCTION
:ADD ADDRESS TO INSTRUCTION
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:LOAD MAR
:CLEAR ADDRESS OF INSTRUCTION
:ADD ADDRESS TO INSTRUCTION
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:BR + A OR B
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:MOVE BR TO PORT4
:PUT "EXPECTED" IN R5
:PUT "FOUND" IN R4
:DATA CORRECT?
:BR IF YES
:ALU ERROR
:SW09=1?
:NEXT DATA
:NEXT ADDRESS
:DONE YET?
:BR IF NO
:SCOPE THIS TEST

5156 .EVEN

***** TEST 122 *****
*ALU TEST

5157
5158
5159
5160

```

5161 ;*TEST OF ALU FUNCTION A XOR B WITH C BIT SET
5162 ;*ALU FUNCTION (A XOR B) CODE=15
5163 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5164 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5165 ;*****
5166
5167 ;
5168 ; TEST 122

```

```

5169 031160 012737 000122 001226 TST122: MOV #122,TSTNO
5170 031166 012737 031334 001216 MOV #TST123,NEXT
5171 031174 012737 031226 001220 MOV #1$,LOCK
5172
5173 031202 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
5174 031204 005000 CLR RO ;MASTER CLEAR M8200-YC
5175 031206 012702 031324 MOV #5$,R2 ;MEM + SP ADDRESS
5176 031212 004737 034736 JSR PC,MEMLD ;POINTER TO CORRECT DATA
5177 031216 035062 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
5178 031220 004737 034772 JSR PC,SPLD ;POINTER TO DATA
5179 031224 035072 SPDAT ;LOAD 8 WORDS OF SP
5180 031226 004737 035050 1$: JSR PC,SETC ;POINTER TO DATA
5181 031232 042737 000017 031246 BIC #17,2$ ;SET C BIT!
5182 031240 050037 031246 BIS RO,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
5183 031244 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5184 031246 010000 2$: 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5185 031250 042737 000017 031264 BIC #17,3$ ;LOAD MAR
5186 031256 050037 031264 BIS RO,3$ ;CLEAR ADDRESS OF INSTRUCTION
5187 031262 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5188 031264 040720 3$: 040400! <15*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5189 031266 104414 ROMCLK ;BR + A XOR B
5190 031270 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5191 031272 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
5192 031274 116104 000004 MOVB 4(R1),R4 ;PUT "EXPECTED" IN R5
5193 031300 120504 CMPB R5,R4 ;PUT "FOUND" IN R4
5194 031302 001401 BEQ 4$ ;DATA CORRECT?
5195 031304 104015 HLT 15 ;BR IF YES
5196 031306 104401 4$: SCOP1 ;ALU ERROR
5197 031310 005202 INC R2 ;SW09=1?
5198 031312 005200 INC RO ;NEXT DATA
5199 031314 022700 000010 CMP #10,RO ;NEXT ADDRESS
5200 031320 001342 BNE 1$ ;DONE YET?
5201 031322 104400 SCOPE ;BR IF NO
5202 031324 000 377 377 5$: .BYTE 0,-1,-1,0,0,-1,-1,0 ;SCOPE THIS TEST
5203 031327 000 377
5204 031332 377 000
5205 .EVEN

```

```

5206
5207 ;***** TEST 123 *****
5208 ;*ALU TEST
5209 ;*TEST OF ALU FUNCTION ADD WITH C BIT SET
5210 ;*ALU FUNCTION (A PLUS B) CODE=00
5211 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5212 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5213 ;*****
5214

```

```

5215 ; TEST 123
5216 ; -----
5217 ;
5218 031334 012737 000123 001226 TST123: MOV #123,TSTNO
5219 031342 012737 031510 001216 MOV #TST124,NEXT
5220 031350 012737 031402 001220 MOV #1$,LOCK
5221 ;
5222 031356 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
5223 031360 005000 CLR ;MASTER CLEAR M8200-YC
5224 031362 012702 031500 MOV #5$,R2 ;MEM + SP ADDRESS
5225 031366 004737 034736 JSR PC,MEMLD ;POINTER TO CORRECT DATA
5226 031372 035062 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
5227 031374 004737 034772 JSR PC,SPLD ;POINTER TO DATA
5228 031400 035072 SPDAT ;LOAD 8 WORDS OF SP
5229 031402 004737 035050 1$: JSR PC,SETC ;POINTER TO DATA
5230 031406 042737 000017 031422 BIC #17,2$ ;SET C BIT!
5231 031414 050037 031422 BIS RO,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
5232 031420 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5233 031422 010000 010000 2$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5234 031424 042737 000017 031440 BIC #17,3$ ;LOAD MAR
5235 031432 050037 031440 BIS RO,3$ ;CLEAR ADDRESS OF INSTRUCTION
5236 031436 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5237 031440 040400! <00*20> 3$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5238 031442 104414 ROMCLK ;BR + ADD
5239 031444 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5240 031446 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
5241 031450 116104 000004 MOVB 4(R1),R4 ;PUT "EXPECTED" IN R5
5242 031454 120504 CMPB R5,R4 ;PUT "FOUND" IN R4
5243 031456 001401 BEQ 4$ ;DATA CORRECT?
5244 031460 104015 HLT 15 ;BR IF YES
5245 031462 104401 4$: SCOP1 ;ALU ERROR
5246 031464 005202 INC R2 ;SW09=1?
5247 031466 005200 INC RO ;NEXT DATA
5248 031470 022700 000010 CMP #10,RO ;NEXT ADDRESS
5249 031474 001342 BNE 1$ ;DONE YET?
5250 031476 104400 SCOPE 1$ ;BR IF NO
5251 031500 000 377 377 5$: .BYTE 0,-1,-1,376,252,-1,-1,124 ;SCOPE THIS TEST
5252 031503 376 252 377
5253 031506 377 124
5254 .EVEN

```

```

***** TEST 124 *****
*ALU TEST
*TEST OF ALU FUNCTION 2A W/C WITH C BIT SET
*ALU FUNCTION (A PLUS A PLUS C) CODE=6
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS
*****

```

```

5265 ; TEST 124
5266 ; -----
5267 031510 012737 000124 001226 TST124: MOV #124,TSTNO
5268 031516 012737 031664 001216 MOV #TST125,NEXT

```

```

5269 031524 012737 031556 001220      MOV      #1$,LOCK
5270                                     ;R1 CONTAINS BASE M8200-YC ADDRESS
5271 031532 104412      MSTCLR   ;MASTER CLEAR M8200-YC
5272 031534 005000      CLR      RO      ;MEM + SP ADDRESS
5273 031536 012702 031654      MOV      #5$,R2   ;POINTER TO CORRECT DATA
5274 031542 004737 034736      JSR      PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
5275 031546 035062      MEMDAT   ;POINTER TO DATA
5276 031550 004737 034772      JSR      PC,SPLD  ;LOAD 8 WORDS OF SP
5277 031554 035072      SPDAT   ;POINTER TO DATA
5278 031556 004737 035050      JSR      PC,SETC  ;SET C BIT!
5279 031562 042737 000017 031576 1$:  BIC      #17,2$   ;CLEAR ADDRESS FIELD OF INSTRUCTION
5280 031570 050037 031576      BIS      RO,2$   ;ADD ADDRESS TO INSTRUCTION
5281 031574 104414      ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5282 031576 010000 010000 2$:  O10000 ;LOAD MAR
5283 031600 042737 000017 031614      BIC      #17,3$   ;CLEAR ADDRESS OF INSTRUCTION
5284 031606 050037 031614      BIS      RO,3$   ;ADD ADDRESS TO INSTRUCTION
5285 031612 104414      ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5286 031614 040540 040400! <6*20> 3$:  ROMCLK ;BR + 2A W/C
5287 031616 104414      ROMCLK   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5288 031620 061224      61224   ;MOVE BR TO PORT4
5289 031622 111205      MOVB     (R2),R5 ;PUT "EXPECTED" IN R5
5290 031624 116104 000004      MOVB     4(R1),R4 ;PUT "FOUND" IN R4
5291 031630 120504      CMPB     R5,R4   ;DATA CORRECT?
5292 031632 001401      BEQ      4$     ;BR IF YES
5293 031634 104015      HLT      1$     ;ALU ERROR
5294 031636 104401 104401 4$:  SCOP1   ;SW09=1?
5295 031640 005202      INC      R2     ;NEXT DATA
5296 031642 005200      INC      RO     ;NEXT ADDRESS
5297 031644 022700 000010      CMP      #10,RO ;DONE YET?
5298 031650 001342      BNE      1$     ;BR IF NO
5299 031652 104400      SCOPE   ;SCOPE THIS TEST
5300 031654 001 001 377 5$:  .BYTE   1,1,-1,-1,253,253,125,125
5301 031657 377 253
5302 031662 125 125
5303                                     .EVEN

```

```

***** TEST 125 *****
;ALU TEST
;TEST OF ALU FUNCTION SUB WITH C BIT SET
;ALU FUNCTION (A-B) CODE=16
;LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
;PERFORM THE FUNCTION, VERIFY THE RESULTS
:*****

```

TEST 125

```

5315
5316 031664 012737 000125 001226 TST125: MOV      #125,TSTNO
5317 031672 012737 032040 001216      MOV      #TST126,NEXT
5318 031700 012737 031732 001220      MOV      #1$,LOCK
5319                                     ;R1 CONTAINS BASE M8200-YC ADDRESS
5320 031706 104412      MSTCLR   ;MASTER CLEAR M8200-YC
5321 031710 005000      CLR      RO      ;MEM + SP ADDRESS
5322 031712 012702 032030      MOV      #5$,R2   ;POINTER TO CORRECT DATA

```

```

5323 031716 004737 034736 JSR PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
5324 031722 035062 MEMDAT ;POINTER TO DATA
5325 031724 004737 034772 JSR PC,SPLD ;LOAD 8 WORDS OF SP
5326 031730 035072 SPDAT ;POINTER TO DATA
5327 031732 004737 035050 1$: JSR PC,SETC ;SET C BIT!
5328 031736 042737 000017 031752 BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
5329 031744 050037 031752 BIS RO,2$ ;ADD ADDRESS TO INSTRUCTION
5330 031750 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5331 031752 010000 010000 2$: ;LOAD MAR
5332 031754 042737 000017 031770 BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
5333 031762 050037 031770 BIS RO,3$ ;ADD ADDRESS TO INSTRUCTION
5334 031766 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5335 031770 040740 040400! <16*20> 3$: BR + SUB
5336 031772 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5337 031774 061224 61224 ;MOVE BR TO PORT4
5338 031776 111205 MOVB (R2),R5 ;PUT "EXPECTED" IN R5
5339 032000 116104 000004 MOVB 4(R1),R4 ;PUT "FOUND" IN R4
5340 032004 120504 CMPB R5,R4 ;DATA CORRECT?
5341 032006 001401 BEQ 4$ ;BR IF YES
5342 032010 104015 HLT 15 ;ALU ERROR
5343 032012 104401 4$: SCOP1 ;SW09=1?
5344 032014 005202 INC R2 ;NEXT DATA
5345 032016 005200 INC RO ;NEXT ADDRESS
5346 032020 022700 000010 CMP #10,RO ;DONE YET?
5347 032024 001342 BNE 1$ ;BR IF NO
5348 032026 104400 SCOPE ;SCOPE THIS TEST
5349 032030 000 001 377 5$: .BYTE 0,1,-1,0,0,253,125,0
5350 032033 000 000 253
5351 032036 125 000
5352 .EVEN

```

```

***** TEST 126 *****
;ALU TEST
;TEST OF ALU FUNCTION ADD W/C WITH C BIT SET
;ALU FUNCTION (A PLUS B PLUS C) CODE=01
;LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
;PERFORM THE FUNCTION, VERIFY THE RESULTS
;*****

```

TEST 126

```

5364 ;-----
5365 032040 012737 000126 001226 TST126: MOV #126,TSTNO
5366 032046 012737 032214 001216 MOV #TST127,NEXT
5367 032054 012737 032106 001220 MOV #1$,LOCK
5368 ;R1 CONTAINS BASE M8200-YC ADDRESS
5369 032062 104412 MSTCLR ;MASTER CLEAR M8200-YC
5370 032064 005000 CLR RO ;MEM + SP ADDRESS
5371 032066 012702 032204 MOV #5$,R2 ;POINTER TO CORRECT DATA
5372 032072 004737 034736 JSR PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
5373 032076 035062 MEMDAT ;POINTER TO DATA
5374 032100 004737 034772 JSR PC,SPLD ;LOAD 8 WORDS OF SP
5375 032104 035072 SPDAT ;POINTER TO DATA
5376 032106 004737 035050 1$: JSR PC,SETC ;SET C BIT!

```

5377	032112	042737	000017	032126		BIC	#17,2\$;	CLEAR ADDRESS FIELD OF INSTRUCTION
5378	032120	050037	032126			BIS	RO,2\$;	ADD ADDRESS TO INSTRUCTION
5379	032124	104414				ROMCLK		;	NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5380	032126	010000			2\$:	010000		;	LOAD MAR
5381	032130	042737	000017	032144		BIC	#17,3\$;	CLEAR ADDRESS OF INSTRUCTION
5382	032136	050037	032144			BIS	RO,3\$;	ADD ADDRESS TO INSTRUCTION
5383	032142	104414				ROMCLK		;	NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5384	032144	040420			3\$:	040400!	<01*20>	;	BR + ADD W/C
5385	032146	104414				ROMCLK		;	NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5386	032150	061224				61224		;	MOVE BR TO PORT4
5387	032152	111205				MOVB	(R2),R5	;	PUT "EXPECTED" IN R5
5388	032154	116104	000004			MOVB	4(R1),R4	;	PUT "FOUND" IN R4
5389	032160	120504				CMPB	R5,R4	;	DATA CORRECT?
5390	032162	001401				BEQ	4\$;	BR IF YES
5391	032164	104015				HLT	1\$;	ALU ERROR
5392	032166	104401			4\$:	SCOPE1		;	SW09=1?
5393	032170	005202				INC	R2	;	NEXT DATA
5394	032172	005200				INC	RO	;	NEXT ADDRESS
5395	032174	022700	000010			CMP	#10,RO	;	DONE YET?
5396	032200	001342				BNE	1\$;	BR IF NO
5397	032202	104400				SCOPE		;	SCOPE THIS TEST
5398	032204	001	000	000	5\$:	.BYTE	1,0,0,-1,253,0,0,125		
5399	032207	377	253	000					
5400	032212	000	125						

.EVEN

```

***** TEST 127 *****
*ALU TEST
*TEST OF ALU FUNCTION SUB W/C WITH C BIT SET
*ALU FUNCTION (A-B-C) CODE=2
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS
*****

```

TEST 127

5414	032214	012737	000127	001226	TST127:	MOV	#127,TSTNO		
5415	032222	012737	032370	001216		MOV	#TST130,NEXT		
5416	032230	012737	032262	001220		MOV	#1\$,LOCK		
5418	032236	104412				MSTCLR		;	R1 CONTAINS BASE M8200-YC ADDRESS
5419	032240	005000				CLR	RO	;	MASTER CLEAR M8200-YC
5420	032242	012702	032360			MOV	#5\$,R2	;	MEM + SP ADDRESS
5421	032246	004737	034736			JSR	PC,MEMLD	;	POINTER TO CORRECT DATA
5422	032252	035062				MEMDAT		;	LOAD 8 WORDS OF MAIN MEMORY
5423	032254	004737	034772			JSR	PC,SPLD	;	POINTER TO DATA
5424	032260	035072				SPDAT		;	LOAD 8 WORDS OF SP
5425	032262	004737	035050		1\$:	JSR	PC,SETC	;	POINTER TO DATA
5426	032266	042737	000017	032302		BIC	#17,2\$;	SET C BIT!
5427	032274	050037	032302			BIS	RO,2\$;	CLEAR ADDRESS FIELD OF INSTRUCTION
5428	032300	104414				ROMCLK		;	ADD ADDRESS TO INSTRUCTION
5429	032302	010000			2\$:	010000		;	NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5430	032304	042737	000017	032320		BIC	#17,3\$;	LOAD MAR
								;	CLEAR ADDRESS OF INSTRUCTION

5431	032312	050037	032320			BIS	RO,3\$:ADD ADDRESS TO INSTRUCTION
5432	032316	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5433	032320	040440		3\$:		040400! <2*20>		:BR + SUB W/C
5434	032322	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5435	032324	061224				61224		:MOVE BR TO PORT4
5436	032326	111205				MOVB	(R2),R5	:PUT "EXPECTED" IN R5
5437	032330	116104	000004			MOVB	4(R1),R4	:PUT "FOUND" IN R4
5438	032334	120504				CMPB	R5,R4	:DATA CORRECT?
5439	032336	001401				BEQ	4\$:BR IF YES
5440	032340	104015				HLT	1\$:ALU ERROR
5441	032342	104401		4\$:		SCOPE1		:SW09=1?
5442	032344	005202				INC	R2	:NEXT DATA
5443	032346	005200				INC	RO	:NEXT ADDRESS
5444	032350	022700	000010			CMP	#10,RO	:DONE YET?
5445	032354	001342				BNE	1\$:BR IF NO
5446	032356	104400				SCOPE		:SCOPE THIS TEST
5447	032360	000	001	377	5\$:	.BYTE	0,1,-1,0,0,253,125,0	
5448	032363	000	000	253				
5449	032366	125	000					

.EVEN

```

***** TEST 130 *****
*ALU TEST
*TEST OF ALU FUNCTION INC A WITH C BIT SET
*ALU FUNCTION (A PLUS 1) CODE=3
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS
*****

```

TEST 130

5463	032370	012737	000130	001226	TST130:	MOV	#130,TSTNO	
5464	032376	012737	032544	001216		MOV	#TST131,NEXT	
5465	032404	012737	032436	001220		MOV	#1\$,LOCK	
5466								:R1 CONTAINS BASE M8200-YC ADDRESS
5467	032412	104412				MSTCLR		:MASTER CLEAR M8200-YC
5468	032414	005000				CLR	RO	:MEM + SP ADDRESS
5469	032416	012702	032534			MOV	#5\$,R2	:POINTER TO CORRECT DATA
5470	032422	004737	034736			JSR	PC,MEMLD	:LOAD 8 WORDS OF MAIN MEMORY
5471	032426	035062				MEMDAT		:POINTER TO DATA
5472	032430	004737	034772			JSR	PC,SPLD	:LOAD 8 WORDS OF SP
5473	032434	035072				SPDAT		:POINTER TO DATA
5474	032436	004737	035050		1\$:	JSR	PC,SETC	:SET C BIT!
5475	032442	042737	000017	032456		BIC	#17,2\$:CLEAR ADDRESS FIELD OF INSTRUCTION
5476	032450	050037	032456			BIS	RO,2\$:ADD ADDRESS TO INSTRUCTION
5477	032454	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5478	032456	010000			2\$:	010000		:LOAD MAR
5479	032460	042737	000017	032474		BIC	#17,3\$:CLEAR ADDRESS OF INSTRUCTION
5480	032466	050037	032474			BIS	RO,3\$:ADD ADDRESS TO INSTRUCTION
5481	032472	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5482	032474	040460			3\$:	040400! <3*20>		:BR + INC A
5483	032476	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5484	032500	061224				61224		:MOVE BR TO PORT4

5485	032502	111205				MOVB	(R2),R5		:PUT "EXPECTED" IN R5
5486	032504	116104	000004			MOVB	4(R1),R4		:PUT "FOUND" IN R4
5487	032510	120504				CMPB	R5,R4		:DATA CORRECT?
5488	032512	001401				BEQ	4\$:BR IF YES
5489	032514	104015				HLT	15		:ALU ERROR
5490	032516	104401			4\$:	SCOPE			:SWD9=1?
5491	032520	005202				INC	R2		:NEXT DATA
5492	032522	005200				INC	R0		:NEXT ADDRESS
5493	032524	022700	000010			CMP	#10,R0		:DONE YET?
5494	032530	001342				BNE	1\$:BR IF NO
5495	032532	104400				SCOPE			:SCOPE THIS TEST
5496	032534	001	001	000	5\$:	.BYTE	1,1,0,0,126,126,253,253		
5497	032537	000	126	126					
5498	032542	253	253						
5499									.EVEN
5500									
5501									
5502									:***** TEST 131 *****
5503									:*ALU TEST
5504									:*TEST OF ALU FUNCTION 2A WITH C BIT SET
5505									:*ALU FUNCTION (A PLUS A) CODE=5
5506									:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5507									:*PERFORM THE FUNCTION, VERIFY THE RESULTS
5508									:*****
5509									
5510									: TEST 131
5511									
5512	032544	012737	000131	001226	TST131:	MOV	#131,TSTNO		
5513	032552	012737	032720	001216		MOV	#TST132,NEXT		
5514	032560	012737	032612	001220		MOV	#1\$,LOCK		
5515									:R1 CONTAINS BASE M8200-YC ADDRESS
5516	032566	104412				MSTCLR			:MASTER CLEAR M8200-YC
5517	032570	005000				CLR	R0		:MEM + SP ADDRESS
5518	032572	012702	032710			MOV	#5\$,R2		:POINTER TO CORRECT DATA
5519	032576	004737	034736			JSR	PC,MEMLD		:LOAD 8 WORDS OF MAIN MEMORY
5520	032602	035062				MEMDAT			:POINTER TO DATA
5521	032604	004737	034772			JSR	PC,SPLD		:LOAD 8 WORDS OF SP
5522	032610	035072				SPDAT			:POINTER TO DATA
5523	032612	004737	035050		1\$:	JSR	PC,SETC		:SET C BIT!
5524	032616	042737	000017	032632		BIC	#17,2\$:CLEAR ADDRESS FIELD OF INSTRUCTION
5525	032624	050037	032632			BIS	R0,2\$:ADD ADDRESS TO INSTRUCTION
5526	032630	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5527	032632	010000			2\$:	010000			:LOAD MAR
5528	032634	042737	000017	032650		BIC	#17,3\$:CLEAR ADDRESS OF INSTRUCTION
5529	032642	050037	032650			BIS	R0,3\$:ADD ADDRESS TO INSTRUCTION
5530	032646	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5531	032650	040520			3\$:	040400! <5*20>			:BR + 2A
5532	032652	104414				ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5533	032654	061224				61224			:MOVE BR TO PORT4
5534	032656	111205				MOVB	(R2),R5		:PUT "EXPECTED" IN R5
5535	032660	116104	000004			MOVB	4(R1),R4		:PUT "FOUND" IN R4
5536	032664	120504				CMPB	R5,R4		:DATA CORRECT?
5537	032666	001401				BEQ	4\$:BR IF YES
5538	032670	104015				HLT	15		:ALU ERROR

```

5539 032672 104401          4$: SCOP1          ;SW09=1?
5540 032674 005202          INC R2          ;NEXT DATA
5541 032676 005200          INC R0          ;NEXT ADDRESS
5542 032700 022700 000010    CMP #10,R0      ;DONE YET?
5543 032704 001342          BNE 1$         ;BR IF NO
5544 032706 104400          SCOPE          ;SCOPE THIS TEST
5545 032710 000 000 376 5$: .BYTE 0,0,376,376,252,252,124,124
5546 032713 376 252
5547 032716 124 124
    
```

.EVEN

```

***** TEST 132 *****
*ALU TEST
*TEST OF ALU FUNCTION A PLUS C WITH C BIT SET
*ALU FUNCTION (A PLUS C) CODE=4
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS
*****
    
```

TEST 132

```

5561 032720 012737 000132 001226 TST132: MOV #132,TSTNO
5562 032726 012737 033074 001216 MOV #TST133,NEXT
5563 032734 012737 032766 001220 MOV #1$,LOCK
5564
5565 032742 104412          MSTCLR
5566 032744 005000          CLR R0
5567 032746 012702 033064    MOV #5$,R2
5568 032752 004737 034736    JSR PC,MEMLD
5569 032756 035062          MEMDAT
5570 032760 004737 034772    JSR PC,SPLD
5571 032764 035072          SPDAT
5572 032766 004737 035050    JSR PC,SETC
5573 032772 042737 000017 033006 1$: BIC #17,2$
5574 033000 050037 033006    BIS R0,2$
5575 033004 104414          ROMCLK
5576 033006 010000          010000
5577 033010 042737 000017 033024 2$: BIC #17,3$
5578 033016 050037 033024    BIS R0,3$
5579 033022 104414          ROMCLK
5580 033024 040500          040400! <4*20>
5581 033026 104414          ROMCLK
5582 033030 061224          61224
5583 033032 111205          MOVB (R2),R5
5584 033034 116104 000004    MOVB 4(R1),R4
5585 033040 120504          CMPB R5,R4
5586 033042 001401          BEQ 4$
5587 033044 104015          HLT 1$
5588 033046 104401          4$: SCOP1
5589 033050 005202          INC R2
5590 033052 005200          INC R0
5591 033054 022700 000010    CMP #10,R0
5592 033060 001342          BNE 1$
    
```

```

;R1 CONTAINS BASE M8200-YC ADDRESS
;MASTER CLEAR M8200-YC
;MEM + SP ADDRESS
;POINTER TO CORRECT DATA
;LOAD 8 WORDS OF MAIN MEMORY
;POINTER TO DATA
;LOAD 8 WORDS OF SP
;POINTER TO DATA
;SET C BIT!
;CLEAR ADDRESS FIELD OF INSTRUCTION
;ADD ADDRESS TO INSTRUCTION
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD MAR
;CLEAR ADDRESS OF INSTRUCTION
;ADD ADDRESS TO INSTRUCTION
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;BR + A PLUS C
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOVE BR TO PORT4
;PUT "EXPECTED" IN R5
;PUT "FOUND" IN R4
;DATA CORRECT?
;BR IF YES
;ALU ERROR
;SW09=1?
;NEXT DATA
;NEXT ADDRESS
;DONE YET?
;BR IF NO
    
```

```

5593 033062 104400          SCOPE          SCOPE THIS TEST
5594 033064          001          001          000 5$: .BYTE 1,1,0,0,126,126,253,253
5595 033067          000          126          126
5596 033072          253          253
5597                                     .EVEN
5598
5599
5600
5601
5602
5603
5604
5605
5606
5607
5608
5609

```

```

:***** TEST 133 *****
:ALU TEST
:TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT SET
:ALU FUNCTION (A-B-1) CODE=17
:LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
:PERFORM THE FUNCTION, VERIFY THE RESULTS
:*****

```

TEST 133

```

5610 033074 012737 000133 001226 TST133: MOV #133,TSTNO
5611 033102 012737 033250 001216 MOV #TST134,NEXT
5612 033110 012737 033142 001220 MOV #1$,LOCK
5613
5614 033116 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
5615 033120 005000 CLR RO ;MASTER CLEAR M8200-YC
5616 033122 012702 033240 MOV #5$,R2 ;MEM + SP ADDRESS
5617 033126 004737 034736 JSR PC,MEMLD ;POINTER TO CORRECT DATA
5618 033132 035062 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
5619 033134 004737 034772 JSR PC,SPLD ;POINTER TO DATA
5620 033140 035072 SPDAT ;LOAD 8 WORDS OF SP
5621 033142 004737 035050 1$: JSR PC,SETC ;POINTER TO DATA
5622 033146 042737 000017 033162 BIC #17,2$ ;SET C BIT!
5623 033154 050037 033162 BIS RO,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
5624 033160 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5625 033162 010000 2$: BIC #17,3$ ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5626 033164 042737 000017 033200 BIS RO,3$ ;LOAD MAR
5627 033172 050037 033200 ROMCLK ;CLEAR ADDRESS OF INSTRUCTION
5628 033176 104414 3$: ROMCLK ;ADD ADDRESS TO INSTRUCTION
5629 033200 040760 040400! <17*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5630 033202 104414 ROMCLK ;BR + 2'S COMP SUB
5631 033204 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5632 033206 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
5633 033210 116104 000004 MOVB 4(R1),R4 ;PUT "EXPECTED" IN R5
5634 033214 120504 CMPB R5,R4 ;PUT "FOUND" IN R4
5635 033216 001401 BEQ 4$ ;DATA CORRECT?
5636 033220 104015 HLT 15 ;BR IF YES
5637 033222 104401 4$: SCOP1 ;ALU ERROR
5638 033224 005202 INC R2 ;SW09=1?
5639 033226 005200 INC RO ;NEXT DATA
5640 033230 022700 000010 CMP #10,RO ;NEXT ADDRESS
5641 033234 001342 BNE 1$ ;DONE YET?
5642 033236 104400 SCOPE ;BR IF NO
5643 033240 377 000 376 5$: .BYTE -1,0,376,-1,-1,252,124,-1
5644 033243 377 377 252
5645 033246 124 377
5646                                     .EVEN

```

5647
5648
5649
5650
5651
5652
5653
5654
5655
5656
5657
5658
5659
5660
5661
5662
5663
5664
5665
5666
5667
5668
5669
5670
5671
5672
5673
5674
5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697
5698
5699
5700

```
***** TEST 134 *****
*ALU TEST
*TEST OF ALU FUNCTION DEC A WITH C BIT SET
*ALU FUNCTION (A-1) CODE=7
*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
*PERFORM THE FUNCTION, VERIFY THE RESULTS
*****
```

TEST 134

```
033250 012737 000134 001226 TST134:
033256 012737 033424 001216
033264 012737 033316 001220
033272 104412
033274 005000
033276 012702 033414
033302 004737 034736
033306 035062
033310 004737 034772
033314 035072
033316 004737 035050 1$:
033322 042737 000017 033336
033330 050037 033336
033334 104414
033336 010000 2$:
033340 042737 000017 033354
033346 050037 033354
033352 104414 3$:
033354 040560 <7*20>
033356 104414
033360 061224
033362 111205
033364 116104 000004
033370 120504
033372 001401
033374 104015
033376 104401 4$:
033400 005202
033402 005200
033404 022700 000010
033410 001342
033412 104400
033414 377 377 376 5$:
033417 376 124 124
033422 251 251 251
```

```
MOV #134,TSTNO
MOV #TST135,NEXT
MOV #1$,LOCK
MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
CLR R0 ;MASTER CLEAR M8200-YC
MOV #5$,R2 ;MEM + SP ADDRESS
JSR PC,MEMLD ;POINTER TO CORRECT DATA
MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
JSR PC,SPLD ;POINTER TO DATA
SPDAT ;LOAD 8 WORDS OF SP
BIC #17,2$ ;POINTER TO DATA
BIS R0,2$ ;SET C BIT!
ROMCLK ;CLEAR ADDRESS FIELD OF INSTRUCTION
010000 ;ADD ADDRESS TO INSTRUCTION
BIC #17,3$ ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
BIS R0,3$ ;LOAD MAR
ROMCLK ;CLEAR ADDRESS OF INSTRUCTION
040400! <7*20> ;ADD ADDRESS TO INSTRUCTION
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
61224 ;BR + DEC A
MOV 4(R1),R5 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
MOV 4(R1),R4 ;MOVE BR TO PORT4
CMPB R5,R4 ;PUT "EXPECTED" IN R5
BEQ 4$ ;PUT "FOUND" IN R4
HLT 15 ;DATA CORRECT?
SCOP1 ;BR IF YES
INC R2 ;ALU ERROR
INC R0 ;SW09=1?
CMP #10,R0 ;NEXT DATA
BNE 1$ ;NEXT ADDRESS
SCOPE ;DONE YET?
.BYTE -1,-1,376,376,124,124,251,251 ;BR IF NO
;SCOPE THIS TEST
```

.EVEN

```
***** TEST 135 *****
*TEST OF PROGRAM CLOCK BIT
*DO A MASTER CLEAR, VERIFY THAT PROGRAM CLOCK IS SET
```

```

5701                                     ;*WRITE PROGRAM CLOCK BIT TO A ONE, VERIFY THAT IT CLEARS,
5702                                     ;*AND THEN SETS SOME TIME LATER
5703                                     ;*****
5704
5705                                     ; TEST 135
5706                                     ;-----
5707 033424 012737 000135 001226 TST135: MOV #135,TSTNO
5708 033432 012737 033604 001216 MOV #TST136,NEXT
5709
5710 033440 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
5711 033442 005037 001416 CLR TEMP ;MASTER CLEAR M8200-YC
5712 033446 005037 001246 CLR TEMP1 ;PREPARE FOR
5713 033452 012702 000011 MOV #11,R2 ;DELAY
5714 033456 012761 000020 000004 1$: MOV #20,4(R1) ;SAVE FOR TYPEOUT
5715 033464 152761 000002 000001 BISB #BIT1,1(R1) ;LOAD PORT 4
5716 033472 012761 121111 000006 MOV #121111,6(R1) ;SET ROMI
5717 033500 152761 000003 000001 BISB #BIT1:BIT0,1(R1) ;SEL6 + INSTRUCTION
5718 033506 012761 121224 000006 MOV #121224,6(R1) ;SET CLOCK BIT
5719 033514 152761 000003 000001 BISB #BIT1:BIT0,1(R1) ;LOAD NEXT INSTRUCTION
5720 033522 142761 000003 000001 BICB #BIT1:BIT0,1(R1) ;READ CLOCK BIT
5721 033530 016104 000004 MOV 4(R1),R4 ;CLEAR MAINT BITS
5722 033534 005005 CLR R5 ;PUT "FOUND" IN R4
5723 033536 120504 CMPB R5,R4 ;PUT "EXPECTED" IN R5
5724 033540 001401 BEQ 2$ ;IS PGM CLOCK CLEAR?
5725 033542 104016 HLT 16 ;ERROR, PGM CLOCK IS NOT CLEAR
5726 033544
5727 033544 104414 2$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5728 033546 121224 121224 PORT4+LUI1
5729 033550 122761 000020 000004 CMPB #20,4(R1) ;PORT4+LUI1
5730 033556 001411 BEQ 3$ ;IS PGM CLOCK SET?
5731 033560 005237 001416 INC TEMP ;BR IF YES
5732 033564 005537 001246 ADC TEMP1 ;INCREMENT DELAY
5733 033570 022737 000006 001246 CMP #6,TEMP1 ;INCREMENT DELAY
5734 033576 001362 BNE 2$ ;IS DELAY DONE
5735 033600 104016 HLT 16 ;BR IF NO
5736 033602 104400 3$: SCOPE ;ERROR PGM CLOCK NOT SET
5737
5738                                     ;***** TEST 136 *****
5739                                     ;*FORCE POWER FAIL TEST
5740                                     ;*SET FORCE POWER FAIL BIT VERIFY THAT PROCESSOR TRAPS TO 24
5741                                     ;*GOING DOWN AND COMING UP. VERIFY ALSO THAT BUS INIT WAS
5742                                     ;*BLOCKED FROM GETTING TO THE DMC DURING THE POWER FAIL
5743                                     ;*THIS TEST MAY HANG ON SOME PROCESSORS IF AN M9301 IS PRESENT.
5744                                     ;*TO AVOID HANGING SW02 (POWER ON REBOOT ENABLE) ON THE M9301
5745                                     ;*MUST BE IN THE OFF POSITION. THIS TEST WILL ALSO FAIL IF THE
5746                                     ;*CPU POWER FAIL VECTOR IS SET TO ANY LOCATION OTHER THAN 24.
5747                                     ;*IF THIS TEST HANGS OR FAILS DUE TO EITHER REASON ABOVE THE
5748                                     ;*FOLLOWING PATCH MAY BE INSTALLED TO SKIP THIS TEST:
5749                                     ;*
5750                                     ;*
5751                                     ;* LOC 33430 WAS 33600 SB 33772
5752                                     ;*****
5753
5754                                     ; TEST 136
    
```

```

5755
5756 033604 012737 000136 001226 TST136: MOV #136,TSTNO
5757 033612 012737 033776 001216 MOV #TST137,NEXT
5758
5759 033620 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
5760 033622 005037 001416 CLR TEMP ;MASTER CLEAR M8200-YC
5761 033626 013746 000024 MOV @#24,-(SP) ;PREPARE FOR DELAY
5762 033632 012737 033676 000024 MOV #15,@#24 ;STORE POWER FAIL ADDRESS
5763 033640 012761 000002 000004 MOV #24(R1) ;SET UP FOR FORCE POWER FAIL
5764 033646 012711 001000 MOV #BIT9,(R1) ;LOAD PORT4
5765 033652 012761 121111 000006 MOV #121111,6(R1) ;SET ROMI
5766 033660 012711 001400 MOV #BIT9:BIT8,(R1) ;LOAD INSTRUCTION
5767 033664 005237 001416 5$: INC TEMP ;CLOCK INSTRUCTION
5768 033670 001375 BNE 5$ ;WAIT FOR POWER FAIL
5769 033672 104017 HLT 17 ;BR IF DELAY NOT DONE
5770 033674 000426 BR 4$ ;ERROR, NO POWER FAIL
5771 033676 012737 033714 000024 1$: MOV #35,@#24 ;POWER UP ADDRESS
5772 033704 010637 033712 MOV SP,2$ ;STORE STACK
5773 033710 000000 HALT ;WAIT FOR POWER UP SEQUENCE
5774 033712 000000 2$: 0
5775 033714 013706 033712 3$: MOV 2$,SP ;RESTORE STACK
5776 033720 022626 POP2SP ;POP STACK TWICE
5777 033722 012637 000024 MOV (SP)+,@#24 ;RESTORE TRUE POWER FAIL ADDRESS
5778 033726 022737 005346 000024 CMP #.PFAIL,@#24 ;IS IT CORRECT?
5779 033734 001406 BEQ 4$ ;BR IF YES
5780 033736 104017 HLT 17 ;ERROR, STACK IS INCORRECT
5781 033740 012737 005346 000024 MOV #.PFAIL,@#24 ;RESTORE TRUE POWER FAIL ADDRESS
5782 033746 012706 001200 MOV #STACK,SP ;RESTORE STACK
5783 033752 012711 003000 4$: MOV #BIT9:BIT10,(R1) ;SEL6 = MAINT IR
5784 033756 012705 121111 MOV #121111,R5 ;R5 = EXPECTED
5785 033762 016104 000004 MOV 4(R1),R4 ;R4 = FOUND
5786 033766 020504 CMP R5,R4 ;MAINT IR SHOULD = 12111
5787 033770 001401 BEQ +4 ;BR IF OK
5788 033772 104025 HLT 25 ;IF = 0 THEN BUS INIT WAS
5789 ;NOT BLOCKED FROM CLEARING
5790 ;THE DMC-11
5791 033774 104400 SCOPE ;SCOPE THIS TEST
5792
5793
5794
5795
5796
5797
5798
5799
5800
5801
5802
5803
5804
5805 033776 012737 000137 001226 TST137: MOV #137,TSTNO
5806 034004 012737 003374 001216 MOV #.EOP,NEXT
5807
5808 034012 104412 MSTCLR ;R1 CONTAINS BASE M8200-YC ADDRESS
;MASTER CLEAR M8200-YC

```

```

***** TEST 137 *****
;MICRO-PROCESSOR NOISE TEST
;WRITE ALL ZERO'S THEN ALL ONE'S THEN A DATA PATTERN
;TO THE IBUS* AND IBUS REGISTERS AND TO THE SP AND MAIN MEM
;THEN GO BACK AND READ THE DATA PATERNS TO VERIFY THAT
;READING AND WRITING OF OTHER LOCATIONS AND REGISTERS
;DID NOT CHANGE THE DATA.
*****

```

TEST 137

5809	034014	005002				CLR	R2	;R2 IS INDEX REGISTER
5810	034016	042737	000017	034042	1\$:	BIC	#17,2\$;CLEAR ADDRESS FIELD
5811	034024	156237	034636	034042		BISB	30\$(R2),2\$;ADD IBUS* REG ADDRESS TO INSTRUCTION
5812	034032	116261	034644	000004		MOVB	31\$(R2),4(R1)	;LOAD PORT4
5813	034040	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5814	034042	121100			2\$:	121100		;WRITE IBUS* REGISTER
5815	034044	005202				INC	R2	;INC INDEX REGISTER
5816	034046	022702	000005			CMP	#5,R2	;DONE YET?
5817	034052	001361				BNE	1\$;BR IF NO
5818	034054	005002				CLR	R2	;R2 IS IBUS REGISTER ADDRESS
5819	034056	042737	000017	034122	3\$:	BIC	#17,4\$;CLEAR ADDRESS FIELD OF INSTRUCTIONS
5820	034064	042737	000017	034134		BIC	#17,5\$	
5821	034072	042737	000017	034144		BIC	#17,6\$	
5822	034100	050237	034122			BIS	R2,4\$;ADD IBUS REG ADDRESS TO INSTRUCTION
5823	034104	050237	034134			BIS	R2,5\$	
5824	034110	050237	034144			BIS	R2,6\$	
5825	034114	105061	000004			CLRB	4(R1)	;CLEAR PORT4
5826	034120	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5827	034122	122100			4\$:	122100		;WRITE 0 TO IBUS REG
5828	034124	112761	000377	000004		MOVB	#377,4(R1)	;LOAD PORT4
5829	034132	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5830	034134	122100			5\$:	122100		;WRITE ALL ONES TO IBUS REG
5831	034136	110261	000004			MOVB	R2,4(R1)	;LOAD PORT4
5832	034142	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5833	034144	122100			6\$:	122100		;WRITE ITS OWN ADDRESS TO IBUS REG
5834	034146	005202				INC	R2	;NEXT ADDRESS
5835	034150	022702	000010			CMP	#10,R2	;DONE YET?
5836	034154	001340				BNE	3\$;BR IF NO
5837	034156	005002				CLR	R2	;START AT SP ADDRESS 0
5838	034160	042737	000017	034224	7\$:	BIC	#17,8\$;CLEAR ADDRESS FIELD
5839	034166	042737	000017	034236		BIC	#17,9\$	
5840	034174	042737	000017	034246		BIC	#17,10\$	
5841	034202	050237	034224			BIS	R2,8\$;ADD ADDRESS TO INSTRUCTION
5842	034206	050237	034236			BIS	R2,9\$	
5843	034212	050237	034246			BIS	R2,10\$	
5844	034216	105061	000004			CLRB	4(R1)	;CLEAR PORT4
5845	034222	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5846	034224	123100			8\$:	123100		;WRITE ZERO TO SP
5847	034226	112761	000377	000004		MOVB	#377,4(R1)	;LOAD PORT4
5848	034234	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5849	034236	123100			9\$:	123100		;WRITE ALL ONES TO SP
5850	034240	110261	000004			MOVB	R2,4(R1)	;LOAD PORT4
5851	034244	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5852	034246	123100			10\$:	123100		;WRITE SP ADDRESS TO ITSELF
5853	034250	005202				INC	R2	;NEXT SP ADDRESS
5854	034252	022702	000020			CMP	#20,R2	;DONE YET?
5855	034256	001340				BNE	7\$;BR IF NO
5856	034260	005002				CLR	R2	;R2 = MAIN MEM ADDRESS
5857	034262	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5858	034264	010000				010000		;MAR ← 0
5859	034266	105061	000004		11\$:	CLRB	4(R1)	;CLEAR PORT4
5860	034272	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5861	034274	122500				122500		;WRITE ZEROS TO MEM
5862	034276	112761	000377	000004		MOVB	#377,4(R1)	;LOAD PORT4

M10

DRLPL MACY11 27(654) 13-DEC-77 11:41 PAGE 113
 DRLPL.P11 MB200-YC ALU TESTS

SEQ 0129

5863	034304	104414			ROMCLK				:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5864	034306	122500			122500				:WRITE ONES TO MEM
5865	034310	110261	000004		MOVW	R2,4(R1)			:LOAD PORT4
5866	034314	104414			ROMCLK				:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5867	034316	136500			136500				:WRITE TO MEM IT OWN ADDRESS
5868	034320	005202			INC	R2			:NEXT MEM ADDRESS
5869	034322	022702	000400		CMP	#400,R2			:DONE YET?
5870	034326	001357			BNE	11\$:BR IF NO
5871									
5872									:NOW GO BACK AND READ EVERYTHING
5873									
5874	034330	104414			ROMCLK				:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5875	034332	010000			010000				:MAR←0
5876	034334	032737	100000	001366	BIT	#BIT15,STAT1			:DMC?
5877	034342	001402			BEQ	.+6			:BR IF YES
5878	034344	104414			ROMCLK				:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5879	034346	004000			4000				:MAR HI ← 0 (KMC ONLY)
5880	034350	005000			CLR	R0			:R0 IS INDEX REGISTER
5881	034352	042737	000360	034410	12\$:	BIC	#360,13\$:CLEAR ADDRESS FIELD
5882	034360	116002	034636		MOVW	30\$(R0),R2			:R2 = IBUS* ADDRESS
5883	034364	010203			MOV	R2,R3			:PUT IBUS* ADDRESS IN R3
5884	034366	006303			ASL	R3			:SHIFT ADDRESS TO BITS 4-7
5885	034370	006303			ASL	R3			
5886	034372	006303			ASL	R3			
5887	034374	006303			ASL	R3			
5888	034376	050337	034410		BIS	R3,13\$:ADD ADDRESS TO INSTRUCTION
5889	034402	116005	034644		MOVW	31\$(R0),R5			:R5 = "EXPECTED"
5890	034406	104414			ROMCLK				:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5891	034410	121004			121004				:PORT4 ← IBUS* REGISTER
5892	034412	016104	000004	13\$:	MOV	4(R1),R4			:R4 = "FOUND"
5893	034416	120504			CMPB	R5,R4			:IBUS* CONTENTS OK?
5894	034420	001401			BEQ	.+4			:BR IF YES
5895	034422	104004			HLT	4			:IBUS* DATA ERROR
5896	034424	005200			INC	R0			:INC COUNTER
5897	034426	022700	000005		CMP	#5,R0			:DONE YET?
5898	034432	001347			BNE	12\$:BR IF NO
5899	034434	005002			CLR	R2			:R2 = IBUS REG ADDRESS
5900	034436	042737	000360	034466	14\$:	BIC	#360,15\$:CLEAR ADDRESS FIELD OF INSTRUCTION
5901	034444	010203			MOV	R2,R3			:R3 = IBUS ADDRESS
5902	034446	006303			ASL	R3			:SHIFT ADDRESS TO BITS 4-7
5903	034450	006303			ASL	R3			
5904	034452	006303			ASL	R3			
5905	034454	006303			ASL	R3			
5906	034456	050337	034466		BIS	R3,15\$:ADD ADDRESS TO INSTRUCTION
5907	034462	010205			MOV	R2,R5			:R5 = "EXPECTED"
5908	034464	104414			ROMCLK				:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5909	034466	021004			021004				:PORT4 ← IBUS REG
5910	034470	016104	000004	15\$:	MOV	4(R1),R4			:R4 = "FOUND"
5911	034474	120504			CMPB	R5,R4			:IBUS CONTENTS OK?
5912	034476	001401			BEQ	.+4			:BR IF YES
5913	034500	104005			HLT	5			:IBUS DATA ERROR
5914	034502	005202			INC	R2			:NEXT IBUS REGISTER
5915	034504	022702	000010		CMP	#10,R2			:DONE YET?
5916	034510	001352			BNE	14\$:BR IF NO

```

5917 034512 005002          CLR      R2          ;R2 = SP ADDRESS
5918 034514 042737 000017 034530 16$: BIC      #17,17$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
5919 034522 050237 034530          BIS      R2,17$    ;ADD ADDRESS TO INSTRUCTION
5920 034526 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5921 034530 040600          040600          BR      + SP
5922 034532 010205          MOV      R2,R5     ;R5 = "EXPECTED"
5923 034534 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5924 034536 061224          061224          PORT4 + BR
5925 034540 016104 000004          MOV      4(R1),R4  ;R4 = "FOUND"
5926 034544 120504          CMPB    R5,R4     ;SP CONTENTS OK?
5927 034546 001401          BEQ     +4        ;BR IF YES
5928 034550 104007          HLT     ?        ;SP DATA ERROR
5929 034552 005202          INC     R2       ;NEXT SP LOCATION
5930 034554 022702 000020          CMP     #20,R2   ;DONE YET?
5931 034560 001355          BNE    16$      ;BR IF NO
5932 034562 005002          CLR     R2      ;R2 = MEMORY ADDRESS
5933 034564 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5934 034566 010000          010000          MAR + 0
5935 034570 032737 100000 001366          BIT     #BIT15,STAT1 ;DMC?
5936 034576 001402          BEQ     +6      ;BR IF YES
5937 034600 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5938 034602 004000          4000          MAR HI + 0 (KMC ONLY)
5939 034604 010205          MOV     R2,R5   ;R5 = "EXPECTED"
5940 034606 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5941 034610 055224          055224          PORT4 + MAIN MEM
5942 034612 016104 000004          MOV     4(R1),R4  ;R4 = "FOUND"
5943 034616 120504          CMPB    R5,R4   ;MAIN MEM CONTENTS OK?
5944 034620 001401          BEQ     +4      ;BR IF YES
5945 034622 104013          HLT     13     ;MAIN MEM DATA ERROR
5946 034624 005202          INC     R2     ;NEXT MEM ADDRESS
5947 034626 022702 000400          CMP     #400,R2 ;DONE YET?
5948 034632 001364          BNE    18$    ;BR IF NO
5949 034634 104400          SCOPE          ;SCOPE THIS TEST
5950 034636 000          002          003 30$: .BYTE 0,2,3,5,10
5951 034641 005          010
5952          034644          .EVEN
5953 034644 001          003          004 31$: .BYTE 1,3,4,6,10
5954 034647 006          010
5955          034652          .EVEN
5956
5957
5958          ; SUBROUTINES
5959          ;-----
5960
5961          034652          SETVEC: ;THIS SUBROUTINE LOADS THE VECTORS AND VECTOR LEVELS
5962
5963
5964 034652 012577 144516          MOV     (R5)+,@DMRVEC ;LOAD BASE VECTOR
5965 034656 012577 144516          MOV     (R5)+,@DMTVEC ;LOAD VECTOR + 2
5966 034662 112577 144510          MOVB   (R5)+,@DMRLVL ;LOAD VECTOR + 4
5967 034666 112577 144510          MOVB   (R5)+,@DMTLVL ;LOAD VECTOR + 6
5968 034672 000205          RTS     R5      ;RETURN
5969
5970

```

```

5971 034674      NPRSET:
5972              ; THIS SUBROUTINE LOADS IBUS REGISTERS 0-7
5973              ; WITH NPR INFORMATION (INBA, OUTBA, OUT DATA)
5974
5975 034674 010246      MOV      R2, -(SP)      ; SAVE R2
5976 034676 005002      CLR      R2            ; START AT IBUS REG 0
5977 034700 112561 000004      MOV      (R5)+, 4(R1)    ; LOAD PORT4
5978 034704 042737 000017 034720 1$:      BIC      #17, 2$      ; CLEAR ADDRESS FIELD OF INSTRUCTION
5979 034712 050237 034720      BIS      R2, 2$      ; ADD ADDRESS TO INSTRUCTION
5980 034716 104414      ROMCLK   ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5981 034720 122100      2$:      122100      ; MOVE PORT4 TO IBUS REG
5982 034722 005202      INC      R2            ; NEXT ADDRESS
5983 034724 022702 000010      CMP      #10, R2      ; ALL DONE?
5984 034730 001363      BNE     1$            ; BR IF NO
5985 034732 012602      MOV      (SP)+, R2    ; RESTORE R2
5986 034734 000205      RTS      R5          ; RETURN
5987
5988
5989 034736      MEMLD:
5990              ; THIS SUBROUTINE LOADS THE FIRST 8 LOCATIONS OF MAIN
5991              ; MEMORY WITH THIS DATA: 0, -1, , 0, -1, 125, 252, 125, 252
5992
5993 034736 013605      MOV      @ (SP)+, R5    ; PUT POINTER TO DATA IN R5
5994 034740 062746 000002      ADD      #2, -(SP)    ; ADJUST STACK
5995 034744 012704 000010      MOV      #10, R4     ; DO 8 LOADS
5996 034750 104414      ROMCLK   ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5997 034752 010000      010000      ; MAR < 0
5998 034754 112577 144432      1$:      MOV      (R5)+, @DMP04 ; LOAD PORT4
5999 034760 104414      ROMCLK   ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6000 034762 136500      136500      ; MOV DATA TO MEM, AUTO INC MAR
6001 034764 005304      DEC      R4          ; DECREMENT COUNT
6002 034766 001372      BNE     1$            ; BR IF NOT DONE
6003 034770 000207      RTS      PC          ; RETURN
6004
6005
6006 034772      SPLD:
6007              ; THIS SUBROUTINE LOADS THE FIRST 8 SCRATCH PAD
6008              ; LOCATIONS WITH: 0, 0, -1, -1, 125, 125, 252, 252
6009
6010 034772 013605      MOV      @ (SP)+, R5    ; PUT POINTER TO DATA IN R5
6011 034774 062746 000002      ADD      #2, -(SP)    ; ADJUST STACK
6012 035000 005004      CLR      R4          ; START AT SP ADDRESS 0
6013 035002 112577 144404      1$:      MOV      (R5)+, @DMP04 ; LOAD PORT4 WITH DATA
6014 035006 042737 000017 035022      BIC      #17, 2$      ; CLEAR ADDRESS FIELD OF INSTRUCTION
6015 035014 050437 035022      BIS      R4, 2$      ; ADD ADDRESS TO INSTRUCTION
6016 035020 104414      ROMCLK   ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6017 035022 123100      2$:      123100      ; MOVE DATA TO SP
6018 035024 005204      INC      R4          ; INCREMENT COUNT
6019 035026 022704 000010      CMP      #10, R4     ; DONE YET?
6020 035032 001363      BNE     1$            ; BR IF NO
6021 035034 000207      RTS      PC          ; RETURN
6022
6023
6024 035036      CLRC:

```

```

6025 ;THIS SUBROUTINE CLEARS THE MICRO PROCESSOR C BIT
6026
6027 035036 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6028 035040 010000 ;MAR+0
6029 035042 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6030 035044 040400 040400! <0*20> ;CLEAR C BIT
6031 035046 000207 RTS PC ;RETURN
6032
6033
6034 035050 SETC: ;THIS SUBROUTINE SETS THE MICRO PROCESSOR C BIT
6035
6036
6037 035050 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6038 035052 010003 ;MAR+3
6039 035054 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6040 035056 040403 040403! <0*20> ;SET C BIT
6041 035060 000207 RTS PC ;RETURN
6042
6043
6044 035062 000 377 000 MEMDAT: .BYTE 0,-1,0,-1,125,252,125,252
6045 035065 377 125 252
6046 035070 125 252
6047 035072 000 000 377 SPDAT: .BYTE 0,0,-1,-1,125,125,252,252
6048 035075 377 125 125
6049 035100 252 252
6050
6051
    
```

```

        .EVEN
        MESWCH: .ASCII <200># NOTE:#
                .ASCII <200>#FOR THIS PROGRAM TO RUN PROPERLY, SWITCH#
                .ASCII <200>#7, OF THE VECTOR ADDRESS SWITCH PACK (E76),#
                .ASCII <200>#MUST BE ON. (M8200-YC BOARD)#<200>
                .ASCII <377>/UNIBUS REGISTER ADDRESSING TIME-OUT/
                .ASCII <377>/UNIBUS REGISTER WRITE/READ TEST/
                .ASCII <377>/MICRO PROCESSOR TEST/
                .ASCII <377>/MICRO PROCESSOR WRITE/READ TEST/
                .ASCII <377>/BR REGISTER TEST/
                .ASCII <377>/SCRATCH PAD TEST/
                .ASCII <377>/DEVICE FAILED TO INTERRUPT/
                .ASCII <377>/DEVICE INTERRUPTED TO WRONG VECTOR/
                .ASCII <377>/NPR TEST/
                .ASCII <377>/MAIN MEMORY TEST/
                .ASCII <377>/MAR TEST/
                .ASCII <377>/ALU TEST/
                .ASCII <377>/PROGRAM CLOCK TEST/
                .ASCII <377>/FORCE POWER FAIL ERROR/
                .ASCII <377>/UNEXPECTED INTERRUPT/
                .ASCII <377>/M8200-YC CONFIGURATION ERROR/
                .ASCII <377>/MAINTENANCE INSTRUCTION REGISTER TEST/
                .ASCII <377>/POWER FAIL INITIALIZE FAILURE/

        DH1: .ASCII <377>/REGISTER TRAPPED FROM/
        DH2: .ASCII <377>/EXPECTED FOUND REGISTER/
        DH3: .ASCII <377>/EXPECTED FOUND/
        DH4: .ASCII <377>/EXPECTED FOUND IBUS* REGISTER/
        DH5: .ASCII <377>/EXPECTED FOUND IBUS REGISTER/
    
```

036413	377	054105	042520	DH6: .EVEN	.ASCIZ	<377>/EXPECTED	FOUND	ADDRESS/
036452	000002			DT1:	2			
036454	006	015			.BYTE		6,15	
036456	001262				SAVR1			
036460	006	002			.BYTE		6,2	
036462	001264				SAVR2			
036464	000003			DT2:	3			
036466	006	004			.BYTE		6,4	
036470	001272				SAVR5			
036472	006	004			.BYTE		6,4	
036474	001270				SAVR4			
036476	006	002			.BYTE		6,2	
036500	001262				SAVR1			
036502	000002			DT3:	2			
036504	006	004			.BYTE		6,4	
036506	001272				SAVR5			
036510	006	002			.BYTE		6,2	
036512	001270				SAVR4			
036514	000003			DT4:	3			
036516	003	007			.BYTE		3,7	
036520	001272				SAVR5			
036522	003	011			.BYTE		3,11	
036524	001270				SAVR4			
036526	002	002			.BYTE		2,2	
036530	001264				SAVR2			
036532	000003			DT5:	3			
036534	003	007			.BYTE		3,7	
036536	001272				SAVR5			
036540	003	007			.BYTE		3,7	
036542	001270				SAVR4			
036544	006	002			.BYTE		6,2	
036546	001264				SAVR2			
036550	000002			DT6:	2			
036552	003	007			.BYTE		3,7	
036554	001272				SAVR5			
036556	003	002			.BYTE		3,2	
036560	001270				SAVR4			
036562	000002			DT7:	2			
036564	006	004			.BYTE		6,4	
036566	001262				SAVR1			
036570	006	002			.BYTE		6,2	
036572	001404				DMC3R			
036574				.ERRTAB:				
036574	000000				0			
036576	000000				0			
036600	000000				0			
036602	035301				EM1			
036604	036170				DH1	;HLT	1	
036606	036452				DT1			
036610	035346				EM2			
036612	036231				DH2	;HLT	2	
036614	036464				DT2			

036616	035407	EM3		
036620	036267	DH3	;HLT	3
036622	036502	DT3		
036624	035435	EM4		
036626	036310	DH4	;HLT	4
036630	036514	DT4		
036632	035435	EM4		
036634	036352	DH5	;HLT	5
036636	036514	DT4		
036640	035476	EM5		
036642	036267	DH3	;HLT	6
036644	036550	DT6		
036646	035520	EM6		
036650	036413	DH6	;HLT	7
036652	036532	DT5		
036654	035542	EM7		
036656	000000	0	;HLT	10
036660	000000	0		
036662	035576	EM10		
036664	000000	0	;HLT	11
036666	000000	0		
036670	035642	EM11		
036672	036267	DH3	;HLT	12
036674	036502	DT3		
036676	035654	EM12		
036700	036413	DH6	;HLT	13
036702	036532	DT5		
036704	035676	EM13		
036706	036413	DH6	;HLT	14
036710	036532	DT5		
036712	035710	EM14		
036714	036267	DH3	;HLT	15
036716	036550	DT6		
036720	035722	EM15		
036722	036310	DH4	;HLT	16
036724	036514	DT4		
036726	035746	EM16		
036730	000000	0	;HLT	17
036732	000000	0		
036734	035776	EM17		
036736	000000	0	;HLT	20
036740	000000	0		
036742	035642	EM11		
036744	036413	DH6	;HLT	21
036746	036532	DT5		
036750	036024	EM20		
036752	036267	DH3	;HLT	22
036754	036562	DT7		
036756	036062	EM21		
036760	036267	DH3	;HLT	23
036762	036502	DT3		
036764	036024	EM20		
036766	000000	0	;HLT	24
036770	000000	0		

DRLPL MACY11 27(654) 13-DEC-77 11:41 PAGE 119
DRLPL.P11 M8200-YC ALU TESTS

SEQ 0135

036772 036131
036774 036267
036776 036502

EM22
DH3 ;HLT 25
DT3

037000 000001

CORMAX:
.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	CHRCNT	004722	CKSWR	007646	CKSWR1	007726
CKSWR2	007740	CKSWR3	007744	CKSWR4	007750	CKSWR5	010054
CLKX	001242	CLRC	035036	CNERR	007323	CNT.MA	001702
CNVRT =	104411	CONERR	007244	CONN	007135	CONTAB	003006
CONVRT=	104410	CORMAX	037000	CRAM	006627	CREAM	001320
CSR	006531	CSRMAP	010554	CYCLE	010120	DATABP	005226
DATACL=	104415	DATAHD	005214	DELAY =	104413	DEVADR	004400
DEVTAB	003020	DH1	036170	DH2	036231	DH3	036267
DH4	036310	DH5	036352	DH6	036413	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	036452	DT2	036464	DT3	036502	DT4	036514
DT5	036532	DT6	036550	DT7	036562	EM1	035301
EM10	035576	EM11	035642	EM12	035654	EM13	035676
EM14	035710	EM15	035722	EM16	035746	EM17	035776
EM2	035346	EM20	036024	EM21	036062	EM22	036131
EM3	035407	EM4	035435	EMS	035476	EM6	035520
EM7	035542	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	ERTAB0	005332	EXIT =	000205
EXITER	005262	FLOAT	002546	FY	002576	HALTS	005232
HILIM	004376	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
LPCNT	001224	LSTERR	001234	MASKX	001244	MASTEK	006152
MCRLF	005702	MCSRX	006102	MDATA	007604	MEMDAT	035062

MEMLD	034736	MEMLIM	001304	MEPASS	005743	MERRPC	006232
MERRX	006127	MERR2	005770	MERR3	006015	ME SWCH	035102
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	NPRSET	034674	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	POPPO =	012600	POP1SP=	005726	POP2SP=	022626
PRI0	006570	PS	= 177776	PUSHRO=	010046	PUSHIS=	005746
PUSH2S=	024646	QV.FLG	001327	RESREG	005230	RESTAR	005360
RESTRT	003544	RES05 =	104407	RETURN	001214	ROMCLK=	104414
ROM1	036450	RUN	001316	R0	=%000000	R1	=%000001
R2	=%000002	R3	=%000003	R4	=%000004	R5	=%000005
SAVACT	001312	SAVNUM	001314	SAVPC	001276	SAVRO	001260
SAVR1	001262	SAVR2	001264	SAVR3	001266	SAVR4	001270
SAVR5	001272	SAVSP	001274	SAV05 =	104406	SCOPE =	104400
SCOPI =	104401	SETC	035050	SETVEC	034652	SKIP	002642
SOFTSW	010112	SP	=%000006	SPACNT=	004723	SPDAT	035072
SPEED	007371	SPLD	034772	STACK =	001200	STAT	001240
STAT1	001366	STAT2	001370	STAT3	001372	STRTSW	001236
SV05	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	TDATA	024614	TEMP	001416
TEMP1	001246	TEMP2	001250	TEMP3	001252	TEMP4	001254
TEMP5	001256	TIMER =	104416	TKCSR	001204	TKDBR	001206
TLAST =	033776	TPCSR	001210	TPDBR	001212	TRPOK	004740
TSTNO	001226	TST1	012372	TST10	013250	TST100	025330
TST101	025504	TST102	025660	TST103	026034	TST104	026210
TST105	026364	TST106	026540	TST107	026714	TST11	013346
TST110	027070	TST111	027244	TST112	027420	TST113	027574
TST114	027750	TST115	030124	TST116	030300	TST117	030454
TST12	013444	TST120	030630	TST121	031004	TST122	031160
TST123	031334	TST124	031510	TST125	031664	TST126	032040
TST127	032214	TST13	013542	TST130	032370	TST131	032544
TST132	032720	TST133	033074	TST134	033250	TST135	033424
TST136	033604	TST137	033776	TST14	013640	TST15	013736
TST16	014034	TST17	014132	TST2	012500	TST20	014230
TST21	014326	TST22	014424	TST23	014522	TST24	014620
TST25	014744	TST26	015070	TST27	015220	TST3	012530
TST30	015360	TST31	015522	TST32	015606	TST33	016006
TST34	016206	TST35	016362	TST36	016536	TST37	016736
TST4	012660	TST40	017156	TST41	017332	TST42	017506
TST43	017662	TST44	020036	TST45	020212	TST46	020366
TST47	020542	TST5	012756	TST50	020716	TST51	021144
TST52	021314	TST53	021562	TST54	022004	TST55	022100

TST56	022172	TST57	022312	TST6	013054	TST60	022456
TST61	022562	TST62	022676	TST63	023000	TST64	023136
TST65	023262	TST66	023372	TST67	023500	TST7	013152
TST70	023676	TST71	024024	TST72	024156	TST73	024356
TST74	024512	TST75	024624	TST76	025000	TST77	025154
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
SCRAP =	177777	SENDAD	003532	\$N =	000137	\$S =	000141
\$Y =	000017	.BEGIN	003162	.CNVRT	004502	.CONVR	004476
.DATAC	005562	.DELAY	005446	.EOP	003374	.ERRTA	036574
.HLT	004760	.INSTE	004164	.INSTR	004060	.INST1	004100
.MSG	004102	.MSTCL	005476	.PARAM	004204	.PFAIL	005346
.RESOS	004444	.ROMCL	005514	.SAVOS	004404	.SCOPE	003606
.SCOPI	003746	.START	002002	.TIMER	005626	.TRPSR	004726
.TRPTA	001330	.TYPE	003776	.	= 037000		

ERRORS DETECTED: 0

J11

DRLPL MACY11 27(654) 13-DEC-77 11:41 PAGE 123
DRLPL.P11

SEQ 0139

*DRLPL,DRLPL/SOL=DRLPL.MAC,DRLPL
RUN-TIME: 20 26 0 SECONDS
CORE USED: 28K