

DMC11

DMC11 BSC W/R UPROC
CZDMCD0

AH-8545D-MC

1 OF 1 OCT 1985

COPYRIGHT © 1976-85

digital

MADE IN USA

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	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

.REM 6

IDENTIFICATION

PRODUCT CODE: AC-8544D-MC
PRODUCT NAME: CZDMCDO DMC11 BSC W/R UPROC
DATE: 17-JUN-1985
MAINTAINER: MK NAC SOFTWARE ENGINEERING
AUTHOR: BRUCE LUHRS

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.

NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES

COPYRIGHT (C) 1976, 1978, 1985 BY DIGITAL EQUIPMENT CORPORATION

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

1. ABSTRACT

THE FUNCTION OF THE DMC11 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 DMC11 ARE CORRECT IN ITS ENVIRONMENT.

PARAMETERS MUST BE SET UP TO ALERT THE DIAGNOSTICS TO THE DMC11 CONFIGURATION. THESE PARAMETERS ARE CONTAINED IN THE STATUS TABLE AND ARE GENERATED IN TWO WAYS: 1) MANUAL INPUT - THE OPERATOR ANSWERS QUESTIONS. 2) AUTOSIZING - THE PROGRAM DETERMINES THE PARAMETERS AUTOMATICALLY.

CZDMC TESTS THE DMC11 MICRO-PROCESSOR (M8200-YA OR M8200-YB). IT PERFORMS WRITE/READ TESTS ON THE DMC UNIBUS REGISTERS, CHECKS THE MICRO-PROCESSOR OPERATION, CHECKS OUT MAIN MEMORY, SCRATCH PAD MEMORY, THE ALU FUNCTIONS AS WELL AS INTERRUPTS AND NPR OPERATION. CZDMC PERFORMS NO TESTS ON THE LINE UNIT OR ANY CROM DEPENDENT TESTS. IT DOES NOT REQUIRE A LINE UNIT TO RUN. NOTE: THIS DIAGNOSTIC WILL RUN ON A KMC11 (M8204), 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.

CURRENTLY THERE ARE FIVE OFF LINE DIAGNOSTICS THAT ARE TO BE RUN IN SEQUENCE TO INSURE THAT IF AN ERROR SHOULD OCCUR IT WILL BE DETECTED AT AN EARLY STAGE.

NOTE: ADDITIONAL DIAGNOSTICS MAY BE ADDED IN THE FUTURE.

THE FIVE DIAGNOSTICS ARE:

1. CZDMC [REV] BASIC W/R AND MICRO-PROCESSOR TESTS
2. CZDME [REV] DDCMP MODE LINE UNIT TESTS
3. DZDMF [REV] BITSTUFF LINE UNIT TESTS
4. DZDMG [REV] JUMP AND CROM TESTS
5. DZDMH [REV] FREE-RUNNING TESTS (HEAT TEST TAPE)

2. REQUIREMENTS

2.1 EQUIPMENT

ANY PDP11 FAMILY CPU (EXCEPT AN LSI-11) WITH MINIMUM 8K MEMORY ASR 33 (OR EQUIVALENT)
DMC11-AR (M8200-YA) OR A DMC11-AL (M8200-YB)

2.2 STORAGE

PROGRAM WILL USE ALL 8K OF MEMORY EXCEPT WHERE ABL AND BOOTSTRAP LOADER RESIDE. LOCATIONS 1500 THRU 1640; CONTAIN THE "STATUS TABLE" INFORMATION WHICH IS GENERATED AT START OF DIAGNOSTICS BY MANUAL INPUT (QUESTIONS) OR AUTOMATICALLY (AUTO-SIZING). THIS AREA IS AN OVERLAY AREA AND SHOULD NOT BE ALTERED BY THE OPERATOR.

3. LOADING PROCEDURE

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 DMC11 DIAGNOSTIC.
- D. DEPRESS 'START KEY' AND RELEASE. THE PROGRAM WILL TYPE MAINDEC NAME AND PROGRAM NAME (IF THIS WAS THE FIRST START UP OF THE PROGRAM) AND ALSO THE FOLLOWING:

MAP OF DMC11 STATUS

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

THE PROGRAM WILL TYPE 'R' AND PROCEED TO RUN THE DIAGNOSTIC. THE ABOVE IS ONLY AN EXAMPLE. THIS WOULD INDICATE THE STATUS TABLE STARTING AT ADD. 1500 IN THE PROGRAM. IN THIS EXAMPLE THE TABLE CONTAINS THE INFORMATION AND STATUS OF TWO DMC11'S. THE STATUS TABLE MUST BE VERIFIED BY THE USER IF AUTO SIZING IS DONE. FOR INFORMATION OF STATUS TABLE SEE SECTION 8.4 FOR HELP.

IF THE DIAGNOSTIC WAS STARTED WITH SW00=1 INDICATING MANUAL PARAMETER INPUT THEN THE FOLLOWING SHOWS AN EXAMPLE OF THE QUESTIONS ASKED AND SOME EXAMPLE ANSWERS:

HOW MANY DMC11'S TO BE TESTED?1

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

FOLLOWING THE QUESTIONS THE STATUS MAP IS PRINTED OUT AS DESCRIBED ABOVE, THE INFORMATION IN THE MAP REFLECTS THE ANSWERS TO THE QUESTIONS. IF THE DIAGNOSTIC WAS STARTED WITH SW00=0 AND SW07=0 (AUTO-SIZING) THEN NO QUESTIONS ARE ASKED AND ONLY THE STATUS-MAP IS PRINTED OUT. IF AUTO-SIZING IS USED THE STATUS INFORMATION MUST BE VERIFIED TO BE CORRECT (MATCH THE HARDWARE). IF IT DOES NOT MATCH THE HARDWARE THE DIAGNOSTIC MUST BE RESTARTED WITH SW00=1 AND THE QUESTIONS ANSWERED.

58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85

4.1 CONTROL SWITCH SETTINGS

SW 15 SET: HALT ON ERROR
SW 14 SET: LOOP ON CURRENT TEST
SW 13 SET: INHIBIT ERROR PRINT OUT
SW 12 SET: INHIBIT TYPE OUT/ABELL ON ERROR.
SW 11 SET: INHIBIT ITERATIONS. (QUICK PASS)
SW 10 SET: ESCAPE TO NEXT TEST ON ERROR
SW 09 SET: LOOP WITH CURRENT DATA
SW 08 SET: CATCH ERROR AND LOOP ON IT
SW 07 SET: USE PREVIOUS STATUS TABLE.
SW 06 SET: HALT IN ROMCLK ROUTINE BEFORE CLOCKING
MICRO-PROCESSOR
SW 05 SET: RESERVED
SW 04 SET: RESERVED
SW 03 SET: RESELECT DMC11'S DESIRED ACTIVE
SW 02 SET: LOCK ON SELECTED TEST
SW 01 SET: RESTART PROGRAM AT SELECTED TEST
SW 00 SET: BUILD NEW STATUS TABLE FROM QUESTIONS. (IF SW07=0
AND SW00=0 A NEW STATUS TABLE IS BUILT BY
AUTO-SIZING)

SWITCH 06 AND 08-15 ARE DYNAMIC AND CAN BE CHANGED AS NEEDED
WHILE THE DIAGNOSTIC IS RUNNING. SWITCHES 00-03 AND SWITCH 07
ARE STATIC, AND ARE USED ONLY ON STARTING OR RESTARTING THE
DIAGNOSTIC.

4.1.2 SWITCH REGISTER OPTIONS (AT START UP)

SW 01 RESTART PROGRAM AT SELECTED TEST. IT IS STRONGLY SUGGESTED THAT AT LEAST ONE PASS HAS BEEN MADE BEFORE TRYING TO SELECT A TEST, THE REASON BEING IS THAT THE PROGRAM HAS TO CLEAR AREAS AND SET UP PARAMETERS. WHEN THIS SWITCH IS USED THE DIAGNOSTIC WILL ASK TEST NO.? ANSWER BY TYPING THE NUMBER OF THE TEST DESIRED AND CARRIGE RETURN TO BEGIN EXECUTION AT THE SELECTED TEST.

SW 02 LOCK ON SELECTED TEST. THIS SWITCH WHEN USED WITH SW01 WILL CAUSE THE PROGRAM TO CONSTANTLY LOOP ON THE SELECTED TEST. HITTING ANY KEY ON THE CONSOLE WILL LET IT ADVANCE TO THE NEXT TEST AND LOOP UNTIL A KEY IS HIT AGAIN. IF SW02=0 WHEN SW01 IS USED. THE PROGRAM WILL BEGIN AT THE SELECTED TEST AND CONTINUE NORMAL OPERATIONS.

SW 03 RESELECT DMC11'S DESIRED ACTIVE. PLEASE NOTE THAT A MESSAGE IS TYPED OUT FOR SETTING THE SWITCH REGISTER EQUAL TO DMC11'S ACTIVE. THIS MEANS IF THE SYSTEM HAS FOUR DMC11S; BITS 00,01,02,03 WILL BE SET IN LOC 'DMACTV' FROM THE SWITCH REGISTER. USING THIS SWITCH(SW00) ALTERS THAT LOCATION; THEREFORE IF FOUR DMC11S ARE IN THE SYSTEM ***DO NOT*** SET SWITCHS GREATER THAN SW 03 IN THE UP POSITION. THIS WOULD BE A FATAL ERROR. DO NOT SELECT MORE ACTIVE DMC11S THAN THERE IS INFORMATION ON IN THE STATUS TABLE.

METHOD: A: LOAD ADDRESS 200
B: START WITH SW 00=1
C: PROGRAM WILL TYPE MESSAGE
D: SET A SWITCH FOR EACH DMC DESIRED ACTIVE.
EXAMPLE: IF YOU HAVE 4 DMC'S BUT ONLY WANT TO RUN THE FIRST AND THE LAST SET SWR BITS 0 AND 3 = 1. PRESS CONTINUE
E: NUMBER (IF VALID) WILL BE IN DATA LIGHTS (EXCLUDING 11/05)
F: SET WITH ANY OTHER SWITCH SETTINGS DESIRED. PRESS CONTINUE.

4.1.3 DYNAMIC SWITCHES

ERROR SWITCHES

1. SW 12 DELETE PRINT OUT/BELL ON ERROR.
2. SW 13 DELETE ERROR PRINTOUT.
3. SW 15 HALT ON THE ERROR.
4. SW 08 GOTO BEGINNING OF THE TEST(ON ERROR).
5. SW 10 GOTO NEXT TEST(ON ERROR).

SCOPE SWITCHES

1. SW06 HALT IN ROMCLK ROUTINE BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION. THIS ALLOWS THE OPERATOR TO SCOPE A MICRO-PROCESSOR INSTRUCTION IN THE STATIC STATE BEFORE IT IS CLOCKED. HIT CONTINUE TO RESUME RUNNING.
2. SW09 (IF ENABLED BY 'SCOPI') ON AN ERROR; IF AN '*' IS PRINTED IN FRONT OF THE TEST NO. (EX. *TEST NO. 10) SW09 IS INCORPORATED IN THAT TEST AND THEREFORE SW09 IS USUALLY THE BEST SWITCH FOR THE SCOPE LOOP (SW14=0, SW10=0, SW09=1, SW08=0). IF SW09 IS NOT ENABELED; AND THERE IS A HARD ERROR (CONSTANT); SW08 IS BEST. (SW14=1,0, SW10=0, SW09=0, SW08=1). FOR 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 DMC11 DIAGNOSTICS. (SEE SECTION 4.0)

NOTE: IF ADDRESS 000042 IS NON-ZERO THE PROGRAM ASSUMES IT IS UNDER ACT11 OR XXDP CONTROL AND WILL ACT ACCORDINGLY AFTER ALL AVAILABLE DMC11'S ARE TESTED THE PROGRAM WILL RETURN TO 'XXDP' OR 'ACT-11'.

5. OPERATING PROCEDURE

WHEN PROGRAM IS INITIALLY STARTED MESSAGES AS DESCRIBED IN SECTION 4.0 WILL BE PRINTED, AND PROGRAM WILL BEGIN RUNNING THE DIAGNOSTIC

5.2 PROGRAM AND/OR OPERATOR ACTION

THE TYPICAL APPROACH SHOULD BE

1. HALT ON ERROR (VIA SW 15=1) WHEN EVER AN ERROR OCCURS.
2. CLEAR SW 15.
3. SET SW 14: (LOOP ON THIS TEST)
4. SET SW 13: (INHIBIT ERROR PRINT OUT)

THE TEST NUMBER AND PC WILL BE TYPED OUT AND 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 THE ERROR MESSAGE TO GIVE THE OPERATOR AN INDICATION OF THE ERROR.

6.2 ERROR RECOVERY

IF FOR SOME REASON THE DMC11 SHOULD 'HANG THE BUS' (GAIN CONTROL OF BUS SO THAT CONSOLE MANUAL FUNCTIONS ARE INHIBITED) AN INIT OR POWER DOWN/UP IS NECESSARY FOR OPERATOR TO REGAIN CONTROL OF CPU. IF THIS SHOULD HAPPEN; LOOK IN LOCATION 'TSTNO' (ADDRESS 1226) FOR THE NUMBER OF THE TEST THAT WAS RUNNING AT THE TIME OF THE CATASTROPHIC ERROR. IN THIS WAY THE OPERATOR WILL HAVE AN IDEA AS TO WHAT THE DMC11 WAS DOING AT THE TIME OF THE ERROR.

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

SEE SECTION 4. (PLEASE)
STATUS TABLE SHOULD BE VERIFIED 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 DMC11 DIAGNOSTIC IS LOADED INTO CORE AND RUN THE STATUS TABLE MUST BE SET UP. THIS IS DONE BY MANUAL INPUT (SW00=1) OR BY AUTOSIZING (SW00=0 AND SW07=0). THEREAFTER HOWEVER THE STATUS TABLE NEED NOT BE SETUP BY SUBSEQUENT RESTARTS OR EVEN LOADING THE NEXT DMC DIAGNOSTIC BECAUSE THE STATUS TABLE IS OVERLAYED. THE CURRENT PARAMETERS IN THE STATUS TABLE ARE USED WHEN SW07=1 ON START UP.

7.3 HARDWARE CONFIGURATION RESTRICTIONS

DMC11(M8200)- JUMPER W1 MUST BE IN, AND SWITCH 7 OF E76 MUST BE IN THE OFF POSITION.

KMC(M8204)- JUMPER W1 MUST BE IN.

8. MISCELLANEOUS

8.1 EXECUTION TIME

ALL DMC11 DEVICE DIAGNOSTICS WILL GIVE AN 'END PASS' MESSAGE (PROVIDING NO ERRORS AND SW12=0) WITHIN 4 MINS. THIS IS ASSUMING SW11=1 (DELETE ITERATIONS) IS SET TO GIVE THE FASTEST POSSIBLE EXECUTION. THE ACTUAL EXECUTION TIME DEPENDS GREATLY ON THE PDP11 CPU CONFIGURATION AND THE AMOUNT OF MEMORY IN THE SYSTEM.

8.2 PASS COMPLETE

NOTE: EVERY TIME THE PROGRAM IS STARTED; THE TESTS WILL RUN AS IF SW11 (DELETE ITERATIONS) WAS UP (=1). THIS IS TO 'VERIFY NO HARD ERRORS' AS SOON AS POSSIBLE. THEREFORE THE FIRST PASS -EACH TIME PROGRAM IS STARTED- WILL BE A 'QUICK PASS' UNTIL ALL DMC11'S IN SYSTEM ARE TESTED. WHEN THE DIAGNOSTIC HAS COMPLETED A PASS THE FOLLOWING IS AN EXAMPLE OF THE PRINT OUT TO BE EXPECTED.

END PASS CZDMC CSR: 175000 VEC: 0300 PASSES: 000001
ERRORS: 000000

NOTE: THE PASS COUNT AND ERROR COUNTS ARE CUMMULITIVE FOR EACH DMC11 THAT IS RUNNING, AND ARE SET TO ZERO ONLY WHEN THE DIAGNOSTIC IS STARTED. THEREFORE AFTER AN OVERNIGHT RUN FOR EXAMPLE, THE TOTAL PASSES AND ERRORS FOR EACH DMC11 SINCE THE DIAGNOSTIC WAS STARTED ARE REFLECTED IN PASSES: AND ERRORS:.

8.4 KEY LOCATIONS

RETURN (1214) CONTAINS THE ADDRESS WHERE PROGRAM WILL RETURN WHEN ITERATION COUNT IS REACHED OR IF LOOP ON TEST IS ASSERTED.

NEXT (1216) CONTAINS THE ADDRESS OF THE NEXT TEST TO BE PERFORMED.

TSTNO (1226) CONTAINS THE NUMBER OF THE TEST NOW BEING PERFORMED.

RUN (1316) THE BIT IN 'RUN' ALWAYS POINTS TO THE DMC11 CURRENTLY BEING TESTED. EXAMPLE: (RUN) 1302/0000000001000000 MEANS THAT DMC11 NO.06 IS THE DMC11 NOW RUNNING.

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

THESE LOCATIONS CONTAIN THE INFORMATION NEEDED TO TEST UP TO 16 (DECIMAL) DMC11S SEQUENTIALY. THEY CONTAIN THE CSR, VECTOR AND STATUS CONCERNING THE CONFIGURATION OF EACH DMC11.

DMACTV (1306) EACH BIT SET IN THIS LOCATION INDICATES THAT THE ASSOCIATED DMC11 WILL BE TESTED IN TURN. EXAMPLE: (DMACTV) 1276/0000000000011111 MEANS THAT DMC11 NO. 00,01,02,03,04 WILL BE TESTED. EXAMPLE: (DMACTV) 1276/0000000000010001 MEANS THAT DMC11 NO. 00,04 WILL BE TESTED.

DMCSR (1404) CONTAINS THE CSR OF THE CURRENT DMC11 UNDER TEST.

8.4A 'STATUS TABLE' (1500-1640)

THE TABLE IS FILLED BY AUTO SIZING OR BY THE MANUAL PARAMETER INPUT (QUESTIONS) AS DESCRIBED PREVIOUSLY. ALSO IF DESIRED BY USER; THE LOCATIONS MAY BE ALTERED BY HAND (TOGGLED IN) TO SUIT THE SPECIFIC CONFIGURATION.

THE EXAMPLE STATUS MAP SHOWN BELOW CONTAINS INFORMATION FOR TWO DMC11'S. THE TABLE CAN CONTAIN UP TO 16 DMC11'S. FOLLOWING THE MAP IS A DESCRIPTION OF THE BITS FOR EACH MAP ENTRY

MAP OF DMC11 STATUS

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

58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85

EACH MAP ENTRY CONTAINS 4 WORDS WHICH CONTAIN THE STATUS INFORMATION FOR 1 DMC11. THE PC SHOWS WHERE IN CORE MEMORY THE FIRST OF THE 4 WORDS IS. IN THE EXAMPLE ABOVE THE FIRST DMC'S STATUS IS IN LOCATIONS, 1500, 1502, 1504, AND 1506. THE SECOND DMC STATUS IS LOCATED AT 1510, 1512, 1514, AND 1516. THE INFORMATION CONTAINED IN EACH 4 WORD ENTRY IS DEFINED AS FOLLOWS:

CSR: CONTAINS DMC11 CSR ADDRESS

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

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

STAT3: BIT0=1 RUN FREE RUNNING TESTS ON KMC11
BIT1=0 DMC11-AR (LOW SPEED)
BIT1=1 DMC11-AL (HIGH SPEED)

8.5 METHOD OF AUTO SIZING

8.5.1 FINDING THE CONTROL STATUS REGISTER.

THE AUTO-SIZING ROUTINE FINDS A DMC11 AS FOLLOWS: IT STARTS AT ADDRESS 160000 AND TESTS ALL ADDRESS IN INCREMENTS OF 10 UP TO AND INCLUDING ADDRESS 167760. IF THE ADDRESS DOES NOT TIME OUT, THE FOLLOWING IS DONE, THE FIRST CROM ADDRESS IS WRITTEN TO A 125252 THEN IT IS READ BACK. IF IT CONTAINS A -1 OR 125252 OR 626 OR 16520 A DMC11 OR KMC11 HAS BEEN FOUND, IF NOT, THE ADDRESS IS UPDATED BY 10 AND THE SEARCH CONTINUES. A -1 INDICATES A DMC11 WITH NO CROM, A 125252 INDICATES A KMC11 WITH CROM, A 626 INDICATES A DMC11-AL AND A 16520 INDICATES A DMC11-AR. FURTHER TESTS ARE PERFORMED AT THIS POINT TO DETERMINE WHICH LINE UNIT, IF ANY, IS INSTALLED, IF A LOOP-BACK CONNECTOR IS INSTALLED AND VARIOUS SWITCH SETTINGS ON THE LINE UNIT. THIS IS WHY THE STATUS TABLE MUST BE VERIFIED BY THE USER AND IF ANY OF THE INFORMATION DOES NOT AGREE WITH THE HARDWARE THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS MUST BE ANSWERED. ALL DMC11'S IN THE SYSTEM WILL BE FOUND BY THE AUTO-SIZER. IF IT DOES NOT FIND A DMC11 THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS ANSWERED.

8.5.2 FINDING THE VECTOR AND BR LEVEL

THE VECTOR AREA (ADDRESS 300-776) IS FILLED WITH THE INSTRUCTION IOT AND '.+2' (NEXT ADDRESS). THE PROCESSOR STATUS IS STARTED AT 7 AND THE DMC IS PROGRAMMED TO INTERRUPT. THE PS IS LOWERED BY 1 UNTIL THE DMC INTERRUPTS, A DELAY IS MADE AND IF NO INTERRUPT OCCURES AT PS LEVEL 3 (BECAUSE OF A BAD DMC11) THE PROGRAM ASSUMES VECTOR ADDRESS 300 AT BR LEVEL 5 AND THE PROBLEM SHOULD BE FIXED IN THE DIAGNOSTIC. ONCE THE PROBLEM IS FIXED, THE PROGRAM SHOULD BE RE-SETUP AGAIN TO GET CORRECT VECTOR. IF AN INTERRUPT OCCURED, THE ADDRESS TO WHICH THE DMC11 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.

8.6 SOFTWARE SWITCH REGISTER

IF THE DIAGNOSTIC IS RUN ON AN 11/04 OR OTHER CPU WITHOUT A SWITCH REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED TO ALLOW USER THE SAME SWITCH OPTIONS AS DESCRIBED PREVIOUSLY. IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES AND IT CONTAINS ALL ONES (177777) THIS SOFTWARE SWITCH REGISTER IS USED.

CONTROL:

TO OBTAIN CONTROL AT ANY ALLOWABLE TIME DURING EXECUTION OF THE DIAGNOSTIC THE OPERATOR TYPES A CTRL G ON THE CONSOLE TERMINAL KEYBOARD. AS SOON AS THE CTRL G IS RECOGNIZED, BY THE DIAGNOSTIC, THE FOLLOWING MESSAGE WILL BE DISPLAYED:

58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95

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.

1683

```
;*AC-8544C-MC CZDMC-DO BASIC DMC11 CONTROLLER TEST
;*COPYRIGHT 1976,1978, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
;*
```

```
;STARTING PROCEDURE
;LOAD PROGRAM
;LOAD ADDRESS 000200
;SWR=0 AUTOSIZE DMC11
;SW07=1 USE CURRENT DMC11 PARAMETERS
;SW00=1 INPUT NEW DMC11 PARAMETERS
;PRESS START
;PROGRAM WILL TYPE "AC-8544C-MC CZDMC-DO BASIC DMC11 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	SW15=100000	;=1,HALT ON ERROR
040000	SW14=40000	;=1,LOOP ON CURRENT TEST
020000	SW13=20000	;=1,INHIBIT ERROR TIMEOUT
010000	SW12=10000	;=1,DELETE TIMEOUT/BELL ON ERROR.
004000	SW11=4000	;=1,INHIBIT ITERATIONS
002000	SW10=2000	;=1,ESCAPE TO NEXT TEST ON ERROR
001000	SW09=1000	;=1,LOOP WITH CURRENT DATA
000400	SW08=400	;=1,LOOP ON ERROR
000200	SW07=200	;=1,USE CURRENT DMC11 PARAMETERS, =0,AUTOSIZE DMC11
000100	SW06=100	;=1, HALT BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION
000040	SW05=40	
000020	SW04=20	
000010	SW03=10	;RESELECT DMC11'S TO BE TESTED (ACTIVE)
000004	SW02=4	;LOCK ON TEST SELECT
000002	SW01=2	;RESTART PROGRAM AT SELECTED TEST
000001	SW00=1	;INPUT DMC11 PARAMETERS

0

;REGISTER DEFINITIONS

;-

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

;LOCATION EQUIVALENCIES

;-

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

;INSTRUCTION DEFINITIONS

;-

005746	PUSH1SP=5746	;DECREMENT PROCESSOR STACK 1 WORD
005726	POP1SP=5726	;INCREMENT PROCESSOR STACK 1 WORD
010046	PUSHR0=10046	;SAVE R0 ON STACK
012600	POPPO=12600	;RESTORE R0 FROM STACK
024646	PUSH2SP=24646	;DECREMENT STACK TWICE
022626	POP2SP=22626	;INCREMENT STACK TWICE

;BIT DEFINITIONS

;-

100000	B1T15=100000
040000	B1T14=40000
020000	B1T13=20000
010000	B1T12=10000
004000	B1T11=4000
002000	B1T10=2000
001000	B1T9=1000
000400	B1T8=400
000200	B1T7=200
000100	B1T6=100
000040	B1T5=40
000020	B1T4=20
000010	B1T3=10
000004	B1T2=4
000002	B1T1=2
000001	B1T0=1


```

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

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

000024      .=24
000024 005336      .PFAIL      ;POWER FAIL HANDLER
000026 000340      340          ;SERVICE AT LEVEL 7
000030 004750      .HLT        ;ERROR HANDLER
000032 000340      340          ;SERVICE AT LEVEL 7
000034 004716      .TRPSRV     ;GENERAL HANDLER DISPATCH SERVICE
000036 000340      340          ;SERVICE AT LEVEL 7
000040 000040      .=40
000040 000000      0           ;SAVE FOR ACT-11 OR XXDP
000042 000000      0           ;RETURN ADDRESS IF UNDER ACT-11 OR XXDP
000044 000000      0           ;SAVE FOR ACT-11 OR XXDP
000046 003522      $ENDAD      ;FOR USE WITH ACT-11 OR XXDP
000052 000052      .=52
000052 000000      0           ;ACT-11 PROGRAM CHARACTERISTICS

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

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

001000 001000      .=1000
001000 377 012 101 MTITLE: .ASCII <377><12>/AC-8544C-MC/<377>
001016 103 132 104 .ASCIZ /CZDMC-DO BASIC DMC11 CONTROLLER TEST/<377>

001200      .=1200
;INDIRECT POINTERS TO SWITCH REGISTER AND LIGHT DISPLAY
;-----

001200 177570      DISPLAY:177570
001202 177570      SWR: 177570

```


;INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS

001204 177560
001206 177562
001210 177564
001212 177566

TKCSR: 177560
TKDBR: 177562
TPCSR: 177564
TPDBR: 177566

;TELETYPE KEYBOARD CONTROL REGISTER
;TELETYPE KEYBOARD DATA BUFFER
;TELEPRINTER CONTROL REGISTER
;TELEPRINTER DATA BUFFER

;PROGRAM CONTROL PARAMETERS

001214 000000
001216 000000
001220 000000
001222 000003
001224 000000
001226 000000
001230 000000
001232 000000
001234 000000

RETURN: 0
NEXT: 0
LOCK: 0
ICOUNT: 3
LPCNT: 0
TSTNO: 0
PASCNT: 0
ERRCNT: 0
LSTERR: 0

;SCOPE ADDRESS FOR LOOP ON TEST
;ADDRESS OF NEXT TEST TO BE EXECUTED
;ADDRESS FOR LOCK ON CURRENT DATA
;NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE EXECUTED
;NUMBER OF ITERATIONS COMPLETED
;NUMBER OF TEST IN PROGRESS
;NUMBER OF PASSES COMPLETED
;TOTAL NUMBER OF ERRORS
;PC OF LAST ERROR CALL

;PROGRAM VARIABLES

001236 000000
001240 000000
001242 000000
001244 000000
001246 000000
001250 000000
001252 000000
001254 000000
001256 000000
001260 000000
001262 000000
001264 000000
001266 000000
001270 000000
001272 000000
001274 000000
001276 000000
001300 000000
001302 000001
001304 000000
001306
001310
001312
001314
001316 000000

STRTSW: 0
STAT: 0
CLKX: 0
MASKX: 0
TEMP1: 0
TEMP2: 0
TEMP3: 0
TEMP4: 0
TEMP5: 0
SAVR0: 0
SAVR1: 0
SAVR2: 0
SAVR3: 0
SAVR4: 0
SAVR5: 0
SAVSP: 0
SAVPC: 0
ZERO: 0
ONE: 1
MEMLIM: 0
DMACTV: .BLKW 1
DMNUM: .BLKW 1
SAVACT: .BLKW 1
SAVNUM: .BLKW 1
RUN: 0
.EVEN

;SWITCHES AT START OF PROGRAM
;DM STATUS WORD STORAGE

;TEMPORARY STORAGE
;TEMPORARY STORAGE
;TEMPORARY STORAGE
;TEMPORARY STORAGE
;TEMPORARY STORAGE

;R0 STORAGE
;R1 STORAGE
;R2 STORAGE
;R3 STORAGE
;R4 STORAGE
;R5 STORAGE
;STACK POINTER STORAGE
;PROGRAM COUNTER STORAGE

;HIGHEST LOCATION FOR NPR'S
;DMC11'S SELECTED ACTIVE.
;OCTAL NUMBER OF DMC11'S.
;ORIGINAL ACTV DEVICES
;WORKABLE NUMBER
;POINTER TO RUNNING DEVICE.

001320 001472
001322 001676

CREAM: DM.MAP-6
MILK: CNT.MAP-4

;TABLE POINTER.
;TABLE POINTER


```

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

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

;*****
;-----
001330      104400      .TRPTAB:
001330      003576      SCOPE=TRAP+0      ;CALL TO SCOPE LOOP AND ITERATION HANDLER
                                .SCOPE
001332      003736      SCOP1=TRAP+1      ;CALL TO LOOP ON CURRENT DATA HANDLER
                                .SCOP1
                                TYPE=TRAP+2      ;CALL TO TELETYPE OUTPUT ROUTINE
001334      003766      .TYPE
                                INSTR=TRAP+3      ;CALL TO ASCII STRING INPUT ROUTINE
001336      004050      .INSTR
                                INSTER=TRAP+4      ;CALL TO INPUT ERROR HANDLER
001340      004154      .INSTER
                                PARAM=TRAP+5      ;CALL TO NUMERICAL DATA INPUT ROUTINE
001342      004174      .PARAM
                                SAV05=TRAP+6      ;CALL TO REGISTER SAVE ROUTINE
001344      004374      .SAV05
                                RES05=TRAP+7      ;CALL TO REGISTER RESTORE ROUTINE
001346      004434      .RES05
                                CONVRT=TRAP+10      ;CALL TO DATA OUTPUT ROUTINE
001350      004466      .CONVRT
                                CNVRT=TRAP+11      ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
001352      004472      .CNVRT
                                MSTCLR=TRAP+12      ;CALL TO ISUE A MASTER CLEAR
001354      005466      .MSTCLR
                                DELAY=TRAP+13      ;CALL TO DELAY
001356      005436      .DELAY
                                ROMCLK=TRAP+14      ;CALL TO CLOCK ROM ONCE
001360      005504      .ROMCLK
                                DATACLK=TRAP+15      ;CALL TO CLK DATA
001362      005552      .DATACLK
                                TIMER=TRAP+16      ;CALL TO DELAY A CLOCK TICK
001364      005616      .TIMER
;-----
;*****

```


;DMC11 CONTROL INDICATORS FOR CURRENT DMC11 UNDER TEST

001366 000000
001370 000000
001372 000000

STAT1: 0
STAT2: 0
STAT3: 0

;DMC11 VECTOR AND REGISTER INDIRECT POINTERS

001374 000000
001376 000000
001400 000000
001402 000000
001404 000000
001406 000000
001410 000000
001412 000000
001414 000000

DMRVEC: 0 ;POINTER TO DMC11 RECEIVER INTERRUPT VECTOR
DMRLVL: 0 ;POINTER TO DMC11 RECEIVER INTERRUPT SERVICE PS
DMTVEC: 0 ;POINTER TO DMC11 TRANSMITTER INTERRUPT VECTOR
DMTLVL: 0 ;POINTER TO DMC11 TRANSMITTER INTERRUPT SERVICE PS
DMCSR: 0 ;POINTER TO DMC11 CONTROL STATUS REGISTER
DMCSRH: 0 ;POINTER TO DMC11 CONTROL STATUS REGISTER HIGH BYTE.
DMCTL: 0 ;POINTER TO DMC11 CONTROL OUT REGISTER
DMP04: 0 ;POINTER TO DMC11 PORT REGISTER(SEL 4)
DMP06: 0 ;POINTER TO DMC11 PORT REGISTER(SEL 6)

;TEMP STORAGE

001416 000000
001460

TEMP: 0
.=.+40

;DMC11 STATUS TABLE AND ADDRESS ASSIGNMENTS

001500 001500

001500
001500
001502
001504
001506

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

001510
001512
001514
001516

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

001520
001522
001524
001526

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

001530
001532
001534
001536

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

001540
001542
001544
001546

DMCR04: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 04
DMS104: .BLKW 1 ;VECTOR FOR DMC11 NUMBER 04
DMS204: .BLKW 1 ;DDCMP LINE# FOR DMC11 NUMBER 04
DMS304: .BLKW 1 ;3RD STATUS WORD

001550

DMCR05: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 05

001552	DMS105: .BLKW	1	;VECTOR FOR DMC11 NUMBER 05
001554	DMS205: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 05
001556	DMS305: .BLKW	1	;3RD STATUS WORD
001560	DMCR06: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 06
001562	DMS106: .BLKW	1	;VECTOR FOR DMC11 NUMBER 06
001564	DMS206: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 06
001566	DMS306: .BLKW	1	;3RD STATUS WORD
001570	DMCR07: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 07
001572	DMS107: .BLKW	1	;VECTOR FOR DMC11 NUMBER 07
001574	DMS207: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 07
001576	DMS307: .BLKW	1	;3RD STATUS WORD
001600	DMCR10: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 10
001602	DMS110: .BLKW	1	;VECTOR FOR DMC11 NUMBER 10
001604	DMS210: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 10
001606	DMS310: .BLKW	1	;3RD STATUS WORD
001610	DMCR11: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 11
001612	DMS111: .BLKW	1	;VECTOR FOR DMC11 NUMBER 11
001614	DMS211: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 11
001616	DMS311: .BLKW	1	;3RD STATUS WORD
001620	DMCR12: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 12
001622	DMS112: .BLKW	1	;VECTOR FOR DMC11 NUMBER 12
001624	DMS212: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 12
001626	DMS312: .BLKW	1	;3RD STATUS WORD
001630	DMCR13: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 13
001632	DMS113: .BLKW	1	;VECTOR FOR DMC11 NUMBER 13
001634	DMS213: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 13
001636	DMS313: .BLKW	1	;3RD STATUS WORD
001640	DMCR14: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 14
001642	DMS114: .BLKW	1	;VECTOR FOR DMC11 NUMBER 14
001644	DMS214: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 14
001646	DMS314: .BLKW	1	;3RD STATUS WORD
001650	DMCR15: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 15
001652	DMS115: .BLKW	1	;VECTOR FOR DMC11 NUMBER 15
001654	DMS215: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 15
001656	DMS315: .BLKW	1	;3RD STATUS WORD
001660	DMCR16: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 16
001662	DMS116: .BLKW	1	;VECTOR FOR DMC11 NUMBER 16
001664	DMS216: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 16
001666	DMS316: .BLKW	1	;3RD STATUS WORD
001670	DMCR17: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 17
001672	DMS117: .BLKW	1	;VECTOR FOR DMC11 NUMBER 17
001674	DMS217: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 17
001676	DMS317: .BLKW	1	;3RD STATUS WORD
001700	000000		
	DM.END: 000000		

;DMC11 PASS COUNT AND ERROR COUNT TABLE

001702		CNT.MAP:	
001702	000000	PACT00: 0	;PASS COUNT FOR DMC11 NUMBER 00
001704	000000	ERCT00: 0	;ERROR COUNT FOR DMC11 NUMBER 00
001706	000000	PACT01: 0	;PASS COUNT FOR DMC11 NUMBER 01
001710	000000	ERCT01: 0	;ERROR COUNT FOR DMC11 NUMBER 01
001712	000000	PACT02: 0	;PASS COUNT FOR DMC11 NUMBER 02
001714	000000	ERCT02: 0	;ERROR COUNT FOR DMC11 NUMBER 02
001716	000000	PACT03: 0	;PASS COUNT FOR DMC11 NUMBER 03
001720	000000	ERCT03: 0	;ERROR COUNT FOR DMC11 NUMBER 03
001722	000000	PACT04: 0	;PASS COUNT FOR DMC11 NUMBER 04
001724	000000	ERCT04: 0	;ERROR COUNT FOR DMC11 NUMBER 04
001726	000000	PACT05: 0	;PASS COUNT FOR DMC11 NUMBER 05
001730	000000	ERCT05: 0	;ERROR COUNT FOR DMC11 NUMBER 05
001732	000000	PACT06: 0	;PASS COUNT FOR DMC11 NUMBER 06
001734	000000	ERCT06: 0	;ERROR COUNT FOR DMC11 NUMBER 06
001736	000000	PACT07: 0	;PASS COUNT FOR DMC11 NUMBER 07
001740	000000	ERCT07: 0	;ERROR COUNT FOR DMC11 NUMBER 07
001742	000000	PACT10: 0	;PASS COUNT FOR DMC11 NUMBER 10
001744	000000	ERCT10: 0	;ERROR COUNT FOR DMC11 NUMBER 10
001746	000000	PACT11: 0	;PASS COUNT FOR DMC11 NUMBER 11
001750	000000	ERCT11: 0	;ERROR COUNT FOR DMC11 NUMBER 11
001752	000000	PACT12: 0	;PASS COUNT FOR DMC11 NUMBER 12
001754	000000	ERCT12: 0	;ERROR COUNT FOR DMC11 NUMBER 12
001756	000000	PACT13: 0	;PASS COUNT FOR DMC11 NUMBER 13
001760	000000	ERCT13: 0	;ERROR COUNT FOR DMC11 NUMBER 13
001762	000000	PACT14: 0	;PASS COUNT FOR DMC11 NUMBER 14
001764	000000	ERCT14: 0	;ERROR COUNT FOR DMC11 NUMBER 14
001766	000000	PACT15: 0	;PASS COUNT FOR DMC11 NUMBER 15
001770	000000	ERCT15: 0	;ERROR COUNT FOR DMC11 NUMBER 15
001772	000000	PACT16: 0	;PASS COUNT FOR DMC11 NUMBER 16
001774	000000	ERCT16: 0	;ERROR COUNT FOR DMC11 NUMBER 16
001776	000000	PACT17: 0	;PASS COUNT FOR DMC11 NUMBER 17
002000	000000	ERCT17: 0	;ERROR COUNT FOR DMC11 NUMBER 17

FORMAT OF STATUS TABLE

CSR

STAT1

STAT2

STAT3

DEFINITION OF FORMAT

```

CSR:      CONTAINS DMC11 CSR ADDRESS

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

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

STAT3:    BIT0=1 DO FREE RUNNING TESTS ON KMC
           (MUST BE SET TO A ONE MANUALLY [PROGRAM DZDMI ONLY])
           KMC MUST HAVE MICRO-CODE WRITTEN FROM RUNNING
           DZDMG TEST 2 FIRST
           BIT1=1 DMC11-AL LOCAL HIGH SPEED MICRO-CODE
           BIT1=0 DMC11-AR REMOTE LOW SPEED MICRO-CODE

```



```

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

002002 012737 000340 177776 .START: MOV #340,PS ;LOCK OUT INTERRUPTS
002010 012706 001200 MOV #STACK,SP ;SET UP STACK
002014 012737 005336 000024 MOV #.PFAIL,#24 ;SET UP POWER FAIL VECTOR
002022 013737 001310 001314 MOV DMNUM,SAVNUM ;SAVE NUMBER OF DEVICES IN SYSTEM.
002030 005037 010016 CLR SWFLG ;CLEAR SOFT TYPEOUT FLAG
002034 105037 001325 CLRB ERRFLG ;CLEAR ERROR FLAG
002040 105037 001327 CLRB QV.FLG ;ZERO QUICK VERIFY FLAG
002044 012737 001470 001320 MOV #DM.MAP-10,CREAM ;GET MAP POINTER.
002052 012737 001676 001322 MOV #CNT.MAP-4,MILK ;GET PASS COUNT MAP POINTER
002060 012737 100000 001316 MOV #BIT15,RUN ;POINT POINTER TO FIRST DEVICE.
002066 012700 001702 MOV #CNT.MAP,RO ;PASS COUNT POINTER TO RO
002072 005020 23$: CLR (RO)+ ;CLEAR TABLE
002074 022700 002002 CMP #CNT.MAP+100,RO ;DONE YET?
002100 001374 BNE 23$ ;KEEP GOING
002102 005037 001234 CLR LSTERR ;CLEAR LAST ERROR POINTER
002106 012737 000001 001226 MOV #1,TSTNO ;SET UP FOR TEST 1
002114 012737 002002 001214 MOV #.START,RETURN ;SET UP FOR POWER FAIL BEFORE
;TESTING STARTS
;SAVE CURRENT VECTORS
;
002122 013746 000006 MOV #6,-(SP) ;SET UP FOR TIMEOUT
002126 013746 000004 MOV #4,-(SP) ;
002132 012737 002166 000004 MOV #6$,#4 ;SET UP FOR TIMEOUT
002140 012737 177570 001202 MOV #177570,SWR ;SET SWR TO HARD SWR ADDRESS
002146 012737 177570 001200 MOV #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
002154 022777 177777 177020 CMP #-1,#SWR ;REFERENCE HARDWARE SWITCH REGISTER
002162 001402 BEQ 6$+2 ;IF = -1 USE SOFT SWR ANYWAY
002164 000407 BR 7$ ;IF IT EXISTS AND NOT = -1 USE HARD SWR
002166 022626 6$: CMP (SP)+,(SP)+ ;ADJUST STACK
002170 012737 000176 001202 MOV #SWREG,SWR ;POINTER TO SOFT SWR
002176 012737 000174 001200 MOV #DISPREG,DISPLAY ;POINTER TO SOFT DISPLAY REG
002204 012637 000004 7$: MOV (SP)+,#4 ;RESTORE VECTORS
002210 012637 000006 MOV (SP)+,#6 ;
002214 105737 001324 TSTB INIFLG ;HAS INITIALIZATION BEEN PERFORMED
002220 001006 BNE 20$ ;BR IF YES
002222 022737 003522 000042 CMP #ENDAD,#42 ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
002230 001402 BEQ 20$ ;
002232 104402 001000 TYPE ,MTITLE ;TYPE TITLE MESSAGE
002236 004737 007606 20$: JSR PC,CKSWR ;CHECK FOR SOFT SWR
002242 017737 176734 001236 MOV #SWR,STRTSW ;STORE STARTING SWITCHES
002250 005737 000042 TST #42 ;IS IT RUNNING IN AUTO MODE?
002254 001402 BEQ .+6 ;BR IF NO
002256 005037 001236 CLR STRTSW ;IF YES, CLEAR SWITCHES
002262 032737 000001 001236 BIT #SW00,STRTSW ;IF SW00=1, QUESTIONS ARE ASKED.
002270 001012 BNE 17$ ;BR IF SW00=1
002272 105737 001236 TSTB STRTSW ;BIT7=1??
002276 100007 BPL 17$ ;BR IF SW07=0
002300 005737 001306 TST DMACTV ;ARE ANY DEVICES SELECTED?
002304 001006 BNE 16$ ;BR IF YES
002306 104402 007154 TYPE, NOACT ;NO DEVICES SELECTED.
002312 000000 HALT ;STOP THE SHOW

```


002314	000776			BR	.-2	;DISQUALIFY CONTINUE SWITCH
002316	004737	010512	17#:	JSR	PC,AUTO.SIZE	;GO DO THE AUTO SIZE
002322	105737	001324	16#:	TSTB	INIFLG	;FIRST TIME?
002326	001410			BEQ	21#	;BR IF YES
002330	105737	001236		TSTB	STRTSW	;IF USING SAME PARAMETERS DONT TYPE MAP
002334	100431			BMI	1#	
002336	032737	000006	001236	BIT	#BIT1!BIT2,STRTSW	;IS TEST NO. OR LOCK SELECTED
002344	001403			BEQ	24#	;IF NO THEN TYPE STATUS
002346	000424			BR	1#	;IF YES DO NOT TYPE STATUS
002350	005137	001324	21#:	COM	INIFLG	;SET FLAG
002354	104402	006224	24#:	TYPE	,XHEAD	;TYPE HEADER
002360	012704	001500		MOV	#DM,MAP,R4	;SET POINTER
002364	010437	001246	5#:	MOV	R4,TEMP1	;SET ADDRESS
002370	012437	001250		MOV	(R4)+,TEMP2	;SET CSR
002374	001411			BEQ	1#	;ALL DONE IF ZERO
002376	012437	001252		MOV	(R4)+,TEMP3	;SET STAT1
002402	012437	001254		MOV	(R4)+,TEMP4	;SET STAT2
002406	012437	001256		MOV	(R4)+,TEMP5	;SET STAT3
002412	104410			CONVRT		;TYPE OUT STATUS MAP
002414	007454			XSTATQ		
002416	000762			BR	5#	
002420	012700	001500	1#:	MOV	#DM,MAP,R0	;R0 POINTS TO STATUS TABLE

```

*****
;;*AUTO SIZE TEST
;;*THIS TEST VERIFYS THAT THE DMC11S AND/OR KMC11S ARE AT THE CORRECT FLOATING
;;*ADDRESSES FOR YOUR SYSTEM. IF THIS TEST FAILS, IT IS NOT A HARDWARE ERROR.
;;*CHECK THE ADDRESSES OF ALL FLOATING DEVICES (DJ,DH,DQ,DU,DUP,LK,DMC,DZ,KMC).
;;*IF THERE ARE NO OTHER FLOATING DEVICES BEFORE THE DMC11, THE FIRST
;;*DMC11 ADDRESS IS 760070, KMC11 IS 760110. NO DEVICE SHOULD EVER BE AT
;;*ADDRESS 760000. THIS TEST MAY REQUIRE 2 OR MORE ATTEMPTS TO GET THE
;;*RIGHT ADDRESSES. AFTER YOU HAVE CHANGED THE ADDRESS TO WHAT IT TOLD
;;*YOU THE FIRST TIME, IT MAY COME BACK AND TELL YOU A DIFFERENT ADDRESS
;;*THE NEXT TIME YOU RUN IT. PLEASE HAVE PATIENCE, THE FINAL ADDRESS
;;*WILL BE CORRECT (AS LONG AS ALL DEVICES IN FRONT OF THE DMC'S ARE
;;*CORRECT).
*****

```

002424	013746	000004		MOV	#4,-(SP)	;SAVE LOC 4
002430	013746	000006		MOV	#6,-(SP)	;SAVE LOC 6
002434	005037	000006		CLR	#6	;CLEAR VEC+2
002440	005037	001252		CLR	TEMP3	;CLEAR FLAG
002444	005005			CLR	R5	;R5=0=DMC, R5=-1=KMC
002446	011037	001404	AUSTRT:	MOV	(R0),DMCSR	;GET NEXT DMC CSR
002452	001564			BEQ	AUDONE	;BR IF DONE
002454	005705			TST	R5	;DMC OR KMC?
002456	001005			BNE	1#	;BR IF KMC
002460	032760	100000	000002	BIT	#BIT15,2(R0)	;CHECK FOR DMC CSR
002466	001061			BNE	SKIP	;SKIP IF NOT DMC
002470	000404			BR	2#	;ITS A DMC SO CONTINUE
002472	032760	100000	000002	1#:	BIT	#BIT15,2(R0)
002500	001454			BEQ	SKIP	;CHECK FOR KMC CSR
002502	012737	002674	000004	2#:	MOV	#NODEV,#4
002510	005705			TST	R5	;SET UP FOR TIMEOUT
002512	001003			BNE	3#	;DMC OR KMC?
002514	012703	000006		MOV	#6,R3	;BR IF KMC
002520	000402			BR	4#	;R3 IS COUNT OF DEVICES BEFORE DMC
						;GO ON

002522	012703	000010	3#:	MOV	#10,R3	;R3 IS COUNT OF DEVICES BEFORE KMC
002526	012702	003010	4#:	MOV	#DEV TAB,R2	;R2 IS DEVICE TABLE POINTER
002532	012701	160010		MOV	#160010,R1	;START WITH ADDRESS 160010
002536	005711		Float:	TST	(R1)	;CHECK ADDRESS IN R1
002540	111204			MOVB	(R2),R4	;IF NO TIMEOUT, GET NEXT ADDRESS
002542	060401			ADD	R4,R1	;IN R1
002544	005201			INC	R1	;
002546	040401			BIC	R4,R1	;
002550	005703			TST	R3	;ANY MORE DEVICES TO CHECK FOR?
002552	001371			BNE	Float	;BR IF YES
002554	012737	002700	000004	MOV	#ERR,8#4	;OK ONLY DMC'S ARE LEFT, SET UP FOR TIMEOUT
002562	010137	003022		MOV	R1,XLOC	;SAVE FIRST DMC/KMC ADDRESS
002566	005705		FY:	TST	R5	;DMC OR KMC?
002570	001005			BNE	1#	;BR IF KMC
002572	032760	100000	000002	BIT	#BIT15,2(R0)	;CHECK FOR DMC CSR
002600	001014			BNE	SKIP	;SKIP IF NOT DMC
002602	000404			BR	2#	;ITS A DMC SO CONTINUE
002604	032760	100000	000002	1#:	BIT	#BIT15,2(R0)
002612	001407			BEQ	SKIP	;CHECK FOR KMC CSR
002614	005711		2#:	TST	(R1)	;SKIP IF NOT KMC
002616	020137	001404		CMP	R1,DMCSR	;CHECK DMC ADDRESS
002622	001411			BEQ	OK	;DOES IT MATCH
002624	062701	000010		ADD	#10,R1	;BR IF YES
002630	000756			BR	FY	;GET NEXT DMC ADDRESS
002632	062700	000010	SKIP:	ADD	#10,R0	;DO IT AGAIN
002636	011037	001404		MOV	(R0),DMCSR	;SKIP TO NEXT CSR IN TABLE
002642	001470			BEQ	AUDONE	;GET NEXT CSR
002644	000750			BR	FY	;BR IF DONE
002646	062700	000010	OK:	ADD	#10,R0	;ELSE CONTINUE
002652	062737	000010	003022	ADD	#10,XLOC	;SKIP TO NEXT DMC CSR
002660	011037	001404		MOV	(R0),DMCSR	;UPDATE EXPECTED DMC/KMC ADDRESS
002664	001457			BEQ	AUDONE	;GET NEXT DMC/KMC CSR
002666	013701	003022		MOV	XLOC,R1	;BR IF DONE
002672	000735			BR	FY	;GET EXPECTED DMC/KMC ADDRESS
002674	122243		NODEV:	CMPB	(R2)+,-(R3)	;CONTINUE
002676	000002			RTI		;ON TIMEOUT, INC R2, DEC R3
002700	005737	001252	ERR:	TST	TEMP3	;RETURN
002704	001014			BNE	1#	;CHECK FLAG IF = 0 TYPE HEADER
002706	104402			TYPE		;SKIP HEADER
002710	007223			CONERR		;TYPEOUT HEADER MESSAGE
002712	012737	002700	001276	MOV	#ERR,SAVPC	;CONFIGURATION ERROR!!!!
002720	104411			CNVRT		;SAVE PC FOR TYPEOUT
002722	002770			ERRPC		;TYPE OUT ERROR PC
002724	104402			TYPE		;
002726	007277			CNERR		;TYPE REST OF HEADER
002730	012737	177777	001252	MOV	#-1,TEMP3	;
002736	010137	001262	1#:	MOV	R1,SAVR1	;SET FLAG SO IT ONLY GETS TYPED ONCE
002742	104410			CONVRT		;SAVE R1 FOR TYPEOUT
002744	002776			CONTAB		;
002746	005705			TST	R5	;TYPE CSR VALUES
002750	001003			BNE	3#	;DMC OR KMC ?
002752	104402			TYPE		;BR IF KMC
002754	007320			DMCM		;
002756	000402			BR	4#	;CONTINUE
002760	104402		3#:	TYPE		;
002762	007330			KMCM		;
002764	022626		4#:	CMP	(SP)+,(SP)+	;ADJUST STACK


```

002766 000727          BR      OK          ;BR TO GET OUT
002770 000001          ERRPC: 1
002772 006            002          .BYTE 6,2
002774 001276          SAVPC
002776 000002          CONTAB: 2
003000 006            004          .BYTE 6,4
003002 003022          XLOC
003004 006            002          .BYTE 6,2
003006 001404          DMCSR
003010 007          DEVTAB: .BYTE 7          ;DJ
003011 017          .BYTE 17          ;DH
003012 007          .BYTE 7          ;DQ
003013 007          .BYTE 7          ;DU
003014 007          .BYTE 7          ;DUP
003015 007          .BYTE 7          ;LK
003016 007          .BYTE 7          ;DMC
003017 007          .BYTE 7          ;DZ
003020 007          .BYTE 7          ;KMC

003022 000000          .EVEN
003024 005705          XLOC: 0
003026 001005          AUDONE: TST      R5
003030 012705 177777    MOV      #-1,R5          ;DMC?
003034 012700 001500    MOV      #DM.MAP,R0          ;BR IF KMC AND ALL DONE
003040 000602          BR      AUSTRT          ;SET R5 TO -1 (KMC)
003042 012637 000006    1$: MOV      (SP)+,B#6          ;RESET R0 TO START OF TABLE
003046 012637 000004    MOV      (SP)+,B#4          ;GO DO KMC'S
003052 032737 000010 001236 BIT      #SW03,STRTSW          ;RESTORE LOC 6
003060 001422          BEQ      3$          ;RESTORE LOC 4
003062 104402 006144    TYPE      ,MNEW          ;SELECT SPECIFIC DEVICES??
003066 005000          CLR      R0          ;BR IF NO.
003070 000000          HALT          ;TYPE THE MESSAGE.
003072 027737 176104 001312 CMP      @SWR,SAVACT          ;ZERO DATA LIGHTS
003100 101404          BLOS      2$          ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
003102 104402 006005    TYPE      ,MERR3          ;IS THE NUMBER VALID?
003106 000000          HALT          ;BR IF NUMBER IS OK.
003110 000776          BR      .-2          ;TELL USER OF INVALID NUMBER.
003112 017737 176064 001306 2$: MOV      @SWR,DMACTV          ;STOP EVERY THING.
003120 013700 001306    MOV      DMACTV,R0          ;RESTART THE PROGRAM AGAIN.
003124 000000          HALT          ;GET NEW DEVICE PATTERN
003126 012700 000300    3$: MOV      #300,R0          ;SHOW THE USER WHAT HE SELECTED.
003132 012701 000302    MOV      #302,R1          ;CONTINUE DYNAMIC SWITCHES.
003136 010120          4$: MOV      R1,(R0)+          ;PREPARE TO CLEAR THE FLOATING
003140 005021          CLR      (R1)+          ;VECTOR AREA. 300-776
003142 022021          CMP      (R0)+,(R1)+          ;START PUTTING "PC+2 - HALT"
003144 022700 001000    CMP      #1000,R0          ;IN VECTOR AREA.
003150 001372          BNE      4$          ;POP POINTERS
                                          ;ALL DONE??
                                          ;BR IF NO.

          ;TEST START AND RESTART
          ;-----

003152 012706 001200    .BEGIN: MOV      @STACK,SP          ;SET UP STACK
003156 013746 000006    MOV      B#6,-(SP)          ;SAVE LOC 6
003162 013746 000004    MOV      B#4,-(SP)          ;SAVE LOC 4
003166 005000          CLR      R0          ;START AT 0
003170 012737 003234 000004 MOV      #21,B#4          ;SET UP FOR TIME OUT
003176 005037 000006    CLR      B#6          ;TO AUTOSIZE MEMORY

```


Address	Hex	Dec	Label	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op41
---------	-----	-----	-------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	------


```

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

003364 000005 .EOP: RESET ;MAKE THE WORLD CLEAN AGAIN.
003366 005037 001234 CLR LSTERR ;CLEAR LAST ERROR PC
003372 105037 001325 CLR ERRFLG ;CLEAR ERROR FLAG
003376 005237 001230 INC PASCNT ;UPDATE PASS COUNT
003402 013777 001230 175570 MOV PASCNT,@DISPLAY ;DISPLAY PASS COUNT
003410 104402 005733 TYPE ,MEPASS ;TYPE END PASS
003414 104402 006072 TYPE ,MCSR ;TYPE CSR
003420 104411 003546 CNVRT ,XCSR ;SHOW IT
003424 104402 006100 TYPE ,MVECX ;TYPE VECTOR
003430 104411 003554 CNVRT ,XVEC ;SHOW IT
003434 104402 006106 TYPE ,MPASSX ;TYPE PASSES
003440 104411 003562 CNVRT ,XPASS ;SHOW IT
003444 104402 006117 TYPE ,MERRX ;TYPE ERRORS
003450 104411 003570 CNVRT ,XERR ;SHOW IT
003454 013700 001322 MOV MILK,RO ;GET POINTER TO PASS COUNT
003460 013720 001230 MOV PASCNT,(RO)+ ;STORE PASS COUNT FOR THIS DMC11
003464 013720 001232 MOV ERRCNT,(RO)+ ;STORE ERROR COUNT FOR THIS DMC11
003470 005337 001314 DEC SAVNUM ;ARE ALL DEVICES TESTED?
003474 001017 BNE RESTR ;BR IF NO.
003476 112737 000377 001327 MOV #377,QV.FLG ;SET THE QUICK VERIFY FLAG.
003504 013737 001310 001314 MOV DMNUM,SAVNUM ;RESTORE THE COUNT
003512 013701 000042 MOV @42,R1 ;CHECK FOR ACT-11 OR DDP
003516 001406 BEQ RESTR ;IF NOT, CONTINUE TESTING
003520 000005 RESET ;STOP THE SHOW--CLEAR THE WORLD
003522 ;ENDAD:
003522 004711 JSR PC,(R1)
003524 000240 NOP
003526 000240 NOP
003530 000240 NOP
003532 000240 NOP
003534 012737 010060 001214 RESTR: MOV #CYCLE,RETURN
003542 000137 010060 JMP CYCLE
003546 000001 XCSR: 1
003550 006 002 .BYTE 6.2
003552 001404 DMCSR
003554 000001 XVEC: 1
003556 004 002 .BYTE 4.2
003560 001374 DMRVEC
003562 000001 XPASS: 1
003564 006 002 .BYTE 6.2
003566 001230 PASCNT
003570 000001 XERR: 1
003572 006 002 .BYTE 6.2
003574 001232 ERRCNT

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

003576 004737 007606 .SCOPE: JSR PC,CKSWR ;CHECK FOR SOFT SWR
003602 010016 MOV RO,(SP) ;SAVE RO ON THE STACK
003604 032777 040000 175370 BIT #BIT14,@SWR ;"LOOP ON THIS TEST"?

```



```

003612 001407          TTST: BEQ 1$          ;BR IF NO. (IF LOCK SW01=1; THIS LOC =240)
003614 000437          BR 3$          ;GOTO 3$ (IF LOCK SW01=1; THIS LOC =240)
003616 005737 003734    TST DONE        ;WAS TKCSR DONE SET?
003622 001434          BEQ 3$          ;BR IF NO (LOCKED ON TEST)
003624 005037 003734    CLR DONE        ;YES, CLEAR FLAG
003630 000415          BR 2$          ;GO TO NEXT TEST
003632 032777 004000 175342 1$: BIT #SW11,@SWR ;DELETE ITERATION? (QUICK PASS)
003640 001011          BNE 2$          ;BR IF YES
003642 105737 001327    TSTB QV.FLG     ;HAVE PASSES BEECOMPLETED?
003646 001406          BEQ 2$          ;BR IF QUICK PASS.
003650 005237 001224    INC LPCNT        ;UPDATE ITERATION COUNTER
003654 023737 001224 001222    CMP LPCNT,ICOUNT ;ARE ALL ITERATIONS DONE??
003662 101414          BLOS 3$         ;BR IF NOT YET
003664 105037 001325    2$: CLRB ERRFLG  ;PREPARE FOR NEW TEST
003670 005037 001224    CLR LPCNT       ;START ICOUNTER AT 0
003674 005037 001220    CLR LOCK
003700 012737 000020 001222    MOV #20,ICOUNT ;RESET ITERATIONS
003706 013737 001216 001214    MOV NEXT,RETURN ;GET NEXT TEST
003714 011600          3$: MOV (SP),R0    ;POP R0 OFF OF THE STACK
003716 022626          POP2SP          ;FAKE AN "RTI"
003720 013701 001404    MOV DMCSR,R1    ;R1 CONTAINS BASE DMC ADDRESS
003724 000177 175264    JMP @RETURN    ;GO DO THE TEST
003730 001407          BRW: 1407
003732 000437          BRX: 437
003734 000000          DONE: 0

```

;CHECK FOR FREEZE ON CURRENT DATA

```

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

```

;TELETYPE OUTPUT ROUTINE

```

003766 010546          .TYPE: MOV R5,-(SP) ;SAVE R5 ON THE STACK.
003770 017605 000002    MOV @2(SP),R5    ;GET ADDRESS OF MESSAGE.
003774 062766 000002 000002    ADD #2,2(SP) ;POP OVER ADDRESS.
004002 005737 010016    4$: TST SWFLG    ;SOFT SWR MESSAGE?
004006 001004          BNE 1$          ;IF YES TYPE IT OUT REGARDLESS OF SW12
004010 032777 010000 175164    BIT #SW12,@SWR ;INHIBIT ALL PRINT OUT??
004016 001012          BNE 3$          ;BR IF NO PRINT OUT WANTED (SW12=1)
004020 105715          1$: TSTB (R5)    ;IS NUMBER MINUS? (MSB=1(BIT7))
004022 100002          BPL 2$          ;BR IF NUMBER IS PLUS
004024 104402 005672    TYPE ,MCRLF     ;TYPE A CR/LF!
004030 105777 175154    2$: TSTB @TPCSR ;TTY READY?
004034 100375          BPL 2$          ;BR IF NO.
004036 112577 175150    MOVB (R5)+,@TPDBR ;PRINT CURRENT CHAR.
004042 001357          BNE 4$          ;IF NOT ZERO KEEP PRINTING!
004044 012605          3$: MOV (SP)+,R5 ;END OF OUTPUT. RESTORE R5
004046 000002          RTI          ;GO HOME

```



```

004050 010346      .INSTR: MOV    R3,-(SP)      ;SAVE R3 ON STACK
004052 010446      MOV    R4,-(SP)      ;SAVE R4 ON STACK
004054 017637 000004 004072      MOV    @4(SP),.MSG
004062 062766 000002 000004      ADD    #2,4(SP)
004070 104402      .INST1: TYPE
004072 000000      .MSG: 0
004074 012704 007502      MOV    #INBUF,R4
004100 012703 000007      MOV    #7,R3
004104 105777 175074      1$:  TSTB   @TKCSR
004110 100375      BPL     1$
004112 117714 175070      MOVB   @TKDBR,(R4)
004116 142714 000200      BICB   #200,(R4)
004122 122427 000015      CMPB   (R4)+,#15
004126 001417      BEQ     INSTR2
004130 105777 175054      2$:  TSTB   @TPCSR
004134 100375      BPL     2$
004136 017777 175044 175046      MOV    @TKDBR,@TPDBR
004144 005303      DEC     R3
004146 001356      BNE     1$
004150 012604      MOV    (SP)+,R4
004152 012603      MOV    (SP)+,R3
004154 104402 005666      .INSTE: TYPE ,MQM
004160 010346      MOV    R3,-(SP)
004162 010446      MOV    R4,-(SP)
004164 000741      BR     .INST1
004166 012604      INSTR2: MOV   (SP)+,R4      ;RESTORE R4
004170 012603      MOV    (SP)+,R3      ;RESTORE R3
004172 000002      RTI

```

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

```

004174 010546      .PARAM: MOV    R5,-(SP)
004176 010446      MOV    R4,-(SP)
004200 016605 000004      MOV    4(SP),R5
004204 012537 004364      MOV    (R5)+,LOLIM
004210 012537 004366      MOV    (R5)+,HILIM
004214 012537 004370      MOV    (R5)+,DEVADR
004220 112537 004372      MOVB   (R5)+,LOBITS
004224 112537 004373      MOVB   (R5)+,ADRCNT
004230 010566 000004      MOV    R5,4(SP)
004234 005005      PARAM1: CLR    R5
004236 012704 007502      MOV    #INBUF,R4
004242 122714 000015      CMPB   #15,(R4)
004246 001420      BEQ     PARERR
004250 121427 000060      1$:  CMPB   (R4),#60
004254 002415      BLT     PARERR
004256 121427 000067      CMPB   (R4),#67
004262 003012      BGT     PARERR
004264 142714 000060      BICB   #60,(R4)
004270 152405      BISB   (R4)+,R5
004272 122714 000015      CMPB   #15,(R4)
004276 001406      BEQ     LIMITS
004300 006305      ASL    R5
004302 006305      ASL    R5
004304 006305      ASL    R5

```


004306 000760
004310 104404
004312 000750

BR 1\$
PARERR: INSTER
BR PARAM1

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

004314 020537 004366
004320 101373
004322 020537 004364
004326 103770
004330 133705 004372
004334 001365

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

;STORE NUMBER AT SPECIFIED ADDRESS

004336 013704 004370
004342 010524
004344 062705 000002
004350 105337 004373
004354 001372
004356 012604
004360 012605
004362 000002
004364 000000
004366 000000
004370 000000
004372 000000
004373

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

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

004374 016637 000004 001276

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

;SAVE R0-R5

004402 010537 001272
004406 010437 001270
004412 010337 001266
004416 010237 001264
004422 010137 001262
004426 010037 001260
004432 000002

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

;RESTORE R0-R5

004434 013700 001260
004440 013701 001262
004444 013702 001264
004450 013703 001266
004454 013704 001270
004460 013705 001272
004464 000002

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

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

004466	104402	005672		.CONVR: TYPE	,MCRLF
004472	010046			.CNVRT: MOV	R0,-(SP)
004474	010146			MOV	R1,-(SP)
004476	010346			MOV	R3,-(SP)
004500	010446			MOV	R4,-(SP)
004502	010546			MOV	R5,-(SP)
004504	017601	000012		MOV	#12(SP),R1
004510	062766	000002	000012	ADD	#2,12(SP)
004516	012137	004710		MOV	(R1)+,WRDCNT
004522	112137	004712	1\$:	MOVB	(R1)+,CHRCNT
004526	112137	004713		MOVB	(R1)+,SPACNT
004532	013137	004714		MOV	#(R1)+,BINWRD
004536	122737	000003	004712	CMPB	#3,CHRCNT
004544	001003			BNE	2\$
004546	042737	177400	004714	BIC	#177400,BINWRD
004554	013704	004714	2\$:	MOV	BINWRD,R4
004560	113705	004712		MOVB	CHRCNT,R5
004564	012700	001416		MOV	#TEMP,R0
004570	010403		3\$:	MOV	R4,R3
004572	042703	177770		BIC	#177770,R3
004576	062703	000060		ADD	#060,R3
004602	110320			MOVB	R3,(R0)+
004604	000241			CLC	
004606	006004			ROR	R4
004610	000241			CLC	
004612	006004			ROR	R4
004614	000241			CLC	
004616	006004			ROR	R4
004620	005305			DEC	R5
004622	001362			BNE	3\$
004624	012703	007544		MOV	#MDATA,R3
004630	114023		4\$:	MOVB	-(R0),(R3)+
004632	105337	004712		DECB	CHRCNT
004636	001374			BNE	4\$
004640	105737	004713		TSTB	SPACNT
004644	001405			BEQ	6\$
004646	112723	000040	5\$:	MOVB	#040,(R3)+
004652	105337	004713		DECB	SPACNT
004656	001373			BNE	5\$
004660	105013		6\$:	CLRB	(R3)
004662	104402	007544		TYPE	,MDATA
004666	005337	004710		DEC	WRDCNT
004672	001313			BNE	1\$
004674	012605			MOV	(SP)+,R5
004676	012604			MOV	(SP)+,R4
004700	012603			MOV	(SP)+,R3
004702	012601			MOV	(SP)+,R1
004704	012600			MOV	(SP)+,R0
004706	000002			RTI	
004710	000000			WRDCNT:	0
004712	000000			CHRCNT:	0
	004713			SPACNT:	CHRCNT+1
004714	000000			BINWRD:	0

;TRAP DISPATCH SERVICE

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

004716	011646			.TRPSR: MOV	(SP),-(SP)	;GET PC OF RETURN
004720	162716	000002		SUB	#2,(SP)	;=PC OF TRAP
004724	017616	000000		MOV	@(SP),(SP)	;GET TRP
004730	006316			TRPOK: ASL	(SP)	;MULTIPLY TRAP ARG BY 2
004732	042716	177001		BIC	#177001,(SP)	;CLEAR UNWANTED BITS
004736	062716	001330		ADD	#.TRPTAB,(SP)	;POINTER TO SUBROUTINE ADDRESS
004742	017616	000000		MOV	@(SP),(SP)	;SUBROUTINE ADDRESS
004746	000136			JMP	@(SP)+	;GO TO SUBROUTINE

;ERROR HANDLER
;-----

004750	004737	007606		.HLT:	JSR	PC,CKSWR	;CHECK FOR SOFT SWR
004754	032777	010000	174220		BIT	#SW12,@SWR	;BELL ON ERROR?
004762	001406				BEQ	XBX	;BR IF NO BELL
004764	105777	174220			TSTB	@TPCSR	;TTY READY.
004770	100003				BPL	XBX	;DON'T WAIT IF TTY NOT READY.
004772	112777	000207	174212		MOVB	#207,@TPDBR	;PUSH A BELL AT THE TTY.
005000	032777	020000	174174	XBX:	BIT	#SW13,@SWR	;DELETE ERROR PRINT OUT?
005006	001105				BNE	HALTS	;BR IF NO PRINT OUT WANTED.
005010	021637	001234			CMP	(SP),LSTERR	;WAS THIS ERROR FOUND LAST TIME?
005014	001404				BEQ	1\$	;BR IF YES
005016	011637	001234			MOV	(SP),LSTERR	;RECORD BEING HERE
005022	105037	001325			CLRB	ERRFLG	;PREPARE HEADER
005026	104406			1\$:	SAVOS		;SAVE ALL PROC REGISTERS
005030	011605				MOV	(SP),R5	;GET THE PC OF ERROR
005032	162705	000002			SUB	#2,R5	;GET ADDRESS OF TRAP CALL
005036	011504				MOV	(R5),R4	;GET HLT INSTRUCTION
005040	006304				ASL	R4	;MULT BY TWO
005042	061504				ADD	(R5),R4	;DOUBLE IT
005044	006304				ASL	R4	;MULT AGAIN
005046	042704	177001			BIC	#177001,R4	;CLEAR JUNK
005052	062704	036362			ADD	#.ERRTAB,R4	;GET POINTER
005056	012437	005172			MOV	(R4)+,ERRMSG	;GET ERROR MESSAGE
005062	012437	005204			MOV	(R4)+,DATAHD	;GET DATA HEADRER
005066	011437	005216			MOV	(R4),DATABP	;GET DATA TABLE
005072	105737	001325			TSTB	ERRFLG	;TYPE HEADREER
005076	001403				BEQ	TYPMSG	;BR IF YES
005100	005737	005216			TST	DATABP	;DOES DATA TABLE EXIST?
005104	001040				BNE	TYPDAT	;BR IF YES.
005106	104402	005672		TYPMSG:	TYPE	,MCRLF	
005112	104402	005672			TYPE	,MCRLF	
005116	005737	001220			TST	LOCK	
005122	001402				BEQ	1\$	
005124	104402	006142			TYPE	,MASTEK	
005130	104402	006130		1\$:	TYPE	,MTSTN	
005134	104411	005330			CNVRT	,XTSTN	;SHOW IT
005140	104402	006217			TYPE	,MERRPC	;TYPE PC.
005144	104411	005322			CNVRT	,ERTABO	;SHOW IT
005150	104402	005672			TYPE	,MCRLF	;GIVE A CR/LF
005154	112737	177777	001325		MOVB	#-1,ERRFLG	;NO MORE HEADER UNLESS NO DATA TABLE.
005162	005737	005172			TST	ERRMSG	;IS THERE AN ERROR MESSAGE?
005166	001402				BEQ	WRKO.FM	;BR IF NO.


```

005170 104402
005172 000000
005174
005174 005737 005204
005200 001402
005202 104402
005204 000000
005206 005737 005216
005212 001402
005214 104410
005216 000000
005220 104407
005222 022737 003522 000042
005230 001403
005232 005777 173744
005236 100005
005240 010046
005242 016600 000002
005246 000000
005250 012600
005252 005237 001232
005256 032777 000400 173716
005264 001007
005266 032777 002000 173706
005274 001411
005276 013737 001216 001214
005304 012706 001200
005310 013701 001404
005314 000177 173674
005320 000002
005322 000001
005324 006 002
005326 001276
005330 000001
005332 003 002
005334 001226

ERRMSG: 0
WRKO.FM:
TST DATAHD
BEQ TYPDAT
TYPE
DATAHD: 0
TYPDAT: TST DATABP
BEQ RESREG
CONVRT
DATABP: 0
RESREG: RES05
HALTS: CMP ##ENDAD, @#42
BEQ 1$
TST @SWR
BPL EXITER
1$: PUSHRO
MOV 2(SP), R0
HALT
POPRO
EXITER: INC ERRCNT
BIT #SW08, @SWR
BNE 1$
BIT #SW10, @SWR
BEQ 2$
MOV NEXT, RETURN
1$: MOV #STACK, SP
MOV DMCSR, R1
JMP @RETURN
2$: RTI
ERTABO: 1
.BYTE 6, 2
SAVPC
XTSTN: 1
.BYTE 3, 2
TSTNO
;ENTER HERE ON POWER FAILURE
;-----

005336
005336 012737 005350 000024
005344 000000
005346 000777
.PFAIL:
MOV #RESTART, 24
HALT
BR
;SET UP FOR POWER UP TRAP
;HALT ON POWER DOWN NORMAL

;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED

005350
005350 012737 005336 000024
005356 012706 001200
005362 013701 001404
005366 005037 001416
005372 005237 001416
005376 001375
005400 104402 005675
005404 104411 005430
005410 105037 001325
RESTAR:
MOV #.PFAIL, 24
MOV #STACK, SP
MOV DMCSR, R1
CLR TEMP
INC TEMP
BNE -4
TYPE .MPFAIL
CNVRT .PFTAB
CLRB ERRFLG
;SET UP FOR POWER FAILURE
;RESET THE STACK POINTER
;RESTORE R1
;READY FOR TIMER
;PLUS ONE TO THE TIMER!
;BR IF MORE TO GO
;TYPE THE MESSAGE
;TELL WHAT TEST TO RETURN TO.
;START CLEAN

```



```

005414 005037 001234 CLR LSTERR ;*****
005420 005011 CLR (R1) ;CLEAR MAINT BITS
005422 104412 MSTCLR ;START CLEAN UP OF DEVICE
005424 000177 173564 JMP @RETURN ;START DOING THAT TEST AGAIN.
005430 000001 PFTAB: 1
005432 003 3,2
005434 001226 .BYTE TSTNO

005436 .DELAY:
005436 012777 000020 173746 MOV #20,@DMP04
005444 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
005446 121111 121111 ;POKE CLOCK DELAY BIT
005450 1# :
005450 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
005452 121224 121224 ;PORT4_IBUS*11
005454 032777 000020 173730 BIT #BIT4,@DMP04 ;IS CLOCK BIT SET?
005462 001772 BEQ 1# ;BR IF NO
005464 000002 RTI

005466 .MSTCLR:
005466 152777 000100 173712 BISB #BIT6,@DMCSRH ;SET MASTER CLEAR
005474 142777 000300 173704 BICB #BIT6!BIT7,@DMCSRH ;CLEAR MASTER CLEAR AND RUN
005502 000002 RTI ;RETURN

005504 .ROMCLK:
005504 152777 000002 173674 BISB #BIT1,@DMCSRH ;SET ROMI
005512 013677 173676 MOV @SP+,@DMP06 ;LOAD INSTRUCTION IN SEL6
005516 062746 000002 ADD #2,-(SP) ;ADJUST STACK
005522 032777 000100 173452 BIT #SW06,@SWR ;HALT IF SW06 =1
005530 001401 BEQ 1# ;BR IF SW06 =0
005532 000000 HALT ;HALT BEFORE CLOCKING INSTRUCTION
005534 152777 000003 173644 1# : BISB #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
005542 142777 000007 173636 BICB #BIT2!BIT1!BIT0,@DMCSRH ;CLEAR ROMO, ROMI, STEP
005550 000002 RTI

005552 .DATACLK:
005552 013637 001416 MOV @SP+,@TEMP ;PUT TICK COUNT IN TEMP
005556 062746 000002 ADD #2,-(SP) ;ADJUST STACK
005562 152777 000020 173616 1# : BISB #BIT4,@DMCSRH ;SET STEP LU
005570 027777 173610 173606 CMP @DMCSR,@DMCSR ;WASTE TIME
005576 142777 000020 173602 BICB #BIT4,@DMCSRH ;CLEAR STEP LU
005604 005337 001416 DEC TEMP ;DEC TICK COUNT
005610 001364 BNE 1# ;BR IF NOT DONE
005612 000002 RTI ;RETURN
005614 3# : .BLKW 1

005616 .TIMER:
005616 013637 001416 MOV @SP+,@TEMP ;MOVE COUNT TO TEMP
005622 062746 000002 ADD #2,-(SP) ;ADJUST STACK
005626 1# :
005626 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
005630 021364 021364 ;PORT4_IBUS* REG11
005632 032777 000002 173552 BIT #2,@DMP04 ;IS PGM CLOCK BIT CLEAR?
005640 001772 BEQ 1# ;BR IF YES
005642 2# :
005642 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
005644 021364 021364 ;PORT4_IBUS* REG11

```



```
005646 032777 000002 173536 BIT #2, @DMP04 ;IS PGM CLOCK BIT SET?
005654 001372 BNE 24 ;BR IF YES
005656 005337 001416 DEC TEMP ;DEC COUNT
005662 001361 BNE 14 ;BR IF NOT DONE
005664 000002 RTI ;RETURN
```

```
005666 040 040 077 MQM: .ASCIZ / ?/
005672 015 012 000 MCRLF: .ASCIZ <15><12>
005675 377 120 127 MPFAIL: .ASCIZ <377>/PWR FAILED. RESTART AT TEST /
005733 377 105 116 MEPASS: .ASCIZ <377>/END PASS CZDMC /
005755 377 122 000 MR: .ASCIZ <377>/R/
005760 377 116 117 MERR2: .ASCIZ <377>/NO DEVICES PRESENT./
006005 377 111 116 MERR3: .ASCIZ <377>/INSUFFICIENT DATA!/
006031 377 124 105 MTSTPC: .ASCIZ <377>/TEST PC-/
006043 377 114 117 MLOCK: .ASCIZ <377>/LOCK ON SELECTED TEST/
006072 103 123 122 MCSRX: .ASCIZ /CSR: /
006100 126 105 103 MVECX: .ASCIZ /VEC: /
006106 120 101 123 MPASSX: .ASCIZ /PASSES: /
006117 105 122 122 MERRX: .ASCIZ /ERRORS: /
006130 124 105 123 MTSTN: .ASCIZ /TEST NO: /
006142 052 000 MASTEK: .ASCIZ /*/
006144 377 123 105 MNEW: .ASCIZ <377>/SET SWITCH REG TO DMC11'S DESIRED ACTIVE./
006217 120 103 072 MERRPC: .ASCIZ /PC: /
006224 212 040 040 XHEAD: .ASCII <212>/ MAP OF DMC11 STATUS/
006263 377 040 040 .ASCII <377>/ -----/
006322 212 040 040 .ASCII <212>/ PC CSR STAT1 STAT2 STAT3/
006374 377 055 055 .ASCIZ <377>/----- ----- ----- -----/
006450 377 110 117 NUM: .ASCIZ <377>/HOW MANY DMC11'S TO BE TESTED?/
006510 377 103 123 CSR: .ASCIZ <377>/CSR ADDRESS?/
006526 377 126 105 VEC: .ASCIZ <377>/VECTOR ADDRESS?/
006547 377 102 122 PRIO: .ASCIZ <377>/BR PRIORITY LEVEL? (4,5,6,7)?/
006606 377 111 106 CRAM: .ASCIZ <377>/IF DMC HAS CRAM (M8204) TYPE "Y", IF CROM (M8200) TYPE "N" ?/
006704 377 127 110 MODU: .ASCIZ <377>/WHICH LINE UNIT? IF NONE TYPE "N", IF M8201 TYPE "1", IF M8202 TYPE "2"
```

"?/"

```
007016 377 123 127 LINE: .ASCIZ <377>/SWITCH PAC#1 (DDCMP LINE #)?/
007054 377 123 127 BM: .ASCIZ <377>/SWITCH PAC#2 (BM873 BOOT ADD)?/
007114 377 111 123 CONN: .ASCIZ <377>/IS THE LOOP BACK CONNECTOR ON?/
007154 377 116 117 NOACT: .ASCIZ <377>/NO DEVICES ARE SELECTED/
007205 377 012 123 SWMES: .ASCIZ <377><12>/SWR= /
007215 116 105 127 SWMES1: .ASCIZ /NEW? /
007223 377 377 104 CONERR: .ASCIZ <377><377>/DMC11 FOUND AT NON-STANDARD ADDRESS PC: /
007277 377 105 130 CNERR: .ASCIZ <377>/EXPECTED FOUND/
007320 040 050 104 DMCM: .ASCIZ / (DMC) /
007330 040 050 113 KMCM: .ASCIZ / (KMC) /
007340 377 104 115 SPEED: .ASCIZ <377>/DMC11-AR(REMOTE,LOW SPEED) OR DMC11-AL(LOCAL,HIGH SPEED) TYPE "R" OR "
```

L" ?/"

```
007454 000005 .EVEN
007456 006 XSTATQ: 5
007460 001246 003 .BYTE 6,3
007462 006 003 .BYTE 6,3
007464 001250 .TEMP2
007466 006 003 .BYTE 6,3
007470 001252 .TEMP3
007472 006 003 .BYTE 6,3
007474 001254 .TEMP4
007476 006 002 .BYTE 6,2
007500 001256 .TEMP5
```

.EVEN

;BUFFERS FOR INPUT-OUTPUT

007502 000000 INBUF: 0
007544 007544 .+.40
007544 000000 MDATA: 0
007606 007606 .+.40

;ROUTINE USED TO CHANGE SOFTWARE SWITCH
;REGISTER USING THE CONSOLE TERMINAL
;-----

007606	022737	000176	001202	CKSWR:	CMP	#SWREG,SWR	;IS THE SOFT SWR BEING USED?
007614	001077				BNE	CKSWR5	;BR IF NO
007616	105777	171362			TSTB	@TKCSR	;IS DONE SET?
007622	100003				BPL	2	;GO ON IF NOT SET
007624	012737	177777	003734		MOV	#-1,DONE	;IF DONE SET, SET FLAG
007632	022777	000007	171346	2:	CMP	#7,@TKDBR	;WAS CTRL G TYPED? (7 BIT ASCII)
007640	001404				BEQ	1	;BR IF YES
007642	022777	000207	171336		CMP	#207,@TKDBR	;WAS CTRL G TYPED? (8 BIT ASCII)
007650	001061				BNE	CKSWR5	;BR IF NO
007652	010246			1:	MOV	R2,-(SP)	;STORE R2
007654	010346				MOV	R3,-(SP)	;STORE R3
007656	010446				MOV	R4,-(SP)	;STORE R4
007660	012737	177777	010016		MOV	#-1,SWFLG	;SET SOFT TYPE OUT FLAG
007666	005002			CKSWR1:	CLR	R2	;CLEAR NEW SWR CONTENTS
007670	012704	177777			MOV	#-1,R4	;SET FLAG TO ALL ONES
007674	104402	007205			TYPE	,SWMES	;TYPE "SWR= "
007700	104411			CKSWR2:	CNVRT		;TYPE OUT PRESENT CONTENTS
007702	010052				SOFTSW		;OF SOFT SWITCH REGISTER
007704	104402	007215		CKSWR3:	TYPE	,SWMES1	;TYPE "NEW? "
007710	004737	010020		CKSWR4:	JSR	PC,INCHAR	;GET RESPONSE
007714	022703	000015			CMP	#15,R3	;WAS IT A CR?
007720	001424				BEQ	5	;BR IF YES
007722	022703	000012			CMP	#12,R3	;WAS IT A LF?
007726	001416				BEQ	4	;BR IF YES
007730	022703	000025			CMP	#25,R3	;WAS IT CTRL U?
007734	001754				BEQ	CKSWR1	;BR IF YES(START OVER)
007736	022703	000007			CMP	#7,R3	;IF CNTL G GET NEXT CHAR
007742	001762				BEQ	CKSWR4	
007744	005004				CLR	R4	;IT MUST BE A DIGIT SO CLR FLAG
007746	042703	177770			BIC	#177770,R3	;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
007752	006302				ASL	R2	;SHIFT R2 3 TIMES
007754	006302				ASL	R2	
007756	006302				ASL	R2	
007760	050302				BIS	R3,R2	;ADD LAST DIGIT
007762	000752				BR	CKSWR4	;GET NEXT CHARACTER
007764	012766	002002	000006	4:	MOV	#.START,6(SP)	;LF WAS TYPED SO GO TO START
007772	005704			5:	TST	R4	;IS FLAG CLEAR?
007774	001002				BNE	6	;IF NOT DON'T CHANGE SOFT SWR
007776	010277	171200			MOV	R2,@SWR	;IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
010002	005037	010016		6:	CLR	SWFLG	;CLEAR TYPEOUT FLAG
010006	012604				MOV	(SP)+,R4	;RESTORE R4
010010	012603				MOV	(SP)+,R3	;RESTORE R3
010012	012602				MOV	(SP)+,R2	;RESTORE R2
010014	000207			CKSWR5:	RTS	PC	;RETURN

010016	000000		SWFLG:	0	
010020	105777	171160	INCHAR:	TSTB	@TKCSR
010024	100375			BPL	.-4
010026	017703	171154		MOV	@TKDBR,R3
010032	105777	171152		TSTB	@TPCSR
010036	100375			BPL	.-4
010040	010377	171146		MOV	R3,@TPDBR
010044	042703	000200		BIC	#BIT7,R3
010050	000207			RTS	PC
010052	000001		SOFTSW:	1	
010054	006	002		.BYTE	6,2
010056	000176			SWREG	


```
;
;ROUTINE USED TO "CYCLE" THROUGH UP TO 16 DMC11'S
;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
;AND RUNS THE SPECIFIED DMC11'S.  THIS ROUTINE *MUST*
;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
;SETUP NECESSARY.
;
```

010060	005737	001306		CYCLE:	TST	DMACTV		;ARE ANY DMC11'S TO BE TESTED?
010064	001004				BNE	1\$		;BR IF OK.
010066	104402	007154			TYPE	,NOACT		;NO DMC11'S SELECTED!!
010072	000000				HALT			;STOP THE SHOW.
010074	000776				BR	.-2		;DISQUALIFY CONT. SW.
010076	000241			1\$:	CLC			;CLEAR PROC. CARRY BIT.
010100	006137	001316			ROL	RUN		;UPDATE POINTER
010104	005537	001316			ADC	RUN		;CATCH CARRY FROM RUN
010110	062737	000004	001322		ADD	#4,MILK		;UPDATE POINTER
010116	062737	000010	001320		ADD	#10,CREAM		;UPDATE ADDRESS POINTER.
010124	022737	001700	001320		CMP	#DM.MAP+200,CREAM		
010132	001006				BNE	2\$		;KEEP GOING; NOT ALL TESTED FOR.
010134	012737	001500	001320		MOV	#DM.MAP,CREAM		;RESET ADDRESS POINTER.
010142	012737	001702	001322		MOV	#CNT.MAP,MILK		;RESET PASS COUNT POINTER
010150	033737	001316	001306	2\$:	BIT	RUN,DMACTV		;IS THIS ONE ACTIVE?
010156	001747				BEQ	1\$		;BR IF NO
010160	013700	001320			MOV	CREAM,R0		;GET ADDRESS POINTER
010164	013702	001322			MOV	MILK,R2		;GET PASS COUNT POINTER
010170	012037	001404			MOV	(R0)+,DMCSR		;LOAD SYSTEM CTRL. REG
010174	011037	001374			MOV	(R0)+,DMRVEC		;LOAD VECTOR
010200	042737	177000	001374		BIC	#177000,DMRVEC		;CLEAR UNWANTED BITS
010206	012037	001366			MOV	(R0)+,STAT1		;LOAD STAT1
010212	012037	001370			MOV	(R0)+,STAT2		;LOAD STAT2
010216	012037	001372			MOV	(R0)+,STAT3		;LOAD STAT3
010222	012237	001230			MOV	(R2)+,PASCNT		;LOAD PASS COUNT
010226	012237	001232			MOV	(R2)+,ERRCNT		;LOAD ERROR COUNT
010232	012700	000002			MOV	#2,R0		;SAVE CORE THIS WAY!
010236	013737	001404	001406		MOV	DMCSR,DMCSRH		
010244	005237	001406			INC	DMCSRH		
010250	013737	001406	001410		MOV	DMCSRH,DMCTL		
010256	005237	001410			INC	DMCTL		
010262	013737	001410	001412		MOV	DMCTL,DMP04		
010270	060037	001412			ADD	R0,DMP04		
010274	013737	001412	001414		MOV	DMP04,DMP06		
010302	060037	001414			ADD	R0,DMP06		
010306	013737	001374	001376		MOV	DMRVEC,DMRLVL		;PTY LVL
010314	060037	001376			ADD	R0,DMRLVL		
010320	013737	001376	001400		MOV	DMRLVL,DMTVEC		;TX VEC
010326	060037	001400			ADD	R0,DMTVEC		
010332	013737	001400	001402		MOV	DMTVEC,DMTLVL		;TX LVL
010340	060037	001402			ADD	R0,DMTLVL		
010344	032737	000002	001236		BIT	#SW01,STRTSW		;IS TEST NO. SELECTED
010352	001450				BEQ	7\$		;BR IF NO
010354				4\$:				
010354	005737	000042			TST	#42		;RUNNING IN AUTO MODE?
010360	001045				BNE	7\$		;BR IF YES

010362	104402	005672		TYPE	.MCRLF	
010366	104403			INSTR		;GET TEST NO.
010370	006130			MTSTN		
010372	104405			PARAM		
010374	000001			1		
010376	001000			1000		
010400	001226			TSTNO		
010402	000			0		
010403	001			.BYTE		
010404	012700	012320		.BYTE		
010410	022710			5\$: MOV	@TST1,R0	
010412	012737			CMP	(PC)+,(R0)	;CMP FIRST WORD TO 12737
010414	001020			MOV	(PC)+,@(PC)+	
010416	023760	001226	000002	BNE	6\$	;BR IF NOT SAME
010424	001014			CMP	TSTNO,2(R0)	;DOES TSTNO MATCH?
010426	022760	001226	000004	BNE	6\$	;BR IF NO
010434	001010			CMP	@TSTNO,4(R0)	;IS LAST WORD OK?
010436	010037	001214		BNE	6\$	;BR IF NO
010442	104402	005755		MOV	RO,RETURN	;IT IS A LEGAL TEST SO DO IT
010446	042737	000002	001236	TYPE	.MR	
010454	000412			BIC	@SW01,STRTSW	
010456	005720			BR	8\$	
010460	020027	033776		6\$: TST	(R0)+	;POP RO
010464	001351			CMP	RO,@TLAST+10	;AT END YET?
010466	104402	005666		BNE	5\$	;BR IF NO
010472	000730			TYPE	.MQM	;YES ILLEGAL TEST NO.
				BR	4\$	;TRY AGAIN
010474	012737	012320	001214	7\$: MOV	@TST1,RETURN	;PREPARE RETURN ADDRESS
010502	013701	001404		8\$: MOV	DMCSR,R1	;R1 = BASE DMC11 ADDRESS
010506	000177	170502		JMP	@RETURN	;GO START TESTING.

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

010512				AUTO.SIZE:		
010512	000005			RESET		;INSURE A BUS INIT.
010514	012702	001500		CSRMAP: MOV	@DM.MAP,R2	;LOAD MAP POINTER.
010520	005022			1\$: CLR	(R2)+	;ZERO ENTIRE MAP
010522	022702	001700		CMP	@DM.END,R2	;ALL DONE?
010526	001374			BNE	1\$	;BR IF NO
010530	005037	001310		CLR	DMNUM	;SET OCTAL NUMBER OF DMC11'S TO 0
010534	012702	001500		MOV	@DM.MAP,R2	;R2 POINTS TO DMC MAP
010540	005037	001306		CLR	DMACTV	;CLEAR ACTIVE
010544	032737	000001	001236	BIT	@SW00,STRTSW	;QUESTIONS?
010552	001002			BNE	.+6	;BR IF YES
010554	000137	011252		JMP	7\$	;IF NO SKIP QUESTIONS
010560	012737	000001	001256	MOV	@1,TEMP5	;START WITH 1
010566	104403			INSTR		
010570	006450			NUM		
010572	104405			PARAM		
010574	000001			1		

010576	000020		16.		
010600	001252		TEMP3		
010602	000		.BYTE	0	
010603	001		.BYTE	1	
010604	013737	001252 001310	MOV	TEMP3,DMNUM	;DMNUM = HOW MANY
010612	104402	005672 12\$:	TYPE	.MCRLF	
010616	104410		CONVRT		;TYPE WHICH DMC IS BEING DONE
010620	012002		WHICH		;TEMP5 IS WHICH DMC
010622	005237	001256	INC	TEMP5	
010626	104403		INSTR		
010630	006510		CSR		
010632	104405		PARAM		
010634	160000		160000		
010636	164000		164000		
010640	001254		TEMP4		
010642	000		.BYTE	0	
010643	001		.BYTE	1	
010644	013722	001254	MOV	TEMP4,(R2)+	;STORE CSR IN MAP
010650	104403		INSTR		
010652	006526		VEC		
010654	104405		PARAM		
010656	000000		0		
010660	000776		776		
010662	001254		TEMP4		
010664	000		.BYTE	0	
010665	001		.BYTE	1	
010666	013712	001254	MOV	TEMP4,(R2)	;STORE VECTOR IN MAP
010672	104402	10\$:	TYPE		
010674	006547		PRI0		;ASK WHAT BR LEVEL
010676	004737	012266	JSR	PC,INTTY	;GET RESPONSE
010702	022703	000024	CMP	#24,R3	
010706	101014		BHI	50\$	;BR IF LESS THAN 4
010710	022703	000027	CMP	#27,R3	
010714	103411		BLO	50\$	;BR IF GREATER THAN 7
010716	012704	000011	MOV	#11,R4	;R4 = NUMBER OF SHIFTS
010722	006303		ASL	R3	;SHIFT R3 LEFT
010724	005304		DEC	R4	;DEC SHIFT COUNT
010726	001375		BNE	.-4	;BR IF NOT DONE
010730	042703	170777	BIC	#170777,R3	;BIC UNWANTED BITS
010734	050312		BIS	R3,(R2)	;PUT BR LEVEL IN STATUS MAP
010736	000403		BR	8\$	;CONTINUE
010740	104402	50\$:	TYPE		
010742	005666		MQM		;RESPONSE IS OUT OF LIMITS
010744	000752		BR	10\$	;TRY AGAIN
010746	104402	8\$:	TYPE		
010750	006606		CRAM		;DOES DMC HAVE CRAM?
010752	004737	012266	JSR	PC,INTTY	;GET REPLY
010756	022703	000131	CMP	#131,R3	
010762	001427		BEQ	9\$	;YES
010764	022703	000116	CMP	#116,R3	;NO
010770	001403		BEQ	40\$	;NOT A Y OR N
010772	104402		TYPE		
010774	005666		MQM		;TYPE "?"
010776	000763		BR	8\$	;ASK AGAIN
011000	104402	40\$:	TYPE		
011002	007340		SPEED		;DMC11-AR OR DMC11-AL?
011004	004737	012266	JSR	PC,INTTY	;GET RESPONSE

011010	022703	000122		CMP	#122,R3	;IS IT R
011014	001414			BEQ	16\$	;BR IF REMOTE
011016	022703	000114		CMP	#114,R3	;IS IT L
011022	001403			BEQ	41\$	;BR IF LOCAL
011024	104402			TYPE		
011026	005666			MQM		
011030	000763			BR	40\$	;TRY AGAIN
011032	052762	000002	000004	41\$: BIS	#BIT1,4(R2)	;SET BIT1 IN STAT3
011040	000402			BR	16\$	;CONTINUE
011042	052712	100000		9\$: BIS	#BIT15,(R2)	;SET BIT 15 IF CRAM
011046	104402			16\$: TYPE		
011050	006704			MODU		;ASK WHICH LINE UNIT
011052	004737	012266		JSR	PC,INTTY	;GET REPLY
011056	022703	000021		CMP	#21,R3	; "1"
011062	001417			BEQ	30\$	
011064	022703	000022		CMP	#22,R3	; "2"
011070	001412			BEQ	31\$	
011072	022703	000116		CMP	#116,R3	; "N"
011076	001403			BEQ	32\$	
011100	104402			TYPE		
011102	005666			MQM		;IF NOT A 1.2 OR N TYPE "?"
011104	000760			BR	16\$	;TRY AGIAN
011106	052722	010000		32\$: BIS	#BIT12,(R2)+	;SET BIT 12 IN STAT2 IF NO LU
011112	022222			CMP	(R2)+,(R2)+	;POP OVER STAT2 AND STAT3
011114	000447			BR	33\$	
011116	052712	020000		31\$: BIS	#BIT13,(R2)	;SET BIT 13 IN STAT2 IF M8202
011122	104402			30\$: TYPE		
011124	007114			CONN		;ASK IF LOOP-BACK IS ON
011126	004737	012266		JSR	PC,INTTY	;GET REPLY
011132	022703	000131		CMP	#131,R3	;Y
011136	001406			BEQ	17\$	
011140	022703	000116		CMP	#116,R3	;N
011144	001406			BEQ	18\$	
011146	104402			TYPE		
011150	005666			MQM		;IF NOT Y OR N TYPE "?"
011152	000763			BR	30\$	;TRY AGAIN
011154	052722	040000		17\$: BIS	#BIT14,(R2)+	;TURNAROUND IS CONNECTED
011160	000402			BR	19\$	
011162	042722	040000		18\$: BIC	#BIT14,(R2)+	;NO TURNAROUND
011166				19\$: INSTR		
011166	104403			LINE		
011170	007016			PARAM		
011172	104405			0		
011174	000000			377		
011176	000377			TEMP4		
011200	001254			.BYTE	0	
011202	000			.BYTE	1	
011203	001			MOVB	TEMP4,(R2)+	;STORE SWITCH PAC IN MAP
011204	113722	001254		INSTR		
011210	104403			BM		
011212	007054			PARAM		
011214	104405			0		
011216	000000			377		
011220	000377			TEMP4		
011222	001254			.BYTE	0	
011224	000			.BYTE	1	
011225	001					

011226	113722	001254		MOVB	TEMP4,(R2)+	;STORE SWITCH PAC IN MAP
011232	005722			TST	(R2)+	;POP OVER STAT3
011234	005337	001252	33\$:	DEC	TEMP3	;DEC DMC COUNT
011240	001402			BEQ	34\$	;BR IF DONE
011242	000137	010612		JMP	12\$	;JUMP IF NOT
011246	000137	011702	34\$:	JMP	13\$	;CONTINUE
011252	012701	160000	7\$:	MOV	#160000,R1	;SET FOR FIRST ADDRESS TO BE TESTED
011256	012737	011774	000004	MOV	#6\$,0#4	;SET FOR NON-EXISTANT DEVICE TIME OUT
011264	005011		2\$:	CLR	(R1)	;CLEAR SEL0
011266	005711			TST	(R1)	;IF DMC11 DMCSR S/B 0
011270	001172			BNE	3\$	;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO DMC11
011272	005061	000006		CLR	6(R1)	;CLEAR SEL6
011276	005761	000006		TST	6(R1)	;IF DMC11 THEN DMRIC S/B =0!
011302	001165			BNE	3\$	;BR IF NOT DMC11
011304	012711	002000		MOV	#BIT10,(R1)	;SET ROM0
011310	005061	000004		CLR	4(R1)	;CLEAR SEL4
011314	012761	125252	000006	MOV	#125252,6(R1)	;WRITE THIS TO SEL6
011322	052711	020000		BIS	#BIT13,(R1)	;WRITE IT!
011326	022761	125252	000004	CMP	#125252,4(R1)	;WAS IT WRITTEN?
011334	001004			BNE	21\$	;IF NO IT IS NOT CRAM
011336	052762	100000	000002	BIS	#BIT15,2(R2)	;SET BIT15 IF CRAM
011344	000431			BR	22\$	
011346	012711	001000	21\$:	MOV	#BIT9,(R1)	;SET ROM1
011352	012761	100430	000006	MOV	#100430,6(R1)	;PUT INSTRUCTION IN SEL6
011360	012711	001400		MOV	#BIT9!BIT8,(R1)	;CLOCK INSTRUCTION (MICRO PROC PC TO 0)
011364	012711	002000		MOV	#BIT10,(R1)	;SET ROM0
011370	022761	016472	000006	CMP	#016472,6(R1)	;IS IT LOCAL CROM?
011376	001411			BEQ	23\$	;BR IF YES
011400	022761	016461	000006	CMP	#016461,6(R1)	;IS IT REMOTE CROM?
011406	001410			BEQ	22\$	;BR IF YES
011410	022761	177777	000006	CMP	#-1,6(R1)	;NO CROM?
011416	001404			BEQ	22\$	;BR IF YES
011420	000516			BR	3\$	;NOT A DMC
011422	052762	000002	000006	23\$:	BIS	#BIT1,6(R2)
						;SET BIT 1 IN STAT3
011430	010122			22\$:	MOV	R1,(R2)+
						;STORE CSR IN CORE TABLE.
011432	012711	001000	15\$:	MOV	#BIT9,(R1)	;CLEAR LINE UNIT LOOP
011436	005061	000004		CLR	4(R1)	;CLEAR PORT4
011442	012761	122113	000006	MOV	#122113,6(R1)	;LOAD INSTRUCTION (CLR DTR)
011450	052711	000400		BIS	#BIT8,(R1)	;CLOCK INSTRUCTION
011454	012761	021264	000006	MOV	#021264,6(R1)	;LOAD INSTRUCTION
011462	052711	000400		BIS	#BIT8,(R1)	;CLOCK INSTRUCTION
011466	122761	000377	000004	CMPB	#377,4(R1)	;IS IT ALL ONES?
011474	001003			BNE	.+10	;BR IF NO
011476	052712	010000		BIS	#BIT12,(R2)	;IF YES, NO LINE UNIT, SET STATUS BIT
011502	000436			BR	20\$	
011504	032761	000002	000004	BIT	#BIT1,4(R1)	;IS SWITCH A ONE?
011512	001403			BEQ	.+10	;BR IF M8201
011514	052712	060000		BIS	#BIT13!BIT14,(R2)	;M8202 ASSUME CONNECTOR
011520	000427			BR	20\$	;CONNECTOR ON)
011522	032761	000010	000004	BIT	#BIT3,4(R1)	;IS MRDY SET
011530	001023			BNE	20\$	;BR IF M8201 NO CONNECTOR (ON LINE)
011532	012761	000100	000004	MOV	#BIT6,4(R1)	;LOAD PORT4
011540	012761	122113	000006	MOV	#122113,6(R1)	;LOAD INSTRUCTION
011546	052711	000400		BIS	#BIT8,(R1)	;CLOCK INSTRUCTION(SET DTR)
011552	012761	021264	000006	MOV	#021264,6(R1)	;LOAD INSTRUCTION
011560	052711	000400		BIS	#BIT8,(R1)	;CLOCK INSTRUCTION(READ MODEM REG)

011564	032761	000010	000004	BIT	#BIT3,4(R1)	;IS MRDY SET NOW?
011572	001402			BEQ	20\$	;BR IF NO CONNECTOR
011574	052712	040000		BIS	#BIT14,(R2)	;SET STATUS BIT FOR CONNECTOR
011600	005722			TST	(R2)+	;POP POINTER
011602	012761	021324	000006	MOV	#021324,6(R1)	;PUT INSTRUCTION IN PORT6
011610	012711	001400		MOV	#BIT9!BIT8,(R1)	;PORT4_LU 15
011614	156122	000004		BISB	4(R1),(R2)+	;STORE DDCMP LINE # IN TABLE
011620	012761	021344	000006	MOV	#021344,6(R1)	;PORT6_INSTRUCTION
011626	012711	001400		MOV	#BIT8!BIT9,(R1)	;CLOCK INSTR.
011632	156122	000004		BISB	4(R1),(R2)+	;STORE BM873 ADD IN TABLE
011636	005722			TST	(R2)+	;POP OVER STAT3
011640	005011			CLR	(R1)	;CLEAR ROMI
011642	005237	001310		INC	DMNUM	;UPDATE DEVICE COUNTER
011646	022737	000020	001310	CMP	#20,DMNUM	;ARE MAX. NO. OF DEV FOUND?
011654	001412			BEQ	13\$	;YES DON'T LOOK FOR ANY MORE.
011656	005011			CLR	(R1)	;CLEAR BIT 10
011660	005061	000006		CLR	6(R1)	;CLEAR SEL 6
011664	062701	000010		ADD	#10,R1	;UPDATE CSR POINTER ADDRESS
011670	022701	164000		CMP	#164000,R1	
011674	001402			BEQ	13\$	;BR IF DONE
011676	000137	011264		JMP	2\$	;JUMP IF NOT
011702	005037	001306		CLR	DMACTV	
011706	005737	001310		TST	DMNUM	;WERE ANY DMC11'S FOUND AT ALL?
011712	001423			BEQ	5\$	;ERROR AUTO SIZER FOUND NO DMC11'S IN THIS SYS.
011714	013701	001310		MOV	DMNUM,R1	
011720	010137	001314		MOV	R1,SAVNUM	;SAVE NUMBER OF DEVICES
011724	000241			CLC		
011726	006137	001306		ROL	DMACTV	;GENERATE ACTIVE REGISTER OF DEVICES.
011732	005237	001306		INC	DMACTV	;SET THE BIT
011736	005301			DEC	R1	
011740	001371			BNE	4\$	;BR IF MORE TO GENERATE
011742	012737	000006	000004	MOV	#6,0#4	;RESTORE TRAP VECTOR
011750	013737	001306	001312	MOV	DMACTV,SAVACT	;SAVE ACTIVE REGISTER
011756	000137	012010		JMP	VECMAP	;GO FIND THE VECTOR NOW.
011762	104402	005760		TYPE	,MERR2	;NOTIFY OPR THAT NO DMC11'S FOUND.
011766	005000			CLR	RO	;MAKE DATA LIGHTS ZERO
011770	000000			HALT		;STOP THE SHOW
011772	000776			BR	.-2	;DISABLE CONT. SW.
011774	012716	011664		MOV	#14\$,(SP)	;ENTERED BY NON-EXISTANT TIME-OUT.
012000	000002			RTI		;RETURN TO MAINSTREAM
012002	000001			WHICH:	1	
012004	002	002		.BYTE	2,2	
012006	001256			TEMP5		
012010	032737	000001	001236	VECMAP:	BIT	#SW00,STRTSW
012016	001114			BNE	5\$	
012020	012737	000340	000022	MOV	#340,0#22	;SET IOT TRAP PRIO TO 7
012026	012737	012202	000020	MOV	#4\$,0#20	;SET IOT TRAP VECTOR
012034	012702	001500		MOV	#DM.MAP,R2	;SET SOFTWARE POINTER
012040	012700	000300		MOV	#300,RO	;FLOATING VECTORS START HERE.
012044	012701	000302		MOV	#302,R1	;PC OF IOT INSTR.
012050	010120			MOV	R1,(R0)+	;START FILLING VECTOR AREA
012052	012721	000004		MOV	#4,(R1)+	;WITH .+2; IOT
012056	022021			CMP	(R0)+,(R1)+	;ADD 2 TO RO +R1
012060	020127	001000		CMP	R1,#1000	
012064	101771			BLOS	1\$	;BR IF MORE TO FILL


```

012066 013737 001306 001246      MOV    DMACTV,TEMP1    ;STORE TEMPORALLY
012074 006037 001246      ROR     TEMP1      ;BRING OUT A BIT
012100 103063              BCC     5$      ;BR IF ALL DONE
012102 012704 000012      MOV     #12,R4      ;R4 IS INDEX REGISTER
012106 016437 012252 177776      MOV     BRLVL(R4),PS    ;SET PS TO 7
012114 011201              MOV     (R2),R1
012116 012761 000200 000004      MOV     #200,4(R1)
012124 012711 001000      MOV     #BIT9,(R1)    ;SET ROMI
012130 012761 121111 000006      MOV     #121111,6(R1) ;PUT INSTRUCTION IN PORT6
012136 012711 001400      MOV     #BIT9:BIT8,(R1) ;FORCE AN INTERRUPT
012142 105200              INCB    R0          ;STALL
012144 001376              BNE     .-2        ;FOR TIME TO INTERRUPT
012146 162704 000002      SUB     #2,R4      ;GET NEXT LOWEST PS LEVEL
012152 001404              BEQ     6$      ;BR IF R4 = 0
012154 016437 012252 177776      MOV     BRLVL(R4),PS    ;MOVE NEXT LOWER LEVEL IN PS
012162 000767              BR      7$      ;BR TO DELAY
012164 052762 005300 000002 6$:  BIS     #5300,2(R2)    ;NO INTERRUPT ASSUME 300 AT LEVEL 5 AND FIX DMC11 LATER
012172 005011 3$:  CLR     (R1)          ;CLEAR ROMI
012174 062702 000010      ADD     #10,R2      ;POP SOFTWARE POINTER
012200 000735              BR      2$      ;KEEP GOING
012202 051662 000002 4$:  BIS     (SP),2(R2)    ;GET VECTOR ADDRESS
012206 042762 000007 000002      BIC     #7,2(R2)    ;CLEAR JUNK
012214 016405 012254      MOV     BRLVL+2(R4),R5 ;GET BR LEVEL OF DMC11
012220 006305              ASL     R5          ;SHIFT LEVEL 4 PLACES
012222 006305              ASL     R5          ;TO THE LEFT FOR THE
012224 006305              ASL     R5          ;STATUS TABLE
012226 006305              ASL     R5
012230 042705 170777      BIC     #170777,R5    ;CLEAR UNWANTED BITS
012234 050562 000002      BIS     R5,2(R2)    ;PUT BR LEVEL IN STATUS TABLE
012240 022626              CMP     (SP)+,(SP)+ ;POP IOT JUNK OFF STACK
012242 012716 012172      MOV     #3$,(SP)    ;SET FOR RETURN
012246 000002              RTI
012250 000207 5$:  RTS     PC          ;ALL DONE WITH "AUTO SIZING"

012252 000000      BRLVL:  0          ;LEVEL 0
012254 000000              0          ;LEVEL 0
012256 000200              200        ;LEVEL 4
012260 000240              240        ;LEVEL 5
012262 000300              300        ;LEVEL 6
012264 000340              340        ;LEVEL 7

012266 105777 166712      INTTY:  TSTB    #TKCSR      ;WAIT FOR DONE
012272 100375              BPL     .-4
012274 017703 166706      MOV     #TKDBR,R3    ;PUT CHAR IN R3
012300 105777 166704      TSTB    #TPCSR      ;WAIT UNTIL PRINTER IS READY
012304 100375              BPL     .-4
012306 010377 166700      MOV     R3,#TPDBR    ;ECHO CHAR
012312 042703 000240      BIC     #BIT7:BIT5,R3 ;MASK OFF LOWER CASE
012316 000207              RTS     PC          ;RETURN

```

```

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

```



```

; TEST 1
;-----
012320 012737 000001 001226 TST1: MOV #1,TSTNO
012326 012737 012426 001216 MOV #TST2,NEXT
012334 012737 012366 001220 MOV #1$,LOCK

012342 013701 001404 MOV DMCSR,R1 ;R1 CONTAINS BASE DMC11 ADDRESS
012346 012700 000004 MOV #4,R0 ;R1 CONTAINS BASE DMC11 ADDRESS
012352 012737 012420 000004 MOV #2$,4 ;4 REGISTERS TO BE TESTED
012360 012737 000340 000006 MOV #340,6 ;SET UP TIMEOUT TRAP
012366 005711 1$: TST (R1) ;LEVEL 7
012370 000240 NOP ;REFERENCE DEVICE REGISTER
012372 104401 SCOP1 ;SW09=1?
012374 062701 000002 ADD #2,R1 ;NEXT REGISTER
012400 005300 DEC R0 ;DEC REGISTER COUNT
012402 001371 BNE 1$ ;BR IF NOT LAST REGISTER
012404 012737 000006 000004 MOV #6,4 ;RESTORE LOC 4
012412 005037 000006 CLR 6 ;RESTORE LOC 6
012416 104400 SCOPE ;SCOPE THIS TEST
012420 011602 2$: MOV (SP),R2 ;GET PC OF TRAP
012422 104001 EMT 1 ;TIME-OUT ERROR
012424 000002 RTI

```

8

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

```

; TEST 2
;-----
012426 012737 000002 001226 TST2: MOV #2,TSTNO
012434 012737 012456 001216 MOV #TST3,NEXT

012442 005011 CLR (R1) ;R1 CONTAINS BASE DMC11 ADDRESS
012444 005005 CLR R5 ;CLEAR DMCSR
012446 011104 MOV (R1),R4 ;CLEAR "EXPECTED"
012450 001401 BEQ 1$ ;PUT DMCSR IN "FOUND"
012452 104002 EMT 2 ;BR IF CLEARED
012454 104400 1$: SCOPE ;ERROR DMCSR NOT CLEARED
;SCOPE THIS TEST

```

9

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

```

; TEST 3
;-----
012456 012737 000003 001226 TST3: MOV #3,TSTNO
012464 012737 012606 001216 MOV #TST4,NEXT
012472 012737 012506 001220 MOV #1$,LOCK

012500 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
012502 012700 000001 MOV #1,R0 ;MASTER CLEAR DMC11
012506 005011 1$: CLR (R1) ;START PATTERN AT 1
;CLEAR REGISTER

```


012510	010005			MOV	R0,R5	;PUT DATA IN "EXPECTED"
012512	010011			MOV	R0,(R1)	;WRITE DMC REGISTER WITH PATTERN
012514	011104			MOV	(R1),R4	;READ DMC REGISTER INTO "FOUND"
012516	020504			CMP	R5,R4	;IS DATA CORRECT
012520	001401			BEQ	2\$	;BR IF YES
012522	104002			EMT	2	;DATA ERROR
012524	104401		2\$:	SCOP1		;SW09=1?
012526	005721			TST	(R1)+	;NEXT REGISTER
012530	005200			INC	R0	;INCREMENT DATA PATTERN
012532	022700	000005		CMP	#5,R0	;LAST REGISTER?
012536	001363			BNE	1\$	;BR IF NO
012540	013701	001404		MOV	DMCSR,R1	;BASE DMC11 ADDRESS TO R1
012544	012700	000001		MOV	#1,R0	;RESTART PATTERN AT 1
012550	012737	012556	001220	MOV	#3\$,LOCK	;NEW SCOP1
012556	010005			MOV	R0,R5	;PUT DATA IN "EXPECTED"
012560	011104			MOV	(R1),R4	;READ DMC REGISTER INTO "FOUND"
012562	020504			CMP	R5,R4	;IS DATA CORRECT
012564	001401			BEQ	4\$	;BR IF YES
012566	104002			EMT	2	;DUAL ADDRESSING ERROR
012570	104401		4\$:	SCOP1		;SW09=1?
012572	005721			TST	(R1)+	;NEXT REGISTER
012574	005200			INC	R0	;INCREMENT PATTERN
012576	022700	000005		CMP	#5,R0	;LAST REGISTER?
012602	001365			BNE	3\$	;BR IF NO
012604	104400			SCOPE		;SCOPE THIS TEST

12

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

				; TEST 4		
012606	012737	000004	001226	TST4:	MOV	#4,TSTNO
012614	012737	012704	001216		MOV	#TST5,NEXT
012622	012737	012632	001220		MOV	#1\$,LOCK
012630	104412				MSTCLR	
012632	013701	001404		1\$:	MOV	DMCSR,R1
012636	012705	000001			MOV	#BIT0,R5
012642	010511				MOV	R5,(R1)
012644	011104				MOV	(R1),R4
012646	020504				CMP	R5,R4
012650	001401				BEQ	2\$
012652	104002				EMT	2
012654	104401			2\$:	SCOP1	
012656	012737	012664	001220		MOV	#3\$,LOCK
012664	042711	000001		3\$:	BIC	#BIT0,(R1)
012670	005005				CLR	R5
012672	011104				MOV	(R1),R4
012674	001402				BEQ	4\$
012676	104002				EMT	2
012700	104401				SCOP1	
012702	104400			4\$:	SCOPE	

;MASTER CLEAR DMC11
;PUT REGISTER ADDRESS IN R1
;PUT DATA IN "EXPECTED"
;WRITE BIT 0
;READ CONTROL STATUS REGISTER
;IS DATA CORRECT
;BR IF YES
;DATA ERROR
;SW09 UP?
;NEW SCOP1
;CLEAR BIT 0
;CLEAR "EXPECTED"
;READ CONTROL STATUS REGISTER
;BR IF ZERO
;DATA ERROR BIT0 NOT CLEARED
;SW09 UP?
;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

012704	012737	000005	001226	TST5:	MOV	#5,TSTNO	
012712	012737	013002	001216		MOV	#TST6,NEXT	
012720	012737	012730	001220		MOV	#1\$,LOCK	
012726	104412				MSTCLR		;MASTER CLEAR DMC11
012730	013701	001404		1\$:	MOV	DMCSR,R1	;PUT REGISTER ADDRESS IN R1
012734	012705	000002			MOV	#BIT1,R5	;PUT DATA IN "EXPECTED"
012740	010511				MOV	R5,(R1)	;WRITE BIT 1
012742	011104				MOV	(R1),R4	;READ CONTROL STATUS REGISTER
012744	020504				CMP	R5,R4	;IS DATA CORRECT
012746	001401				BEQ	2\$	;BR IF YES
012750	104002				EMT	2	;DATA ERROR
012752	104401			2\$:	SCOP1		;SW09 UP?
012754	012737	012762	001220		MOV	#3\$,LOCK	;NEW SCOP1
012762	042711	000002		3\$:	BIC	#BIT1,(R1)	;CLEAR BIT 1
012766	005005				CLR	R5	;CLEAR "EXPECTED"
012770	011104				MOV	(R1),R4	;READ CONTROL STATUS REGISTER
012772	001402				BEQ	4\$	;BR IF ZERO
012774	104002				EMT	2	;DATA ERROR BIT1 NOT CLEARED
012776	104401				SCOP1		;SW09 UP?
013000	104400			4\$:	SCOPE		;SCOPE THIS TEST

***** TEST 6 *****
;CONTROL STATUS REGISTER WRITE/READ TEST
;SET BIT2, VERIFY BIT2 WAS SET
;CLEAR BIT2, VERIFY BIT2 WAS CLEARED

; TEST 6

013002	012737	000006	001226	TST6:	MOV	#6,TSTNO	
013010	012737	013100	001216		MOV	#TST7,NEXT	
013016	012737	013026	001220		MOV	#1\$,LOCK	
013024	104412				MSTCLR		;MASTER CLEAR DMC11
013026	013701	001404		1\$:	MOV	DMCSR,R1	;PUT REGISTER ADDRESS IN R1
013032	012705	000004			MOV	#BIT2,R5	;PUT DATA IN "EXPECTED"
013036	010511				MOV	R5,(R1)	;WRITE BIT 2
013040	011104				MOV	(R1),R4	;READ CONTROL STATUS REGISTER
013042	020504				CMP	R5,R4	;IS DATA CORRECT
013044	001401				BEQ	2\$	;BR IF YES
013046	104002				EMT	2	;DATA ERROR
013050	104401			2\$:	SCOP1		;SW09 UP?
013052	012737	013060	001220		MOV	#3\$,LOCK	;NEW SCOP1
013060	042711	000004		3\$:	BIC	#BIT2,(R1)	;CLEAR BIT 2
013064	005005				CLR	R5	;CLEAR "EXPECTED"
013066	011104				MOV	(R1),R4	;READ CONTROL STATUS REGISTER
013070	001402				BEQ	4\$	;BR IF ZERO
013072	104002				EMT	2	;DATA ERROR BIT2 NOT CLEARED
013074	104401				SCOP1		;SW09 UP?

013076 104400

4\$:

SCOPE

;SCOPE THIS TEST

***** TEST 7 *****
;CONTROL STATUS REGISTER WRITE/READ TEST
;SET BITS, VERIFY BITS WAS SET
;CLEAR BITS, VERIFY BITS WAS CLEARED

; TEST 7

013100	012737	000007	001226	TST7:	MOV	#7,TSTNO	
013106	012737	013176	001216		MOV	#TST10,NEXT	
013114	012737	013124	001220		MOV	#1\$,LOCK	
013122	104412				MSTCLR		;MASTER CLEAR DMC11
013124	013701	001404		1\$:	MOV	DMCSR,R1	;PUT REGISTER ADDRESS IN R1
013130	012705	000040			MOV	#BIT5,R5	;PUT DATA IN "EXPECTED"
013134	010511				MOV	R5,(R1)	;WRITE BIT 5
013136	011104				MOV	(R1),R4	;READ CONTROL STATUS REGISTER
013140	020504				CMP	R5,R4	;IS DATA CORRECT
013142	001401				BEQ	2\$	;BR IF YES
013144	104002				EMT	2	;DATA ERROR
013146	104401			2\$:	SCOP1		;SW09 UP?
013150	012737	013156	001220		MOV	#3\$,LOCK	;NEW SCOP1
013156	042711	000040		3\$:	BIC	#BIT5,(R1)	;CLEAR BIT 5
013162	005005				CLR	R5	;CLEAR "EXPECTED"
013164	011104				MOV	(R1),R4	;READ CONTROL STATUS REGISTER
013166	001402				BEQ	4\$	;BR IF ZERO
013170	104002				EMT	2	;DATA ERROR BITS NOT CLEARED
013172	104401				SCOP1		;SW09 UP?
013174	104400			4\$:	SCOPE		;SCOPE THIS TEST

***** TEST 10 *****
;CONTROL STATUS REGISTER WRITE/READ TEST
;SET BIT6, VERIFY BIT6 WAS SET
;CLEAR BIT6, VERIFY BIT6 WAS CLEARED

; TEST 10

013176	012737	000010	001226	TST10:	MOV	#10,TSTNO	
013204	012737	013274	001216		MOV	#TST11,NEXT	
013212	012737	013222	001220		MOV	#1\$,LOCK	
013220	104412				MSTCLR		;MASTER CLEAR DMC11
013222	013701	001404		1\$:	MOV	DMCSR,R1	;PUT REGISTER ADDRESS IN R1
013226	012705	000100			MOV	#BIT6,R5	;PUT DATA IN "EXPECTED"
013232	010511				MOV	R5,(R1)	;WRITE BIT 6
013234	011104				MOV	(R1),R4	;READ CONTROL STATUS REGISTER
013236	020504				CMP	R5,R4	;IS DATA CORRECT
013240	001401				BEQ	2\$	;BR IF YES
013242	104002				EMT	2	;DATA ERROR
013244	104401			2\$:	SCOP1		;SW09 UP?
013246	012737	013254	001220		MOV	#3\$,LOCK	;NEW SCOP1
013254	042711	000100		3\$:	BIC	#BIT6,(R1)	;CLEAR BIT 6
013260	005005				CLR	R5	;CLEAR "EXPECTED"
013262	011104				MOV	(R1),R4	;READ CONTROL STATUS REGISTER

013264 001402
013266 104002
013270 104401
013272 104400

4\$:

BEQ 4\$
EMT 2
SCOP1
SCOPE

;BR IF ZERO
;DATA ERROR BIT6 NOT CLEARED
;SW09 UP?
;SCOPE THIS TEST

***** TEST 11 *****
;*CONTROL STATUS REGISTER WRITE/READ TEST
;*SET BIT7, VERIFY BIT7 WAS SET
;*CLEAR BIT7, VERIFY BIT7 WAS CLEARED
;*****

; TEST 11

013274 012737 000011 001226 TST11:
013302 012737 013372 001216
013310 012737 013320 001220
013316 104412
013320 013701 001404 1\$:
013324 012705 000200
013330 010511
013332 011104
013334 020504
013336 001401
013340 104002
013342 104401
013344 012737 013352 001220 2\$:
013352 042711 000200 3\$:
013356 005005
013360 011104
013362 001402
013364 104002
013366 104401
013370 104400 4\$:

MOV #11,TSTNO
MOV #TST12,NEXT
MOV #1\$,LOCK
MSTCLR
MOV DMCSR,R1
MOV #BIT7,R5
MOV R5,(R1)
MOV (R1),R4
CMP R5,R4
BEQ 2\$
EMT 2
SCOP1
MOV #3\$,LOCK
BIC #BIT7,(R1)
CLR R5
MOV (R1),R4
BEQ 4\$
EMT 2
SCOP1
SCOPE

;MASTER CLEAR DMC11
;PUT REGISTER ADDRESS IN R1
;PUT DATA IN "EXPECTED"
;WRITE BIT 7
;READ CONTROL STATUS REGISTER
;IS DATA CORRECT
;BR IF YES
;DATA ERROR
;SW09 UP?
;NEW SCOP1
;CLEAR BIT 7
;CLEAR "EXPECTED"
;READ CONTROL STATUS REGISTER
;BR IF ZERO
;DATA ERROR BIT7 NOT CLEARED
;SW09 UP?
;SCOPE THIS TEST

***** TEST 12 *****
;*CONTROL STATUS REGISTER WRITE/READ TEST
;*SET BIT9, VERIFY BIT9 WAS SET
;*CLEAR BIT9, VERIFY BIT9 WAS CLEARED
;*****

; TEST 12

013372 012737 000012 001226 TST12:
013400 012737 013470 001216
013406 012737 013416 001220
013414 104412
013416 013701 001404 1\$:
013422 012705 001000
013426 010511
013430 011104
013432 020504
013434 001401
013436 104002
013440 104401
013442 012737 013450 001220 2\$:

MOV #12,TSTNO
MOV #TST13,NEXT
MOV #1\$,LOCK
MSTCLR
MOV DMCSR,R1
MOV #BIT9,R5
MOV R5,(R1)
MOV (R1),R4
CMP R5,R4
BEQ 2\$
EMT 2
SCOP1
MOV #3\$,LOCK

;MASTER CLEAR DMC11
;PUT REGISTER ADDRESS IN R1
;PUT DATA IN "EXPECTED"
;WRITE BIT 9
;READ CONTROL STATUS REGISTER
;IS DATA CORRECT
;BR IF YES
;DATA ERROR
;SW09 UP?
;NEW SCOP1

013450	042711	001000	3#:	BIC	#BIT9,(R1)	;CLEAR BIT 9
013454	005005			CLR	R5	;CLEAR "EXPECTED"
013456	011104			MOV	(R1),R4	;READ CONTROL STATUS REGISTER
013460	001402			BEQ	4#	;BR IF ZERO
013462	104002			EMT	2	;DATA ERROR BIT9 NOT CLEARED
013464	104401			SCOP1		;SW09 UP?
013466	104400		4#:	SCOPE		;SCOPE THIS TEST

***** TEST 13 *****
;CONTROL STATUS REGISTER WRITE/READ TEST
;SET BIT11, VERIFY BIT11 WAS SET
;CLEAR BIT11, VERIFY BIT11 WAS CLEARED
;*****

; TEST 13

013470	012737	000013	001226	TST13:	MOV	#13,TSTNO	
013476	012737	013566	001216		MOV	#TST14,NEXT	
013504	012737	013514	001220		MOV	#1#,LOCK	
013512	104412				MSTCLR		;MASTER CLEAR DMC11
013514	013701	001404		1#:	MOV	DMCSR,R1	;PUT REGISTER ADDRESS IN R1
013520	012705	004000			MOV	#BIT11,R5	;PUT DATA IN "EXPECTED"
013524	010511				MOV	R5,(R1)	;WRITE BIT 11
013526	011104				MOV	(R1),R4	;READ CONTROL STATUS REGISTER
013530	020504				CMP	R5,R4	;IS DATA CORRECT
013532	001401				BEQ	2#	;BR IF YES
013534	104002				EMT	2	;DATA ERROR
013536	104401			2#:	SCOP1		;SW09 UP?
013540	012737	013546	001220		MOV	#3#,LOCK	;NEW SCOP1
013546	042711	004000		3#:	BIC	#BIT11,(R1)	;CLEAR BIT 11
013552	005005				CLR	R5	;CLEAR "EXPECTED"
013554	011104				MOV	(R1),R4	;READ CONTROL STATUS REGISTER
013556	001402				BEQ	4#	;BR IF ZERO
013560	104002				EMT	2	;DATA ERROR BIT11 NOT CLEARED
013562	104401				SCOP1		;SW09 UP?
013564	104400			4#:	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

013566	012737	000014	001226	TST14:	MOV	#14,TSTNO	
013574	012737	013664	001216		MOV	#TST15,NEXT	
013602	012737	013612	001220		MOV	#1#,LOCK	
013610	104412				MSTCLR		;MASTER CLEAR DMC11
013612	013701	001404		1#:	MOV	DMCSR,R1	;PUT REGISTER ADDRESS IN R1
013616	012705	010000			MOV	#BIT12,R5	;PUT DATA IN "EXPECTED"
013622	010511				MOV	R5,(R1)	;WRITE BIT 12
013624	011104				MOV	(R1),R4	;READ CONTROL STATUS REGISTER
013626	020504				CMP	R5,R4	;IS DATA CORRECT
013630	001401				BEQ	2#	;BR IF YES

```

013632 104002
013634 104401
013636 012737 013644 001220 2#: EMT 2 ;DATA ERROR
013644 042711 010000 3#: MOV #3$,LOCK ;SW09 UP?
013650 005005 BIC #BIT12,(R1) ;NEW SCOP1
013652 011104 CLR R5 ;CLEAR BIT 12
013654 001402 MOV (R1),R4 ;CLEAR "EXPECTED"
013656 104002 BEQ 4$ ;READ CONTROL STATUS REGISTER
013660 104401 EMT 2 ;BR IF ZERO
013662 104400 4#: SCOP1 ;DATA ERROR BIT12 NOT CLEARED
SCOPE ;SW09 UP?
;SCOPE THIS TEST

```

15

```

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

```

; TEST 15

```

013664 012737 000015 001226 TST15: MOV #15,TSTNO
013672 012737 013762 001216 MOV #TST16,NEXT
013700 012737 013710 001220 MOV #1$,LOCK
013706 104412 MSTCLR ;MASTER CLEAR DMC11
013710 013701 001410 1#: MOV DMCTL,R1 ;PUT REGISTER ADDRESS IN R1
013714 012705 000001 MOV #BIT0,R5 ;PUT DATA IN "EXPECTED"
013720 010511 MOV R5,(R1) ;WRITE BIT 0
013722 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
013724 020504 CMP R5,R4 ;IS DATA CORRECT
013726 001401 BEQ 2$ ;BR IF YES
013730 104002 EMT 2 ;DATA ERROR
013732 104401 2#: SCOP1 ;SW09 UP?
013734 012737 013742 001220 MOV #3$,LOCK ;NEW SCOP1
013742 042711 000001 3#: BIC #BIT0,(R1) ;CLEAR BIT 0
013746 005005 CLR R5 ;CLEAR "EXPECTED"
013750 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
013752 001402 BEQ 4$ ;BR IF ZERO
013754 104002 EMT 2 ;DATA ERROR BIT0 NOT CLEARED
013756 104401 4#: SCOP1 ;SW09 UP?
104400 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

```

013762 012737 000016 001226 TST16: MOV #16,TSTNO
013770 012737 014060 001216 MOV #TST17,NEXT
013776 012737 014006 001220 MOV #1$,LOCK
014004 104412 MSTCLR ;MASTER CLEAR DMC11
014006 013701 001410 1#: MOV DMCTL,R1 ;PUT REGISTER ADDRESS IN R1
014012 012705 000002 MOV #BIT1,R5 ;PUT DATA IN "EXPECTED"
014016 010511 MOV R5,(R1) ;WRITE BIT 1

```


014020	011104				MOV	(R1),R4	;READ CONTROL OUT REGISTER
014022	020504				CMP	R5,R4	;IS DATA CORRECT
014024	001401				BEQ	2\$	;BR IF YES
014026	104002				EMT	2	;DATA ERROR
014030	104401			2\$:	SCOP1		;SW09 UP?
014032	012737	014040	001220		MOV	#3\$,LOCK	;NEW SCOP1
014040	042711	000002		3\$:	BIC	#BIT1,(R1)	;CLEAR BIT 1
014044	005005				CLR	R5	;CLEAR "EXPECTED"
014046	011104				MOV	(R1),R4	;READ CONTROL OUT REGISTER
014050	001402				BEQ	4\$	;BR IF ZERO
014052	104002				EMT	2	;DATA ERROR BIT1 NOT CLEARED
014054	104401				SCOP1		;SW09 UP?
014056	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

014060	012737	000017	001226	TST17:	MOV	#17,TSTNO	
014066	012737	014156	001216		MOV	#TST20,NEXT	
014074	012737	014104	001220		MOV	#1\$,LOCK	
014102	104412				MSTCLR		;MASTER CLEAR DMC11
014104	013701	001410		1\$:	MOV	DMCTL,R1	;PUT REGISTER ADDRESS IN R1
014110	012705	000004			MOV	#BIT2,R5	;PUT DATA IN "EXPECTED"
014114	010511				MOV	R5,(R1)	;WRITE BIT 2
014116	011104				MOV	(R1),R4	;READ CONTROL OUT REGISTER
014120	020504				CMP	R5,R4	;IS DATA CORRECT
014122	001401				BEQ	2\$	;BR IF YES
014124	104002				EMT	2	;DATA ERROR
014126	104401			2\$:	SCOP1		;SW09 UP?
014130	012737	014136	001220		MOV	#3\$,LOCK	;NEW SCOP1
014136	042711	000004		3\$:	BIC	#BIT2,(R1)	;CLEAR BIT 2
014142	005005				CLR	R5	;CLEAR "EXPECTED"
014144	011104				MOV	(R1),R4	;READ CONTROL OUT REGISTER
014146	001402				BEQ	4\$	;BR IF ZERO
014150	104002				EMT	2	;DATA ERROR BIT2 NOT CLEARED
014152	104401				SCOP1		;SW09 UP?
014154	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

014156	012737	000020	001226	TST20:	MOV	#20,TSTNO	
014164	012737	014254	001216		MOV	#TST21,NEXT	
014172	012737	014202	001220		MOV	#1\$,LOCK	
014200	104412				MSTCLR		;MASTER CLEAR DMC11

014202	013701	001410	1\$:	MOV	DMCTL,R1	;PUT REGISTER ADDRESS IN R1
014206	012705	000100		MOV	#BIT6,R5	;PUT DATA IN "EXPECTED"
014212	010511			MOV	R5,(R1)	;WRITE BIT 6
014214	011104			MOV	(R1),R4	;READ CONTROL OUT REGISTER
014216	020504			CMP	R5,R4	;IS DATA CORRECT
014220	001401			BEQ	2\$	;BR IF YES
014222	104002			EMT	2	;DATA ERROR
014224	104401		2\$:	SCOP1		;SW09 UP?
014226	012737	014234	001220	MOV	#3\$,LOCK	;NEW SCOP1
014234	042711	000100	3\$:	BIC	#BIT6,(R1)	;CLEAR BIT 6
014240	005005			CLR	R5	;CLEAR "EXPECTED"
014242	011104			MOV	(R1),R4	;READ CONTROL OUT REGISTER
014244	001402			BEQ	4\$	;BR IF ZERO
014246	104002			EMT	2	;DATA ERROR BIT6 NOT CLEARED
014250	104401			SCOP1		;SW09 UP?
014252	104400		4\$:	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

014254	012737	000021	001226	TST21:	MOV	#21,TSTNO	
014262	012737	014352	001216		MOV	#TST22,NEXT	
014270	012737	014300	001220		MOV	#1\$,LOCK	
014276	104412				MSTCLR		;MASTER CLEAR DMC11
014300	013701	001410		1\$:	MOV	DMCTL,R1	;PUT REGISTER ADDRESS IN R1
014304	012705	000200			MOV	#BIT7,R5	;PUT DATA IN "EXPECTED"
014310	010511				MOV	R5,(R1)	;WRITE BIT 7
014312	011104				MOV	(R1),R4	;READ CONTROL OUT REGISTER
014314	020504				CMP	R5,R4	;IS DATA CORRECT
014316	001401				BEQ	2\$	;BR IF YES
014320	104002				EMT	2	;DATA ERROR
014322	104401			2\$:	SCOP1		;SW09 UP?
014324	012737	014332	001220		MOV	#3\$,LOCK	;NEW SCOP1
014332	042711	000200		3\$:	BIC	#BIT7,(R1)	;CLEAR BIT 7
014336	005005				CLR	R5	;CLEAR "EXPECTED"
014340	011104				MOV	(R1),R4	;READ CONTROL OUT REGISTER
014342	001402				BEQ	4\$	;BR IF ZERO
014344	104002				EMT	2	;DATA ERROR BIT7 NOT CLEARED
014346	104401				SCOP1		;SW09 UP?
014350	104400			4\$:	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

014352	012737	000022	001226	TST22:	MOV	#22,TSTNO
--------	--------	--------	--------	--------	-----	-----------

014360	012737	014450	001216		MOV	#TST23,NEXT	
014366	012737	014376	001220		MOV	#1\$,LOCK	
014374	104412				MSTCLR		;MASTER CLEAR DMC11
014376	013701	001410		1\$:	MOV	DMCTL,R1	;PUT REGISTER ADDRESS IN R1
014402	012705	010000			MOV	#BIT12,R5	;PUT DATA IN "EXPECTED"
014406	010511				MOV	R5,(R1)	;WRITE BIT 12
014410	011104				MOV	(R1),R4	;READ CONTROL OUT REGISTER
014412	020504				CMP	R5,R4	;IS DATA CORRECT
014414	001401				BEQ	2\$	;BR IF YES
014416	104002				EMT	2	;DATA ERROR
014420	104401			2\$:	SCOP1		;SW09 UP?
014422	012737	014430	001220		MOV	#3\$,LOCK	;NEW SCOP1
014430	042711	010000		3\$:	BIC	#BIT12,(R1)	;CLEAR BIT 12
014434	005005				CLR	R5	;CLEAR "EXPECTED"
014436	011104				MOV	(R1),R4	;READ CONTROL OUT REGISTER
014440	001402				BEQ	4\$	;BR IF ZERO
014442	104002				EMT	2	;DATA ERROR BIT12 NOT CLEARED
014444	104401				SCOP1		;SW09 UP?
014446	104400			4\$:	SCOPE		;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

014450	012737	000023	001226	TST23:	MOV	#23,TSTNO	
014456	012737	014546	001216		MOV	#TST24,NEXT	
014464	012737	014474	001220		MOV	#1\$,LOCK	
014472	104412				MSTCLR		;MASTER CLEAR DMC11
014474	013701	001410		1\$:	MOV	DMCTL,R1	;PUT REGISTER ADDRESS IN R1
014500	012705	020000			MOV	#BIT13,R5	;PUT DATA IN "EXPECTED"
014504	010511				MOV	R5,(R1)	;WRITE BIT 13
014506	011104				MOV	(R1),R4	;READ CONTROL OUT REGISTER
014510	020504				CMP	R5,R4	;IS DATA CORRECT
014512	001401				BEQ	2\$	;BR IF YES
014514	104002				EMT	2	;DATA ERROR
014516	104401			2\$:	SCOP1		;SW09 UP?
014520	012737	014526	001220		MOV	#3\$,LOCK	;NEW SCOP1
014526	042711	020000		3\$:	BIC	#BIT13,(R1)	;CLEAR BIT 13
014532	005005				CLR	R5	;CLEAR "EXPECTED"
014534	011104				MOV	(R1),R4	;READ CONTROL OUT REGISTER
014536	001402				BEQ	4\$	;BR IF ZERO
014540	104002				EMT	2	;DATA ERROR BIT13 NOT CLEARED
014542	104401				SCOP1		;SW09 UP?
014544	104400			4\$:	SCOPE		;SCOPE THIS TEST

16

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

```

; TEST 24
-----
014546 012737 000024 001226 TST24: MOV #24,TSTNO
014554 012737 014672 001216 MOV #TST25,NEXT
014562 012737 014602 001220 MOV #64$,LOCK
014570 104412 MSTCLR ;MASTER CLEAR DMC11
014572 013701 001412 MOV DMP04,R1 ;PUT REGISTER ADDRESS IN R1
014576 012700 000001 MOV #1,R0 ;START WITH BIT0
014602 64$: MOV R0,R5 ;PUT "EXPECTED" IN R5
014604 010005 MOV R5,(R1) ;WRITE PORT4 REGISTER
014606 011104 MOV (R1),R4 ;READ PORT4 REGISTER
014610 020504 CMP R5,R4 ;COMPARE EXPECTED AND FOUND
014612 001401 BEQ 65$ ;BR IF OK
014614 104002 EMT 2 ;WRITE/READ ERROR
014616 104401 65$: SCOP1 ;LOOP TO 64$ IF SW09=1
014620 000241 CLC ;CLEAR CARRY
014622 006100 ROL R0 ;SHIFT TO NEXT BIT
014624 001366 BNE 64$ ;BR IF NOT DONE YET?
014626 012737 014640 001220 MOV #66$,LOCK ;NEW SCOP1
014634 012700 000001 MOV #1,R0 ;START WITH BIT0
014640 66$: COM R0 ;CHANGE TO A FLOATING ZERO
014642 010005 MOV R0,R5 ;PUT "EXPECTED" IN R5
014644 010511 MOV R5,(R1) ;WRITE PORT4 REGISTER
014646 011104 MOV (R1),R4 ;READ PORT4 REGISTER
014650 020504 CMP R5,R4 ;COMPARE EXPECTED AND FOUND
014652 001401 BEQ 67$ ;BR IF OK
014654 104002 EMT 2 ;WRITE/READ ERROR
014656 104401 67$: SCOP1 ;LOOP TO 66$ IF SW09=1
014660 005100 COM R0 ;CHANGE BACK TO A FLOATING ONE
014662 000241 CLC ;CLEAR CARRY
014664 006100 ROL R0 ;SHIFT TO NEXT BIT
014666 001366 BNE 66$ ;BR IF NOT DONE YET?
014670 104400 SCOPE ;SCOPE THIS TEST

```

17

```

***** TEST 25 *****
;PORT6 REGISTER WRITE/READ TEST
;FLOAT A ONE THROUGH PORT6 REGISTER
;FLOAT A ZERO THROUGH PORT6 REGISTER
;*****

```

```

; TEST 25
-----
014672 012737 000025 001226 TST25: MOV #25,TSTNO
014700 012737 015016 001216 MOV #TST26,NEXT
014706 012737 014726 001220 MOV #64$,LOCK
014714 104412 MSTCLR ;MASTER CLEAR DMC11
014716 013701 001414 MOV DMP06,R1 ;PUT REGISTER ADDRESS IN R1
014722 012700 000001 MOV #1,R0 ;START WITH BIT0
014726 64$: MOV R0,R5 ;PUT "EXPECTED" IN R5
014728 010005 MOV R5,(R1) ;WRITE PORT6 REGISTER
014730 010511 MOV (R1),R4 ;READ PORT6 REGISTER
014732 011104 CMP R5,R4 ;COMPARE EXPECTED AND FOUND
014734 020504 BEQ 65$ ;BR IF OK
014736 001401

```


Address	Hex	Label	Instruction	Register	Comment
014740	104002		EMT	2	;WRITE/READ ERROR
014742	104401	65\$:	SCOP1		;LOOP TO 64\$ IF SW09=1
014744	000241		CLC		;CLEAR CARRY
014746	006100		ROL	R0	;SHIFT TO NEXT BIT
014750	001366		BNE	64\$	;BR IF NOT DONE YET?
014752	012737	014764 001220	MOV	#66\$,LOCK	;NEW SCOP1
014760	012700	000001	MOV	#1,R0	;START WITH BIT0
014764		66\$:			
014764	005100		COM	R0	;CHANGE TO A FLOATING ZERO
014766	010005		MOV	R0,R5	;PUT "EXPECTED" IN R5
014770	010511		MOV	R5,(R1)	;WRITE PORT6 REGISTER
014772	011104		MOV	(R1),R4	;READ PORT6 REGISTER
014774	020504		CMP	R5,R4	;COMPARE EXPECTED AND FOUND
014776	001401		BEQ	67\$	;BR IF OK
015000	104002		EMT	2	;WRITE/READ ERROR
015002	104401	67\$:	SCOP1		;LOOP TO 66\$ IF SW09=1
015004	005100		COM	R0	;CHANGE BACK TO A FLOATING ONE
015006	000241		CLC		;CLEAR CARRY
015010	006100		ROL	R0	;SHIFT TO NEXT BIT
015012	001364		BNE	66\$	;BR IF NOT DONE YET?
015014	104400		SCOPE		;SCOPE THIS TEST

```

***** TEST 26 *****
*UNIBUS REGISTER BYTE DUAL ADDRESSING TEST
*LOAD ALL REGISTERS WITH INCREMENTING PATTERN
*READ BACK ALL REGISTERS TO VERIFY CORRECT ADDRESSING
*: *****

```

: TEST 26

Address	Hex	Hex	Hex	Label	Instruction	Comment
015016	012737	000026	001226	TST26:	MOV	#26,TSTNO
015024	012737	015146	001216		MOV	#TST27,NEXT
015032	012737	015046	001220		MOV	#1\$,LOCK
015040	104412				MSTCLR	;R1 CONTAINS BASE DMC11 ADDRESS
015042	012700	000001			MOV	#1,R0
015046	105011			1\$:	CLRB	(R1)
015050	110005				MOVB	R0,R5
015052	110011				MOVB	R0,(R1)
015054	111104				MOVB	(R1),R4
015056	020504				CMP	R5,R4
015060	001401				BEQ	2\$
015062	104002				EMT	2
015064	104401			2\$:	SCOP1	;DATA ERROR
015066	105721				TSTB	(R1)+
015070	005200				INC	R0
015072	022700	000011			CMP	#11,R0
015076	001363				BNE	1\$
015100	013701	001404			MOV	DMCSR,R1
015104	012700	000001			MOV	#1,R0
015110	012737	015116	001220		MOV	#3\$,LOCK
015116	110005			3\$:	MOVB	R0,R5
015120	111104				MOVB	(R1),R4
015122	020504				CMP	R5,R4
015124	001401				BEQ	4\$
015126	104002				EMT	2
						;IS DATA CORRECT
						;BR IF YES
						;SW09=1?
						;NEXT REGISTER
						;INCREMENT DATA PATTERN
						;LAST REGISTER?
						;BR IF NO
						;BASE DMC11 ADDRESS TO R1
						;RESTART PATTERN AT 1
						;NEW SCOP1
						;PUT DATA IN "EXPECTED"
						;READ DMC REGISTER INTO "FOUND"
						;IS DATA CORRECT
						;BR IF YES
						;DUAL ADDRESSING ERROR

015130 104401
015132 105721
015134 005200
015136 022700 000011
015142 001365
015144 104400

4\$: SCOP1 ;SW09=1?
TSTB (R1)+ ;NEXT REGISTER
INC R0 ;INCREMENT PATTERN
CMP #11,R0 ;LAST REGISTER?
BNE 3\$;BR IF NO
SCOPE ;SCOPE THIS TEST

19

***** 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

015146 012737 000027 001226 TST27:
015154 012737 015306 001216
015162 012737 015200 001220

015170 104412
015172 012711 003000
015176 005005
015200 010561 000006 1\$:
015204 016104 000006
015210 020504
015212 001401
015214 104023
015216 104401 2\$:
015220 012737 015232 001220
015226 012705 177777
015232 010561 000006 3\$:
015236 016104 000006
015242 020504
015244 001401
015246 104023
015250 104401 4\$:
015252 012737 015262 001220
015260 005005
015262 000005 5\$:
015264 012711 003000
015270 016104 000006
015274 020504
015276 001401
015300 104023
015302 104401 6\$:
015304 104400

MOV #27,TSTNO
MOV #TST30,NEXT
MOV #1\$,LOCK

MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
MOV #BIT9:BIT10,(R1) ;MASTER CLEAR DMC11
CLR R5 ;SEL6 IS NOW THE IR
1\$: MOV R5,6(R1) ;PUT "EXPECTED" IN R5
MOV 6(R1),R4 ;CLEAR THE IR
CMP R5,R4 ;READ THE IR
BEQ 2\$;IS IT CLEARED?
EMT 23 ;BR IF YES
2\$: SCOP1 ;ERROR IR IS NOT CLEAR
MOV #3\$,LOCK ;LOOP TO 1\$ IF SW09=1
MOV #1,R5 ;NEW SCOP1
3\$: MOV R5,6(R1) ;PUT "EXPECTED" IN R5
MOV 6(R1),R4 ;WRITE ALL ONES TO THE IR
CMP R5,R4 ;READ THE IR
BEQ 4\$;IS IT ALL ONES?
EMT 23 ;BR IF YES
4\$: SCOP1 ;ERROR IR IS NOT = ALL ONES
MOV #5\$,LOCK ;LOOP TO 3\$ IF SW09=1
CLR R5 ;NEW SCOP1
5\$: RESET ;PUT "EXPECTED" IN R5
MOV #BIT9:BIT10,(R1) ;BUS RESET
MOV 6(R1),R4 ;SEL6 IS IR
CMP R5,R4 ;READ THE IR
BEQ 6\$;IS IT CLEARED?
EMT 23 ;BR IF YES
6\$: SCOP1 ;ERROR IR IS NOT CLEARED
SCOPE ;LOOP TO 5\$ IF SW09=1
;SCOPE THIS TEST

20

***** 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


```
015306 012737 000030 001226 TST30: MOV #30,TSTNO
015314 012737 015450 001216 MOV #TST31,NEXT
015322 012737 015340 001220 MOV #1$,LOCK

015330 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
015332 012711 003000 MOV #BIT9:BIT10,(R1) ;MASTER CLEAR DMC11
015336 005005 CLR R5 ;SEL6 IS NOW THE IR
015340 010561 000006 1$: MOV R5,6(R1) ;PUT "EXPECTED" IN R5
015344 016104 000006 MOV 6(R1),R4 ;CLEAR THE IR
015350 020504 CMP R5,R4 ;READ THE IR
015352 001401 BEQ 2$ ;IS IT CLEARED?
015354 104023 EMT 23 ;BR IF YES
015356 104401 2$: SCOP1 ;ERROR IR IS NOT CLEAR
015360 012737 015372 001220 MOV #3$,LOCK ;LOOP TO 1$ IF SW09=1
015366 012705 177777 MOV #-1,R5 ;NEW SCOP1
015372 010561 000006 3$: MOV R5,6(R1) ;PUT "EXPECTED" IN R5
015376 016104 000006 MOV 6(R1),R4 ;WRITE ALL ONES TO THE IR
015402 020504 CMP R5,R4 ;READ THE IR
015404 001401 BEQ 4$ ;IS IT ALL ONES?
015406 104023 EMT 23 ;BR IF YES
015410 104401 4$: SCOP1 ;ERROR IR IS NOT = ALL ONES
015412 012737 015422 001220 MOV #5$,LOCK ;LOOP TO 3$ IF SW09=1
015420 005005 CLR R5 ;NEW SCOP1
015422 052711 040000 5$: BIS #BIT14,(R1) ;PUT "EXPECTED" IN R5
015426 012711 003000 MOV #BIT9:BIT10,(R1) ;MASTER CLEAR
015432 016104 000006 MOV 6(R1),R4 ;SEL6 IS IR
015436 020504 CMP R5,R4 ;READ THE IR
015440 001401 BEQ 6$ ;IS IT CLEARED?
015442 104023 EMT 23 ;BR IF YES
015444 104401 6$: SCOP1 ;ERROR, IR IS NOT CLEARED
015446 104400 SCOPE ;LOOP TO 5$ IF SW09=1
;SCOPE THIS TEST
```

24

```
***** 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

```
015450 012737 000031 001226 TST31: MOV #31,TSTNO
015456 012737 015534 001216 MOV #TST32,NEXT

015464 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
015466 012761 000377 000004 MOV #377,4(R1) ;MASTER CLEAR DMC11
015474 012711 001000 MOV #BIT9,(R1) ;PORT4 HI-BYTE=0'S LO-BYTE=1'S
015500 012761 121105 000006 MOV #121105,6(R1) ;SET ROMI
015506 052711 001400 BIS #BIT8:BIT9,(R1) ;INSTRUCTION TO PORT6
015512 000240 NOP ;CLK INSTRUCTION, MOVE IBUS*4 TO IBUS*5
015514 012705 177777 MOV #-1,R5 ;
015520 016104 000004 MOV 4(R1),R4 ;PUT "EXPECTED" IN R5
015524 020504 CMP R5,R4 ;PUT "FOUND" INTO R4
015526 001401 BEQ 1$ ;IS DATA CORRECT
015530 104003 EMT 3 ;BR IF YES
;ERROR
```

015532 104400
25

14: SCOPE

;SCOPE THIS TEST

***** TEST 32 *****
;MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
;FLOAT A 1 THROUGH IBUS* REGISTER 0
;FLOAT A 0 THROUGH IBUS* REGISTER 0
;*****

; TEST 32

015534 012737 000032 001226 TST32:
015542 012737 015734 001216
015550 012737 015570 001220

MOV #32,TSTNO
MOV #TST33,NEXT
MOV #64\$,LOCK

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SAVE REGISTER ADDRESS FOR TYPEOUT
;START WITH BIT 0

015556 104412
015560 012702 000000
015564 012700 000001
015570

MSTCLR
MOV #0,R2
MOV #1,R0

;PUT PATTERN INTO PORT4
;CLEAR UNWANTED BITS
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOV DATA TO IBUS* REGISTER 0
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;READ FROM IBUS* REGISTER 0

015570 010061 000004
015574 042761 000030 000004
015602 104414
015604 121100
015606 104414
015610 121005
015612 010005
015614 042705 000030
015620 116104 000005
015624 120504
015626 001401
015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

64\$: MOV R0,4(R1)
BIC #30,4(R1)
ROMCLK
121100!0
ROMCLK
121005!<0*20>
MOV R0,R5
BIC #30,R5
MOVB 5(R1),R4
CMPB R5,R4
BEQ 65\$
EMT 4

;PUT EXPECTED IN R5
;CLEAR UNWANTED BITS
;PUT "FOUND" INTO R4
;DATA CORRECT?
;BR IF YES

015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

65\$: SCOP1
CLC
ROLB R0
BNE 64\$
MOV #67\$,LOCK
MOV #1,R0
COM R0

;ERROR
;SW09=1?
;CLEAR CARRY
;SHIFT BIT IN R0
;IF R0=0 THEN DONE
;NEW SCOP1
;START WITH BIT 0
;CHANGE TO FLOATING ZERO

015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

66\$: MOV R0,4(R1)
BIC #30,4(R1)
ROMCLK
121100!0
ROMCLK
121005!<0*20>
MOV R0,R5
BIC #30,R5
MOVB 5(R1),R4
CMPB R5,R4
BEQ 68\$
EMT 4

;PUT PATTERN INTO PORT4
;CLEAR UNWANTED BITS
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOV DATA TO IBUS* REGISTER 0
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;READ FROM IBUS* REGISTER 0

015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

67\$: MOV R0,4(R1)
BIC #30,4(R1)
ROMCLK
121100!0
ROMCLK
121005!<0*20>
MOV R0,R5
BIC #30,R5
MOVB 5(R1),R4
CMPB R5,R4
BEQ 68\$
EMT 4

;PUT EXPECTED IN R5
;CLEAR UNWANTED BITS
;PUT "FOUND" INTO R4
;DATA CORRECT?
;BR IF YES

015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

68\$: SCOP1
COM R0
CLC
ROLB R0
BNE 69\$
SCOPE

;ERROR
;SW09=1?
;CHANGE TO FLOATING 1
;CLEAR CARRY
;SHIFT BIT IN R0
;IF R0=0 THEN DONE
;SCOPE THIS TEST

015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

69\$: MOV R0,4(R1)
BIC #30,4(R1)
ROMCLK
121100!0
ROMCLK
121005!<0*20>
MOV R0,R5
BIC #30,R5
MOVB 5(R1),R4
CMPB R5,R4
BEQ 68\$
EMT 4

015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

70\$: MOV R0,4(R1)
BIC #30,4(R1)
ROMCLK
121100!0
ROMCLK
121005!<0*20>
MOV R0,R5
BIC #30,R5
MOVB 5(R1),R4
CMPB R0,R5
BEQ 68\$
EMT 4

015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

71\$: MOV R0,4(R1)
BIC #30,4(R1)
ROMCLK
121100!0
ROMCLK
121005!<0*20>
MOV R0,R5
BIC #30,R5
MOVB 5(R1),R4
CMPB R0,R5
BEQ 68\$
EMT 4

015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

72\$: MOV R0,4(R1)
BIC #30,4(R1)
ROMCLK
121100!0
ROMCLK
121005!<0*20>
MOV R0,R5
BIC #30,R5
MOVB 5(R1),R4
CMPB R0,R5
BEQ 68\$
EMT 4

015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

73\$: MOV R0,4(R1)
BIC #30,4(R1)
ROMCLK
121100!0
ROMCLK
121005!<0*20>
MOV R0,R5
BIC #30,R5
MOVB 5(R1),R4
CMPB R0,R5
BEQ 68\$
EMT 4

015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

74\$: MOV R0,4(R1)
BIC #30,4(R1)
ROMCLK
121100!0
ROMCLK
121005!<0*20>
MOV R0,R5
BIC #30,R5
MOVB 5(R1),R4
CMPB R0,R5
BEQ 68\$
EMT 4

015630 104004
015632 104401
015634 000241
015636 106100
015640 001353
015642 012737 015656 001220
015650 012700 000001
015654 005100
015656

75\$: MOV R0,4(R1)
BIC #30,4(R1)
ROMCLK
121100!0
ROMCLK
121005!<0*20>
MOV R0,R5
BIC #30,R5
MOVB 5(R1),R4
CMPB R0,R5
BEQ 68\$
EMT 4

26

```
***** 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

015734	012737	000033	001226	TST33:	MOV	#33,TSTNO	
015742	012737	016134	001216		MOV	#TST34,NEXT	
015750	012737	015770	001220		MOV	#64\$,LOCK	
015756	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
015760	012702	000002			MOV	#2,R2	;MASTER CLEAR DMC11
015764	012700	000001			MOV	#1,R0	;SAVE REGISTER ADDRESS FOR TYPEOUT
015770				64\$:			;START WITH BIT 0
015770	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
015774	042761	000070	000004		BIC	#70,4(R1)	;CLEAR UNWANTED BITS
016002	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016004	121102				121100!2		;MOV DATA TO IBUS* REGISTER 2
016006	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016010	121045				121005!<2*20>		;READ FROM IBUS* REGISTER 2
016012	010005				MOV	R0,R5	;PUT EXPECTED IN R5
016014	042705	000070			BIC	#70,R5	;CLEAR UNWANTED BITS
016020	116104	000005			MOVB	5(R1),R4	;PUT "FOUND" INTO R4
016024	120504				CMPB	R5,R4	;DATA CORRECT?
016026	001401				BEQ	65\$	;BR IF YES
016030	104004				EMT	4	;ERROR
016032	104401			65\$:	SCOP1		;SW09=1?
016034	000241				CLC		;CLEAR CARRY
016036	106100				ROLB	R0	;SHIFT BIT IN R0
016040	001353				BNE	64\$	;IF R0=0 THEN DONE
016042	012737	016056	001220		MOV	#67\$,LOCK	;NEW SCOP1
016050	012700	000001			MOV	#1,R0	;START WITH BIT 0
016054	005100			69\$:	COM	R0	;CHANGE TO FLOATING ZERO
016056				67\$:			
016056	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
016062	042761	000070	000004		BIC	#70,4(R1)	;CLEAR UNWANTED BITS
016070	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016072	121102				121100!2		;MOV DATA TO IBUS* REGISTER 2
016074	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016076	121045				121005!<2*20>		;READ FROM IBUS* REGISTER 2
016100	010005				MOV	R0,R5	;PUT EXPECTED IN R5
016102	042705	000070			BIC	#70,R5	;CLEAR UNWANTED BITS
016106	116104	000005			MOVB	5(R1),R4	;PUT "FOUND" INTO R4
016112	120504				CMPB	R5,R4	;DATA CORRECT?
016114	001401				BEQ	68\$	;BR IF YES
016116	104004				EMT	4	;ERROR
016120	104401			68\$:	SCOP1		;SW09=1?
016122	005100				COM	R0	;CHANGE TO FLOATING 1
016124	000241				CLC		;CLEAR CARRY
016126	106100				ROLB	R0	;SHIFT BIT IN R0
016130	001351				BNE	69\$	;IF R0=0 THEN DONE
016132	104400				SCOPE		;SCOPE THIS TEST

27

***** TEST 34 *****
;MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
;FLOAT A 1 THROUGH IBUS* REGISTER 4
;FLOAT A 0 THROUGH IBUS* REGISTER 4

; TEST 34

016134	012737	000034	001226	TST34:	MOV	#34,TSTNO	
016142	012737	016310	001216		MOV	#TST35,NEXT	
016150	012737	016170	001220		MOV	#64\$,LOCK	
016156	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
016160	012702	000004			MOV	#4,R2	;MASTER CLEAR DMC11
016164	012700	000001			MOV	#1,R0	;SAVE REGISTER ADDRESS FOR TYPEOUT
016170							;START WITH BIT 0
016170	010061	000004		64\$:	MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
016174	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016176	121104				121100!4		;MOV DATA TO IBUS* REGISTER 4
016200	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016202	121105				121005!<4*20>		;READ FROM IBUS* REGISTER 4
016204	010005				MOV	R0,R5	;PUT EXPECTED IN R5
016206	116104	000005			MOVB	5(R1),R4	;PUT "FOUND" INTO R4
016212	120504				CMPB	R5,R4	;DATA CORRECT?
016214	001401				BEQ	65\$	;BR IF YES
016216	104004				EMT	4	;ERROR
016220	104401			65\$:	SCOP1		;SW09=1?
016222	000241				CLC		;CLEAR CARRY
016224	106100				ROLB	R0	;SHIFT BIT IN R0
016226	001360				BNE	64\$	;IF R0=0 THEN DONE
016230	012737	016244	001220		MOV	#67\$,LOCK	;NEW SCOP1
016236	012700	000001			MOV	#1,R0	;START WITH BIT 0
016242	005100			69\$:	COM	R0	;CHANGE TO FLOATING ZERO
016244				67\$:			
016244	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
016250	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016252	121104				121100!4		;MOV DATA TO IBUS* REGISTER 4
016254	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016256	121105				121005!<4*20>		;READ FROM IBUS* REGISTER 4
016260	010005				MOV	R0,R5	;PUT EXPECTED IN R5
016262	116104	000005			MOVB	5(R1),R4	;PUT "FOUND" INTO R4
016266	120504				CMPB	R5,R4	;DATA CORRECT?
016270	001401				BEQ	68\$	;BR IF YES
016272	104004				EMT	4	;ERROR
016274	104401			68\$:	SCOP1		;SW09=1?
016276	005100				COM	R0	;CHANGE TO FLOATING 1
016300	000241				CLC		;CLEAR CARRY
016302	106100				ROLB	R0	;SHIFT BIT IN R0
016304	001356				BNE	69\$	;IF R0=0 THEN DONE
016306	104400				SCOPE		;SCOPE THIS TEST

28

***** TEST 35 *****
;MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
;FLOAT A 1 THROUGH IBUS* REGISTER 5
;FLOAT A 0 THROUGH IBUS* REGISTER 5

; TEST 35

016310	012737	000035	001226	TST35:	MOV	#35,TSTNO	
016316	012737	016464	001216		MOV	#TST36,NEXT	
016324	012737	016344	001220		MOV	#64\$,LOCK	
016332	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
016334	012702	000005			MOV	#5,R2	;MASTER CLEAR DMC11
016340	012700	000001			MOV	#1,R0	;SAVE REGISTER ADDRESS FOR TYPEOUT
016344				64\$:			;START WITH BIT 0
016344	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
016350	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016352	121105				121100!5		;MOV DATA TO IBUS* REGISTER 5
016354	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016356	121125				121005!<5*20>		;READ FROM IBUS* REGISTER 5
016360	010005				MOV	R0,R5	;PUT EXPECTED IN R5
016362	116104	000005			MOVB	5(R1),R4	;PUT "FOUND" INTO R4
016366	120504				CMPB	R5,R4	;DATA CORRECT?
016370	001401				BEQ	65\$	;BR IF YES
016372	104004				EMT	4	;ERROR
016374	104401			65\$:	SCOP1		;SW09=1?
016376	000241				CLC		;CLEAR CARRY
016400	106100				ROLB	R0	;SHIFT BIT IN R0
016402	001360				BNE	64\$	;IF R0=0 THEN DONE
016404	012737	016420	001220		MOV	#67\$,LOCK	;NEW SCOP1
016412	012700	000001			MOV	#1,R0	;START WITH BIT 0
016416	005100			69\$:	COM	R0	;CHANGE TO FLOATING ZERO
016420				67\$:			
016420	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
016424	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016426	121105				121100!5		;MOV DATA TO IBUS* REGISTER 5
016430	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016432	121125				121005!<5*20>		;READ FROM IBUS* REGISTER 5
016434	010005				MOV	R0,R5	;PUT EXPECTED IN R5
016436	116104	000005			MOVB	5(R1),R4	;PUT "FOUND" INTO R4
016442	120504				CMPB	R5,R4	;DATA CORRECT?
016444	001401				BEQ	68\$	;BR IF YES
016446	104004				EMT	4	;ERROR
016450	104401			68\$:	SCOP1		;SW09=1?
016452	005100				COM	R0	;CHANGE TO FLOATING 1
016454	000241				CLC		;CLEAR CARRY
016456	106100				ROLB	R0	;SHIFT BIT IN R0
016460	001356				BNE	69\$	;IF R0=0 THEN DONE
016462	104400				SCOPE		;SCOPE THIS TEST

29

***** TEST 36 *****
;MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
;FLOAT A 1 THROUGH IBUS* REGISTER 10
;FLOAT A 0 THROUGH IBUS* REGISTER 10
;THE NPR RQ BIT (BIT 0) IS MASKED DURING THIS TEST

; TEST 36

016464	012737	000036	001226	TST36:	MOV	#36,TSTNO	
016472	012737	016664	001216		MOV	#TST37,NEXT	
016500	012737	016520	001220		MOV	#64\$,LOCK	
							;R1 CONTAINS BASE DMC11 ADDRESS
016506	104412				MSTCLR		;MASTER CLEAR DMC11
016510	012702	000010			MOV	#10,R2	;SAVE REGISTER ADDRESS FOR TYPEOUT
016514	012700	000001			MOV	#1,R0	;START WITH BIT 0
016520				64\$:			
016520	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
016524	042761	000141	000004		BIC	#141,4(R1)	;CLEAR UNWANTED BITS
016532	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016534	121110				121100!10		;MOV DATA TO IBUS* REGISTER 10
016536	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016540	121205				121005!<10*20>		;READ FROM IBUS* REGISTER 10
016542	010005				MOV	R0,R5	;PUT EXPECTED IN R5
016544	042705	000141			BIC	#141,R5	;CLEAR UNWANTED BITS
016550	116104	000005			MOVB	5(R1),R4	;PUT "FOUND" INTO R4
016554	120504				CMPB	R5,R4	;DATA CORRECT?
016556	001401				BEQ	65\$	;BR IF YES
016560	104004				EMT	4	;ERROR
016562	104401			65\$:	SCOP1		;SW09=1?
016564	000241				CLC		;CLEAR CARRY
016566	106100				ROLB	R0	;SHIFT BIT IN R0
016570	001353				BNE	64\$	;IF R0=0 THEN DONE
016572	012737	016606	001220		MOV	#67\$,LOCK	;NEW SCOP1
016600	012700	000001			MOV	#1,R0	;START WITH BIT 0
016604	005100			69\$:	COM	R0	;CHANGE TO FLOATING ZERO
016606				67\$:			
016606	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
016612	042761	000141	000004		BIC	#141,4(R1)	;CLEAR UNWANTED BITS
016620	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016622	121110				121100!10		;MOV DATA TO IBUS* REGISTER 10
016624	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016626	121205				121005!<10*20>		;READ FROM IBUS* REGISTER 10
016630	010005				MOV	R0,R5	;PUT EXPECTED IN R5
016632	042705	000141			BIC	#141,R5	;CLEAR UNWANTED BITS
016636	116104	000005			MOVB	5(R1),R4	;PUT "FOUND" INTO R4
016642	120504				CMPB	R5,R4	;DATA CORRECT?
016644	001401				BEQ	68\$	;BR IF YES
016646	104004				EMT	4	;ERROR
016650	104401			68\$:	SCOP1		;SW09=1?
016652	005100				COM	R0	;CHANGE TO FLOATING 1
016654	000241				CLC		;CLEAR CARRY
016656	106100				ROLB	R0	;SHIFT BIT IN R0
016660	001351				BNE	69\$	;IF R0=0 THEN DONE
016662	104400				SCOPE		;SCOPE THIS TEST

30

```

***** TEST 37 *****
;MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
;FLOAT A 1 THROUGH IBUS* REGISTER 11
;FLOAT A 0 THROUGH IBUS* REGISTER 11
;THE BR RQ BIT, PGM CLOCK BIT, FORCE POWER FAIL BIT
;(BITS 7,4,1) ARE ALL MASKED DURING THIS TEST
;*****
; TEST 37

```


016664	012737	000037	001226	TST37:	MOV	#37,TSTNO	
016672	012737	017104	001216		MOV	#TST40,NEXT	
016700	012737	016720	001220		MOV	#64\$,LOCK	
;R1 CONTAINS BASE DMC11 ADDRESS							
016706	104412				MSTCLR		;MASTER CLEAR DMC11
016710	012702	000011			MOV	#11,R2	;SAVE REGISTER ADDRESS FOR TYPEOUT
016714	012700	000001			MOV	#1,R0	;START WITH BIT 0
016720				64\$:			
016720	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
016724	042761	000262	000004		BIC	#262,4(R1)	;CLEAR UNWANTED BITS
016732	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016734	121111				121100!11		;MOV DATA TO IBUS* REGISTER 11
016736	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
016740	121225				121005!<11*20>		;READ FROM IBUS* REGISTER 11
016742	010005				MOV	R0,R5	;PUT EXPECTED IN R5
016744	042705	000262			BIC	#262,R5	;CLEAR UNWANTED BITS
016750	052705	000020			BIS	#20,R5	;ADD THESE BITS
016754	116104	000005			MOVB	5(R1),R4	;PUT "FOUND" INTO R4
016760	052704	000020			BIS	#20,R4	;ADD THIS BIT
016764	120504				CMPIB	R5,R4	;DATA CORRECT?
016766	001401				BEQ	65\$	;BR IF YES
016770	104004				EMT	4	;ERROR
016772	104401			65\$:	SCOP1		;SW09-1?
016774	000241				CLC		;CLEAR CARRY
016776	106100				ROLB	R0	;SHIFT BIT IN R0
017000	001347				BNE	64\$	;IF R0=0 THEN DONE
017002	012737	017016	001220		MOV	#67\$,LOCK	;NEW SCOP1
017010	012700	000001			MOV	#1,R0	;START WITH BIT 0
017014	005100			69\$:	COM	R0	;CHANGE TO FLOATING ZERO
017016				67\$:			
017016	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
017022	042761	000262	000004		BIC	#262,4(R1)	;CLEAR UNWANTED BITS
017030	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017032	121111				121100!11		;MOV DATA TO IBUS* REGISTER 11
017034	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017036	121225				121005!<11*20>		;READ FROM IBUS* REGISTER 11
017040	010005				MOV	R0,R5	;PUT EXPECTED IN R5
017042	042705	000262			BIC	#262,R5	;CLEAR UNWANTED BITS
017046	052705	000020			BIS	#20,R5	;ADD THESE BITS
017052	116104	000005			MOVB	5(R1),R4	;PUT "FOUND" INTO R4
017056	052704	000020			BIS	#20,R4	;ADD THIS BIT
017062	120504				CMPIB	R5,R4	;DATA CORRECT?
017064	001401				BEQ	68\$	;BR IF YES
017066	104004				EMT	4	;ERROR
017070	104401			68\$:	SCOP1		;SW09-1?
017072	005100				COM	R0	;CHANGE TO FLOATING 1
017074	000241				CLC		;CLEAR CARRY
017076	106100				ROLB	R0	;SHIFT BIT IN R0
017100	001345				BNE	69\$	;IF R0=0 THEN DONE
017102	104400				SCOPE		;SCOPE THIS TEST

36

***** 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
;-----
017104 012737 000040 001226 TST40: MOV #40,TSTNO
017112 012737 017260 001216 MOV #TST41,NEXT
017120 012737 017140 001220 MOV #64$,LOCK

017126 104412 MSTCLR
017130 012702 000000 MOV #0,R2
017134 012700 000001 MOV #1,R0
017140 64: MOV R0,4(R1)
017144 104414 ROMCLK
017146 122100 122100!0
017150 104414 ROMCLK
017152 021005 21005!<0*20>
017154 010005 MOV R0,R5
017156 116104 000005 MOVB 5(R1),R4
017162 120504 CMPB R5,R4
017164 001401 BEQ 65$
017166 104005 EMT 5
017170 104401 65$: SCOP1
017172 000241 CLC
017174 106100 ROLB R0
017176 001360 BNE 64$
017200 012737 017214 001220 MOV #67$,LOCK
017206 012700 000001 MOV #1,R0
017212 005100 69$: COM R0
017214 67$: MOV R0,4(R1)
017220 104414 ROMCLK
017222 122100 122100!0
017224 104414 ROMCLK
017226 021005 21005!<0*20>
017230 010005 MOV R0,R5
017232 116104 000005 MOVB 5(R1),R4
017236 120504 CMPB R5,R4
017240 001401 BEQ 68$
017242 104005 EMT 5
017244 104401 68$: SCOP1
017246 005100 COM R0
017250 000241 CLC
017252 106100 ROLB R0
017254 001356 BNE 69$
017256 104400 SCOPE

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SAVE REGISTER ADDRESS FOR TYPEOUT
;START WITH BIT 0

;PUT PATTERN INTO PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOV DATA TO IBUS REGISTER 0
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;READ FROM IBUS REGISTER 0
;PUT EXPECTED IN R5
;PUT "FOUND" INTO R4
;DATA CORRECT?
;BR IF YES
;ERROR
;SW09=1?
;CLEAR CARRY
;SHIFT BIT IN R0
;IF R0=0 THEN DONE
;NEW SCOP1
;START WITH BIT 0
;CHANGE TO FLOATING ZERO

;PUT PATTERN INTO PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOV DATA TO IBUS REGISTER 0
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;READ FROM IBUS REGISTER 0
;PUT EXPECTED IN R5
;PUT "FOUND" INTO R4
;DATA CORRECT?
;BR IF YES
;ERROR
;SW09=1?
;CHANGE TO FLOATING 1
;CLEAR CARRY
;SHIFT BIT IN R0
;IF R0=0 THEN DONE
;SCOPE THIS TEST

;***** TEST 41 *****
;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
;*FLOAT A 1 THROUGH IBUS REGISTER 1
;*FLOAT A 0 THROUGH IBUS REGISTER 1
;*****

; TEST 41
;-----
017260 012737 000041 001226 TST41: MOV #41,TSTNO

```


017266	012737	017434	001216	MOV	#TST42,NEXT	
017274	012737	017314	001220	MOV	#64\$,LOCK	
017302	104412			MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
017304	012702	000001		MOV	#1,R2	;MASTER CLEAR DMC11
017310	012700	000001		MOV	#1,R0	;SAVE REGISTER ADDRESS FOR TYPEOUT
017314						;START WITH BIT 0
017314	010061	000004		MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
017320	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017322	122101			122100!1		;MOV DATA TO IBUS REGISTER 1
017324	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017326	021025			21005!<1*20>		;READ FROM IBUS REGISTER 1
017330	010005			MOV	R0,R5	;PUT EXPECTED IN R5
017332	116104	000005		MOVB	5(R1),R4	;PUT "FOUND" INTO R4
017336	120504			CMPB	R5,R4	;DATA CORRECT?
017340	001401			BEQ	65\$	;BR IF YES
017342	104005			EMT	5	;ERROR
017344	104401			SCOP1		;SW09=1?
017346	000241			CLC		;CLEAR CARRY
017350	106100			ROLB	R0	;SHIFT BIT IN R0
017352	001360			BNE	64\$	;IF R0=0 THEN DONE
017354	012737	017370	001220	MOV	#67\$,LOCK	;NEW SCOP1
017362	012700	000001		MOV	#1,R0	;START WITH BIT 0
017366	005100			COM	R0	;CHANGE TO FLOATING ZERO
017370						
017370	010061	000004		MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
017374	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017376	122101			122100!1		;MOV DATA TO IBUS REGISTER 1
017400	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017402	021025			21005!<1*20>		;READ FROM IBUS REGISTER 1
017404	010005			MOV	R0,R5	;PUT EXPECTED IN R5
017406	116104	000005		MOVB	5(R1),R4	;PUT "FOUND" INTO R4
017412	120504			CMPB	R5,R4	;DATA CORRECT?
017414	001401			BEQ	68\$	;BR IF YES
017416	104005			EMT	5	;ERROR
017420	104401			SCOP1		;SW09=1?
017422	005100			COM	R0	;CHANGE TO FLOATING 1
017424	000241			CLC		;CLEAR CARRY
017426	106100			ROLB	R0	;SHIFT BIT IN R0
017430	001356			BNE	69\$	;IF R0=0 THEN DONE
017432	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

017434	012737	000042	001226	TST42:	MOV	#42,TSTNO	
017442	012737	017610	001216		MOV	#TST43,NEXT	
017450	012737	017470	001220		MOV	#64\$,LOCK	
017456	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
017460	012702	000002			MOV	#2,R2	;MASTER CLEAR DMC11
							;SAVE REGISTER ADDRESS FOR TYPEOUT

017464	012700	000001		MOV	#1,R0	;START WITH BIT 0
017470			64\$:			
017470	010061	000004		MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
017474	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017476	122102			122100!2		;MOV DATA TO IBUS REGISTER 2
017500	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017502	021045			21005!<2*20>		;READ FROM IBUS REGISTER 2
017504	010005			MOV	R0,R5	;PUT EXPECTED IN R5
017506	116104	000005		MOVB	5(R1),R4	;PUT "FOUND" INTO R4
017512	120504			CMPB	R5,R4	;DATA CORRECT?
017514	001401			BEQ	65\$	;BR IF YES
017516	104005			EMT	5	;ERROR
017520	104401		65\$:	SCOP1		;SW09=1?
017522	000241			CLC		;CLEAR CARRY
017524	106100			ROLB	R0	;SHIFT BIT IN R0
017526	001360			BNE	64\$	;IF R0=0 THEN DONE
017530	012737	017544	001220	MOV	#67\$,LOCK	;NEW SCOP1
017536	012700	000001		MOV	#1,R0	;START WITH BIT 0
017542	005100		69\$:	COM	R0	;CHANGE TO FLOATING ZERO
017544			67\$:			
017544	010061	000004		MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
017550	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017552	122102			122100!2		;MOV DATA TO IBUS REGISTER 2
017554	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017556	021045			21005!<2*20>		;READ FROM IBUS REGISTER 2
017560	010005			MOV	R0,R5	;PUT EXPECTED IN R5
017562	116104	000005		MOVB	5(R1),R4	;PUT "FOUND" INTO R4
017566	120504			CMPB	R5,R4	;DATA CORRECT?
017570	001401			BEQ	68\$	;BR IF YES
017572	104005			EMT	5	;ERROR
017574	104401		68\$:	SCOP1		;SW09=1?
017576	005100			COM	R0	;CHANGE TO FLOATING 1
017600	000241			CLC		;CLEAR CARRY
017602	106100			ROLB	R0	;SHIFT BIT IN R0
017604	001356			BNE	69\$	;IF R0=0 THEN DONE
017606	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

017610	012737	000043	001226	TST43:	MOV	#43,TSTNO	
017616	012737	017764	001216		MOV	#TST44,NEXT	
017624	012737	017644	001220		MOV	#64\$,LOCK	
017632	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
017634	012702	000003			MOV	#3,R2	;MASTER CLEAR DMC11
017640	012700	000001			MOV	#1,R0	;SAVE REGISTER ADDRESS FOR TYPEOUT
017644				64\$:			;START WITH BIT 0
017644	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
017650	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017652	122103				122100!3		;MOV DATA TO IBUS REGISTER 3

017654	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017656	021065			21005!<3*20>		;READ FROM IBUS REGISTER 3
017660	010005			MOV R0,R5		;PUT EXPECTED IN R5
017662	116104	000005		MOVB 5(R1),R4		;PUT "FOUND" INTO R4
017666	120504			CMPB R5,R4		;DATA CORRECT?
017670	001401			BEQ 65\$		;BR IF YES
017672	104005			EMT 5		;ERROR
017674	104401		65\$:	SCOP1		;SW09=1?
017676	000241			CLC		;CLEAR CARRY
017700	106100			ROLB R0		;SHIFT BIT IN R0
017702	001360			BNE 64\$		;IF R0=0 THEN DONE
017704	012737	017720	001220	MOV #67\$,LOCK		;NEW SCOP1
017712	012700	000001		MOV #1,R0		;START WITH BIT 0
017716	005100		69\$:	COM R0		;CHANGE TO FLOATING ZERO
017720			67\$:			
017720	010061	000004		MOV R0,4(R1)		;PUT PATTERN INTO PORT4
017724	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017726	122103			122100!3		;MOV DATA TO IBUS REGISTER 3
017730	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017732	021065			21005!<3*20>		;READ FROM IBUS REGISTER 3
017734	010005			MOV R0,R5		;PUT EXPECTED IN R5
017736	116104	000005		MOVB 5(R1),R4		;PUT "FOUND" INTO R4
017742	120504			CMPB R5,R4		;DATA CORRECT?
017744	001401			BEQ 68\$		;BR IF YES
017746	104005			EMT 5		;ERROR
017750	104401		68\$:	SCOP1		;SW09=1?
017752	005100			COM R0		;CHANGE TO FLOATING 1
017754	000241			CLC		;CLEAR CARRY
017756	106100			ROLB R0		;SHIFT BIT IN R0
017760	001356			BNE 69\$		;IF R0=0 THEN DONE
017762	104400			SCOPE		;SCOPE THIS TEST

***** TEST 44 *****
;MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
;FLOAT A 1 THROUGH IBUS REGISTER 4
;FLOAT A 0 THROUGH IBUS REGISTER 4
;*****

				; TEST 44		

017764	012737	000044	001226	TST44:	MOV #44,TSTNO	
017772	012737	020140	001216		MOV #TST45,NEXT	
020000	012737	020020	001220		MOV #64\$,LOCK	
						;R1 CONTAINS BASE DMC11 ADDRESS
020006	104412			MSTCLR		;MASTER CLEAR DMC11
020010	012702	000004		MOV #4,R2		;SAVE REGISTER ADDRESS FOR TYPEOUT
020014	012700	000001		MOV #1,R0		;START WITH BIT 0
020020			64\$:			
020020	010061	000004		MOV R0,4(R1)		;PUT PATTERN INTO PORT4
020024	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020026	122104			122100!4		;MOV DATA TO IBUS REGISTER 4
020030	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020032	021105			21005!<4*20>		;READ FROM IBUS REGISTER 4
020034	010005			MOV R0,R5		;PUT EXPECTED IN R5
020036	116104	000005		MOVB 5(R1),R4		;PUT "FOUND" INTO R4
020042	120504			CMPB R5,R4		;DATA CORRECT?

020044	001401			BEQ	65\$				
020046	104005			EMT	5				
020050	104401		65\$:	SCOP1					
020052	000241			CLC					
020054	106100			ROLB	R0				
020056	001360			BNE	64\$				
020060	012737	020074	001220	MOV	#67\$,LOCK				
020066	012700	000001		MOV	#1,R0				
020072	005100		69\$:	COM	R0				
020074			67\$:						
020074	010061	000004		MOV	R0,4(R1)				
020100	104414			ROMCLK					
020102	122104			122100!4					
020104	104414			ROMCLK					
020106	021105			21005!<4*20>					
020110	010005			MOV	R0,R5				
020112	116104	000005		MOVB	5(R1),R4				
020116	120504			CMPB	R5,R4				
020120	001401			BEQ	68\$				
020122	104005			EMT	5				
020124	104401		68\$:	SCOP1					
020126	005100			COM	R0				
020130	000241			CLC					
020132	106100			ROLB	R0				
020134	001356			BNE	69\$				
020136	104400			SCOPE					

***** 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

020140	012737	000045	001226	TST45:	MOV	#45,TSTNO			
020146	012737	020314	001216		MOV	#TST46,NEXT			
020154	012737	020174	001220		MOV	#64\$,LOCK			
020162	104412				MSTCLR				
020164	012702	000005			MOV	#5,R2			
020170	012700	000001			MOV	#1,R0			
020174				64\$:					
020174	010061	000004			MOV	R0,4(R1)			
020200	104414				ROMCLK				
020202	122105				122100!5				
020204	104414				ROMCLK				
020206	021125				21005!<5*20>				
020210	010005				MOV	R0,R5			
020212	116104	000005			MOVB	5(R1),R4			
020216	120504				CMPB	R5,R4			
020220	001401				BEQ	65\$			
020222	104005				EMT	5			
020224	104401			65\$:	SCOP1				
020226	000241				CLC				
020230	106100				ROLB	R0			

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SAVE REGISTER ADDRESS FOR TYPEOUT
;START WITH BIT 0

;PUT PATTERN INTO PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOV DATA TO IBUS REGISTER 5
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;READ FROM IBUS REGISTER 5
;PUT EXPECTED IN R5
;PUT "FOUND" INTO R4
;DATA CORRECT?
;BR IF YES
;ERROR
;SW09=1?
;CLEAR CARRY
;SHIFT BIT IN R0

020232	001360			BNE	64\$		;IF R0=0 THEN DONE
020234	012737	020250	001220	MOV	#67\$,LOCK		;NEW SCOP1
020242	012700	000001		MOV	#1,R0		;START WITH BIT 0
020246	005100			COM	R0		;CHANGE TO FLOATING ZERO
020250			69\$:				
			67\$:				
020250	010061	000004		MOV	R0,4(R1)		;PUT PATTERN INTO PORT4
020254	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020256	122105			122100!5			;MOV DATA TO IBUS REGISTER 5
020260	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020262	021125			21005!<5*20>			;READ FROM IBUS REGISTER 5
020264	010005			MOV	R0,R5		;PUT EXPECTED IN R5
020266	116104	000005		MOVB	5(R1),R4		;PUT "FOUND" INTO R4
020272	120504			CMPB	R5,R4		;DATA CORRECT?
020274	001401			BEQ	68\$		;BR IF YES
020276	104005			EMT	5		;ERROR
020300	104401		68\$:	SCOP1			;SW09=1?
020302	005100			COM	R0		;CHANGE TO FLOATING 1
020304	000241			CLC			;CLEAR CARRY
020306	106100			ROLB	R0		;SHIFT BIT IN R0
020310	001356			BNE	69\$		;IF R0=0 THEN DONE
020312	104400			SCOPE			;SCOPE THIS TEST

***** TEST 46 *****
;MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
;FLOAT A 1 THROUGH IBUS REGISTER 6
;FLOAT A 0 THROUGH IBUS REGISTER 6
:*****

; TEST 46

020314	012737	000046	001226	TST46:	MOV	#46,TSTNO	
020322	012737	020470	001216		MOV	#TST47,NEXT	
020330	012737	020350	001220		MOV	#64\$,LOCK	
							;R1 CONTAINS BASE DMC11 ADDRESS
020336	104412			MSTCLR			;MASTER CLEAR DMC11
020340	012702	000006		MOV	#6,R2		;SAVE REGISTER ADDRESS FOR TYPEOUT
020344	012700	000001		MOV	#1,R0		;START WITH BIT 0
020350			64\$:				
020350	010061	000004		MOV	R0,4(R1)		;PUT PATTERN INTO PORT4
020354	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020356	122106			122100!6			;MOV DATA TO IBUS REGISTER 6
020360	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020362	021145			21005!<6*20>			;READ FROM IBUS REGISTER 6
020364	010005			MOV	R0,R5		;PUT EXPECTED IN R5
020366	116104	000005		MOVB	5(R1),R4		;PUT "FOUND" INTO R4
020372	120504			CMPB	R5,R4		;DATA CORRECT?
020374	001401			BEQ	65\$		;BR IF YES
020376	104005			EMT	5		;ERROR
020400	104401		65\$:	SCOP1			;SW09=1?
020402	000241			CLC			;CLEAR CARRY
020404	106100			ROLB	R0		;SHIFT BIT IN R0
020406	001360			BNE	64\$		;IF R0=0 THEN DONE
020410	012737	020424	001220	MOV	#67\$,LOCK		;NEW SCOP1
020416	012700	000001		MOV	#1,R0		;START WITH BIT 0
020422	005100		69\$:	COM	R0		;CHANGE TO FLOATING ZERO
020424			67\$:				

020424	010061	000004		MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
020430	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020432	122106			122100!6		;MOV DATA TO IBUS REGISTER 6
020434	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020436	021145			21005!<6*20>		;READ FROM IBUS REGISTER 6
020440	010005			MOV	R0,R5	;PUT EXPECTED IN R5
020442	116104	000005		MOVB	5(R1),R4	;PUT "FOUND" INTO R4
020446	120504			CMPB	R5,R4	;DATA CORRECT?
020450	001401			BEQ	68\$	;BR IF YES
020452	104005			EMT	5	;ERROR
020454	104401		68\$:	SCOP1		;SW09=1?
020456	005100			COM	R0	;CHANGE TO FLOATING 1
020460	000241			CLC		;CLEAR CARRY
020462	106100			ROLB	R0	;SHIFT BIT IN R0
020464	001356			BNE	69\$	;IF R0=0 THEN DONE
020466	104400			SCOPE		;SCOPE THIS TEST

***** TEST 47 *****
;MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
;FLOAT A 1 THROUGH IBUS REGISTER 7
;FLOAT A 0 THROUGH IBUS REGISTER 7

; TEST 47

020470	012737	000047	001226	TST47:	MOV	#47,TSTNO	
020476	012737	020644	001216		MOV	#TST50,NEXT	
020504	012737	020524	001220		MOV	#64\$,LOCK	
020512	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
020514	012702	000007			MOV	#7,R2	;MASTER CLEAR DMC11
020520	012700	000001			MOV	#1,R0	;SAVE REGISTER ADDRESS FOR TYPE0..T
020524				64\$:			;START WITH BIT 0
020524	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
020530	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020532	122107				122100!7		;MOV DATA TO IBUS REGISTER 7
020534	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020536	021165				21005!<7*20>		;READ FROM IBUS REGISTER 7
020540	010005				MOV	R0,R5	;PUT EXPECTED IN R5
020542	116104	000005			MOVB	5(R1),R4	;PUT "FOUND" INTO R4
020546	120504				CMPB	R5,R4	;DATA CORRECT?
020550	001401				BEQ	65\$	;BR IF YES
020552	104005				EMT	5	;ERROR
020554	104401		65\$:		SCOP1		;SW09=1?
020556	000241				CLC		;CLEAR CARRY
020560	106100				ROLB	R0	;SHIFT BIT IN R0
020562	001360				BNE	64\$	;IF R0=0 THEN DONE
020564	012737	020600	001220		MOV	#67\$,LOCK	;NEW SCOP1
020572	012700	000001			MOV	#1,R0	;START WITH BIT 0
020576	005100			69\$:	COM	R0	;CHANGE TO FLOATING ZERO
020600				67\$:			
020600	010061	000004			MOV	R0,4(R1)	;PUT PATTERN INTO PORT4
020604	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020606	122107				122100!7		;MOV DATA TO IBUS REGISTER 7
020610	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020612	021165				21005!<7*20>		;READ FROM IBUS REGISTER 7

Address	Hex	Dec	Label	Instruction	Comment
020614	010005			MOV R0,R5	;PUT EXPECTED IN R5
020616	116104	000005		MOVB 5(R1),R4	;PUT "FOUND" INTO R4
020622	120504			CMPB R5,R4	;DATA CORRECT?
020624	001401			BEQ 68\$	;BR IF YES
020626	104005			EMT 5	;ERROR
020630	104401		68\$:	SCOP1	;SW09=1?
020632	005100			COM R0	;CHANGE TO FLOATING 1
020634	000241			CLC	;CLEAR CARRY
020636	106100			ROLB R0	;SHIFT BIT IN R0
020640	001356			BNE 69\$	;IF R0=0 THEN DONE
020642	104400			SCOPE	;SCOPE THIS TEST
***** 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					
020644	012737	000050	001226	TST50: MOV #50,TSTNO	
020652	012737	021072	001216	MOV #TST51,NEXT	
020660	012737	020676	001220	MOV #1\$,LOCK	
020666	104412			MSTCLR	;R1 CONTAINS BASE DMC11 ADDRESS
020670	012700	000001		MOV #1,R0	;MASTER CLEAR DMC11
020674	005002			CLR R2	;START WITH A ONE
020676	010203		1\$:	MOV R2,R3	;R2 CONTAINS ADDRESS OF REGISTER
020700	010061	000004		MOV R0,4(R1)	;R3=REGISTER ADDRESS
020704	042737	000017	020720	BIC #17,5\$	;WRITE DATA TO PORT4
020712	050337	020720		BIS R3,5\$	;CLEAR ADDRESS FIELD OF INSTRUCTION
020716	104414			ROMCLK	;ADD ADDRESS TO INSTRUCTION
020720	122100		5\$:	122100	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020722	006303			ASL R3	;MOVE DATA TO IBUS REGISTER
020724	006303			ASL R3	;SHIFT ADDRESS
020726	006303			ASL R3	;4 TIMES TO GET
020730	006303			ASL R3	;IT TO BITS 4-7
020732	042737	000360	020746	BIC #360,6\$	;OF NEXT INSTRUCTION
020740	050337	020746		BIS R3,6\$	;CLEAR ADDRESS FIELD
020744	104414			ROMCLK	;ADD ADDRESS TO INSTRUCTION
020746	021005		6\$:	21005	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020750	010005			MOV R0,R5	;READ FROM IBUS REGISTER
020752	116104	000005		MOVB 5(R1),R4	;PUT "EXPECTED" IN R5
020756	120504			CMPB R5,R4	;PUT "FOUND" IN R4
020760	001401			BEQ 2\$	;IS DATA CORRECT?
020762	104005			EMT 5	;BR IF YES
020764	104401		2\$:	SCOP1	;DATA ERROR
020766	005200			INC R0	;SW09=1?
020770	005202			INC R2	;INCREMENT PATTERN
020772	022702	000010		CMP #7+1,R2	;INCREMENT REGISTER ADDRESS
020776	001337			BNE 1\$	;LAST ADDRESS DONE?
021000	012737	021016	001220	MOV #3\$,LOCK	;BR IF NO
021006	012700	000001		MOV #1,R0	;NEW SCOP1
021012	005002			CLR R2	;RESTART PATTERN TO 1
021014	005003			CLR R3	;RESTART AT ADDRESS 0
021016	042737	000360	021032	3\$:	;RESTART AT ADDRESS 0
				BIC #360,7\$	;CLEAR ADDRESS FIELD OF INSTRUCTION

```

021024 050337 021032      BIS      R3,7$      ;ADD ADDRESS TO INSTRUCTION
021030 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021032 021005      7$:    21005      ;READ FROM IBUS REGISTER
021034 010005      MOV      R0,R5      ;PUT "EXPECTED" IN R5
021036 116104 000005      MOVB     5(R1),R4 ;PUT "FOUND" IN R5
021042 120504      CMPB     R5,R4      ;DATA CORRECT?
021044 001401      BEQ      4$          ;BR IF YES
021046 104005      EMT      5          ;DUAL ADDRESSING ERROR
021050 104401      4$:    SCOP1        ;SW09=1?
021052 005200      INC      R0          ;INCREMENT PATTERN
021054 005202      INC      R2          ;NEXT ADDRESS
021056 062703 000020      ADD      #20,R3 ;ADD 1 TO ADDRESS IN R3(SHIFTED 4 TIMES)
021062 022702 000010      CMP      #7+1,R2 ;LAST ADDRESS DONE?
021066 001353      BNE      3$          ;BR IF NO
021070 104400      SCOPE              ;SCOPE THIS TEST

```

38

***** TEST 51 *****
;MICRO PROCESSOR BR REGISTER TEST
;FLOAT A 1 THROUGH THE BR
;FLOAT A 0 THROUGH THE BR

; TEST 51

```

021072 012737 000051 001226 TST51: MOV      #51,TSTNO
021100 012737 021242 001216      MOV      #TST52,NEXT
021106 012737 021122 001220      MOV      #64$,LOCK

021114 104412      MSTCLR              ;R1 CONTAINS BASE DMC11 ADDRESS
021116 012700 000001      MOV      #1,R0 ;MASTER CLEAR DMC11
021122              64$:    MOV      R0,4(R1) ;START PATTERN WITH BIT0
021122 010061 000004      ROMCLK     120500 ;WRITE PATTERN IN PORT4
021126 104414      ROMCLK     061225      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021130 120500      MOV      R0,R5      ;MOVE DATA TO THE BR REGISTER
021132 104414      MOVB     5(R1),R4 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021134 061225      CMPB     R5,R4      ;MOVE BR TO PORT 5
021136 010005      BEQ      65$        ;PUT "EXPECTED" IN R5
021140 116104 000005      EMT      6          ;PUT "FOUND" IN R4
021144 120504      BEQ      65$        ;DATA CORRECT?
021146 001401      EMT      6          ;BR IF YES
021150 104006      65$:    SCOP1        ;DATA ERROR
021152 104401      CLC                  ;CLEAR CARRY
021154 000241      ROLB      R0          ;SHIFT BIT IN R0
021156 106100      BNE      64$        ;DONE IF R0=0
021160 001360      MOV      #67$,LOCK ;NEW SCOP1
021162 012737 021176 001220      MOV      #1,R0 ;START PATTERN WITH BIT0
021170 012700 000001      COM      R0      ;CHANGE TO FLOATING ZERO
021174 005100      69$:    MOV      R0,4(R1)
021176              67$:    ROMCLK     120500 ;WRITE PATTERN IN PORT4
021202 104414      ROMCLK     061225      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021204 120500      MOV      R0,R5      ;MOVE DATA TO THE BR REGISTER
021206 104414      MOVB     5(R1),R4 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021210 061225      MOV      R0,R5      ;MOVE BR TO PORT 5
021212 010005      BEQ      65$        ;PUT "EXPECTED" IN R5
021214 116104 000005      EMT      6          ;PUT "FOUND" IN R4

```



```

021220 120504
021222 001401
021224 104006
021226 104401
021230 005100
021232 000241
021234 106100
021236 001356
021240 104400
42

```

```

CMPB R5,R4 ;DATA CORRECT?
BEQ 68$ ;BR IF YES
EMT 6 ;DATA ERROR
68$: SCOP1
COM R0 ;CHANGE BACK TO A ONE
CLC ;CLEAR CARRY
ROLB R0 ;SHIFT BIT IN R0
BNE 69$ ;DONE IF R0=0
SCOPE ;SCOPE THIS TEST

```

```

;***** TEST 52 *****
;*SCRATCH PAD TEST
;*FLOAT A 1 THROUGH EACH SCRATCH PAD LOCATION
;*FLOAT A 0 THROUGH EACH SCRATCH PAD LOCATION
;*****

```

; TEST 52

```

021242 012737 000052 001226 TST52: MOV #52,TSTNO
021250 012737 021510 001216 MOV #TST53,NEXT
021256 012737 021274 001220 MOV #64$,LOCK

021264 104412 MSTCLR
021266 005002 CLR R2
021270 012700 000001 MOV #1,R0
021274 042737 000017 021314 64$: BIC #17,65$
021302 050237 021314 BIS R2,65$
021306 010061 000004 MOV R0,4(R1)
021312 104414 ROMCLK
021314 123100 65$: 123100
021316 042737 000017 021332 BIC #17,66$
021324 050237 021332 BIS R2,66$
021330 104414 ROMCLK
021332 040600 66$: 040600
021334 104414 ROMCLK
021336 061225 061225
021340 010005 MOV R0,R5
021342 116104 000005 MOVB 5(R1),R4
021346 120504 CMPB R5,R4
021350 001401 BEQ 67$
021352 104007 EMT 7
021354 104401 67$: SCOP1
021356 000241 CLC
021360 106100 ROLB R0
021362 001344 BNE 64$
021364 012737 021400 001220 MOV #69$,LOCK
021372 012700 000001 MOV #1,R0
021376 005100 73$: COM R0
021400 042737 000017 021420 69$: BIC #17,70$
021406 050237 021420 BIS R2,70$
021412 010061 000004 MOV R0,4(R1)
021416 104414 ROMCLK
021420 123100 70$: 123100
021422 042737 000017 021436 BIC #17,71$
021430 050237 021436 BIS R2,71$
021434 104414 ROMCLK

```

```

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;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 R0
;DONE IF R0=0
;NEW SCOP1
;START WITH BIT0
;CHANGE TO FLOATING ZERO
;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

```

43

021436	040600		71:	040600	;MOV SP TO BR
021440	104414			ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021442	061225			061225	;MOVE BR TO PORT5
021444	010005			MOV	R0,R5
021446	116104	000005		MOVB	5(R1),R4
021452	120504			CMPB	R5,R4
021454	001401			BEQ	72:
021456	104007			EMT	7
021460	104401		72:	SCOP1	;SW09=1?
021462	005100			COM	R0
021464	000241			CLC	;CHANGE BACK TO A ONE
021466	106100			ROLB	R0
021470	001342			BNE	73:
021472	012700	000001		MOV	#1,R0
021476	005202			INC	R2
021500	022702	000020		CMP	#20,R2 ;LAST ADDRESS?
021504	001273			BNE	64:
021506	104400			SCOPE	;BR IF NO
					;SCOPE THIS TEST

***** TEST 53 *****
;*SCRATCH PAD DUAL ADDRESSING TEST
;*WRITE AN INCREMENTING PATTERN IN ALL SP LOCATIONS
;*READ ALL SP LOCATIONS TO VERIFY CORRECT ADDRESSING
;*****

; TEST 53

021510	012737	000053	001226	TST53:	MOV	#53,TSTNO	
021516	012737	021732	001216		MOV	#TST54,NEXT	
021524	012737	021542	001220		MOV	#1\$,LOCK	
021532	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
021534	012700	000001			MOV	#1,R0	;MASTER CLEAR DMC11
021540	005003				CLR	R3	;START WITH A 1
021542	010302			1:	MOV	R3,R2	;ADDRESS 0
021544	042737	000017	021564		BIC	#17,2:	;MOVE ADDRESS TO R2
021552	050237	021564			BIS	R2,2:	;CLEAR ADDRESS FIELD
021556	010061	000004			MOV	R0,4(R1)	;ADD ADDRESS TO INSTRUCTION
021562	104414				ROMCLK		;WRITE PATTERN TO PORT4
021564	123100			2:	123100		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021566	042737	000017	021602		BIC	#17,3:	;WRITE SP(ADDRESS IN R2)
021574	050237	021602			BIS	R2,3:	;CLEAR ADDRESS FIELD OF INSTRUCTION
021600	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
021602	060600			3:	60600		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021604	104414				ROMCLK		;MOV SP TO BR
021606	061225				61225		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021610	010005				MOV	R0,R5	;MOVE BR TO PORT5
021612	116104	000005			MOVB	5(R1),R4	;PUT "EXPECTED" IN R5
021616	120504				CMPB	R5,R4	;PUT "FOUND" IN R4
021620	001401				BEQ	4:	;DATA CORRECT?
021622	104007				EMT	7	;BR IF YES
021624	104401			4:	SCOP1		;DATA ERROR
021626	005200				INC	R0	;SW09=0
021630	005203				INC	R3	;INCREMENT PATTERN
021632	022703	000020			CMP	#20,R3	;NEXT ADDRESS
021636	001341				BNE	1:	;LAST ADDRESS DONE?
							;BR IF NO

47

021640	012737	021654	001220	MOV	#5\$,LOCK	;NEW SCOP1
021646	012700	000001		MOV	#1,R0	;RESTART PATTERN AT 1
021652	005003			CLR	R3	;RESTART AT ADDRESS ZERO
021654	010302			MOV	R3,R2	;PUT ADDRESS IN R2
021656	042737	000017	021672	BIC	#17,6\$	;CLEAR ADDRESS FIELD OF INSTRUCTION
021664	050237	021672		BIS	R2,6\$	;ADD ADDRESS TO INSTRUCTION
021670	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021672	060600			60600		;MOV SP TO BR
021674	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021676	061225			61225		;MOV BR TO PORT5
021700	010005			MOV	R0,R5	;PUT "EXPECTED" IN R5
021702	116104	000005		MOVB	5(R1),R4	;PUT "FOUND" IN R4
021706	120504			CMPB	R5,R4	;DATA CORRECT?
021710	001401			BEQ	7\$	;BR IF YES
021712	104007			EMT	7	;SP ADDRESSING ERROR
021714	104401			SCOP1		;SW09=1?
021716	005200			INC	R0	;INCREMENT PATTERN
021720	005203			INC	R3	;NEXT ADDRESS
021722	022703	000020		CMP	#20,R3	;LAST ADDRESS DONE?
021726	001352			BNE	5\$	;BR IF NO
021730	104400			SCOPE		;SCOPE THIS TEST

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

; TEST 54

48

021732	012737	000054	001226	TST54:	MOV	#54,TSTNO	
021740	012737	022026	001216		MOV	#TST55,NEXT	
021746	000005			RESET			;R1 CONTAINS BASE DMC11 ADDRESS
021750	005011			CLR	(R1)		;BUS RESET
021752	004537	034642		JSR	R5,SETVEC		;CLEAR RUN
021756	022020			3\$			;SET UP VECTORS
021760	022016			2\$			;XX0
021762	340	340		.BYTE	340,340		;XX4
021764	012737	000340	177776	1\$:	MOV	#340,PS	;LEVEL 7
021772	012761	000200	000004		MOV	#200,4(R1)	;PS = LEVEL 7
022000	104414			ROMCLK			;WRITE PORT4
022002	121111			121111			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022004	005037	177776		CLR	PS		;SET BR RQ IN IBUS* REG 11
022010	000240			NOP			;ALLOW INTERRUPT
022012	104010			EMT	10		;NO INTERRUPT
022014	000403			BR	4\$		
022016	104011			EMT	11		;WRONG VECTOR
022020	012706	001200		3\$:	MOV	#STACK,SP	;RESET STACK
022024	104400			4\$:	SCOPE		;SCOPE THIS TEST

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

```

; TEST 55
-----
022026 012737 000055 001226 TST55: MOV #55,TSTNO
022034 012737 022120 001216 MOV #TST56,NEXT

022042 104412 MSTCLR
022044 004537 034642 JSR R5,SETVEC
022050 022110 2#
022052 022112 3#
022054 340 340 .BYTE 340,340
022056 012737 000340 177776 1#: MOV #340,PS
022064 012761 000300 000004 MOV #300,4(R1)
022072 104414 ROMCLK
022074 121111 121111
022076 005037 177776 CLR PS
022102 000240 NOP
022104 104010 EMT 10
022106 000403 BR 4#
022110 104011 2#: EMT 11
022112 012706 001200 3#: MOV #STACK,SP
022116 104400 4#: SCOPE

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;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
;ALLOW INTERRUPT

;NO INTERRUPT

;WRONG VECTOR
;RESET STACK
;SCOPE THIS TEST

```

49

```

***** TEST 56 *****
;PRIORITY INTERRUPT TESTS
;SET PS TO ALL BR LEVELS EQUAL OR GREATER THAN
;THE DMC11 LEVEL,VERIFY THAT DMC11 DOES NOT INTERRUPT
*****

```

```

; TEST 56
-----
022120 012737 000056 001226 TST56: MOV #56,TSTNO
022126 012737 022240 001216 MOV #TST57,NEXT

022134 104412 MSTCLR
022136 012702 000340 MOV #340,R2
022142 010237 177776 MOV R2,PS
022146 013700 001366 MOV STAT1,R0
022152 006200 ASR R0
022154 006200 ASR R0
022156 006200 ASR R0
022160 006200 ASR R0
022162 042700 177437 BIC #177437,R0
022166 004537 034642 JSR R5,SETVEC
022172 022234 2#
022174 022234 2#
022176 340 340 .BYTE 340,340
022200 012761 000200 000004 4#: MOV #200,4(R1)
022206 104414 ROMCLK
022210 121111 121111
022212 010237 177776 5#: MOV R2,PS
022216 000240 NOP
022220 020002 CMP R0,R2
022222 001403 BEQ 1#
022224 162702 000040 SUB #40,R2
022230 000770 BR 5#
022232 104400 1#: SCOPE

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;PUT LEVEL 7 IN R2
;SET PRIORITY TO 7
;GET BR LEVEL OF DMC11
;SHIFT R0 4 TIMES
;TO GET PROPER LEVEL

;CLEAR UNWANTED BITS
;SET UP VECTORS
;A VECTOR
;B VECTOR
;PRIORITY 7
;LOAD PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET BR REQUEST
;PUT LEVEL IN R2 IN PS

;IS PRESENT PS LEVEL = TO DMC LEVEL
;BR IF YES
;NO GET NEXT LOWER LEVEL IN R2
;AND CONTINUE WITH TEST
;SCOPE THIS TEST

```


022234 104020
022236 000002
502\$: EMT 20 ;ERROR UNEXPECTED INTERRUPT
RTI;***** TEST 57 *****
;PRIORITY INTERRUPT TESTS
;SET PS TO ALL BR LEVELS LESS THAN THE DMC11 LEVEL
;VERIFY THAT THE DMC11 WILL INTERRUPT
;*****

; TEST 57

022240	012737	000057	001226	TST57:	MOV	#57,TSTNO	
022246	012737	022404	001216		MOV	#TST60,NEXT	
022254	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
022256	012702	000340			MOV	#340,R2	;MASTER CLEAR DMC11
022262	010237	177776			MOV	R2,PS	;PUT LEVEL 7 IN R2
022266	013700	001366			MOV	STAT1,R0	;SET PRIORITY TO 7
022272	006200				ASR	R0	;GET BR LEVEL OF DMC11
022274	006200				ASR	R0	;SHIFT R0 4 TIMES
022276	006200				ASR	R0	;TO GET PROPER LEVEL
022300	006200				ASR	R0	
022302	042700	177437			BIC	#177437,R0	;CLEAR UNWANTED BITS
022306	010002				MOV	R0,R2	;PUT DMC LEVEL IN R2
022310	162702	000040			SUB	#40,R2	;GET NEXT LOWER LEVEL IN R2
022314	004537	034642			JSR	R5,SETVEC	;SET UP VECTORS
022320	022366				2\$		;A VECTOR
022322	022374				3\$		;B VECTOR
022324	340	340			.BYTE	340,340	;PRIORITY 7
022326	012761	000200	000004	4\$:	MOV	#200,4(R1)	;LOAD PORT4
022334	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022336	121111				121111		;SET BR REQUEST
022340	010237	177776		5\$:	MOV	R2,PS	;PUT LEVEL IN R2 IN PS
022344	000240				NOP		
022346	104010				EMT	10	;ERROR, NO INTERRUPT
022350	022702	000140		6\$:	CMP	#140,R2	;IS IT DOWN TO LEVEL 3 YET?
022354	001403				BEQ	1\$	;YES,DMC DID NOT INTERRUPT, ERROR
022356	162702	000040			SUB	#40,R2	;PUT NEXT LOWER LEVEL IN R2
022362	000761				BR	4\$	;CONTINUE TEST
022364	104400			1\$:	SCOPE		;SCOPE THIS TEST
022366	012716	022350		2\$:	MOV	#6\$,(SP)	;SET UP FOR RTI
022372	000002				RTI		
022374	104011			3\$:	EMT	11	;ERROR, WRONG VECTOR
022376	012716	022350			MOV	#6\$,(SP)	;SET UP FOR RTI
022402	000002				RTI		

54

;***** TEST 60 *****
;NPR TEST
;TEST OF DAT0, 1 WORD FROM UPROC TO 11 MEMORY
;*****

; TEST 60

022404	012737	000060	001226	TST60:	MOV	#60,TSTNO
022412	012737	022510	001216		MOV	#TST61,NEXT

022420 000005
022422 005011
022424 005061 000004
022430 004537 034664
022434 000000
022436 177777
022440 022506
022442 022504
022444 005037 022504
022450 012761 000021 000004
022456 104414
022460 121110
022462 000240
022464 012705 177777
022470 013704 022504
022474 020504
022476 001401
022500 104012
022502 104400
022504 000000
022506 000000

4\$:
2\$:
3\$:

RESET
CLR (R1)
CLR 4(R1)
JSR R5,NPRSET
0
-1
3\$
2\$
CLR 2\$
MOV #21,4(R1)
ROMCLK
121110
NOP
MOV #-1,R5
MOV 2\$,R4
CMP R5,R4
BEQ 4\$
EMT 12
SCOPE
0
0

;R1 CONTAINS BASE DMC11 ADDRESS
;BUS RESET
;CLEAR RUN
;CLR PORT4
;SET UP IBUS REG 0-7
;IN DATA
;OUT DATA
;IN BA
;OUT BA
;CLEAR 2\$
;WRITE PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET NPR BITS IN IBUS* REG 11

;PUT "EXPECTED" IN R5
;PUT "FOUND" IN R4
;DATA CORRECT?
;BR IF YES
;ERROR NPR FAILED
;SCOPE THIS TEST
;OUT BA
;IN BA

55

***** TEST 61 *****
;NPR TEST
;TEST OF DATI, 1 WORD FROM 11 MEMORY TO UPROC
;*****

; TEST 61

022510 012737 000061 001226 TST61:
022516 012737 022624 001216

022524 104412
022526 005061 000004
022532 004537 034664
022536 000000
022540 177777
022542 022622
022544 022620
022546 012737 177777 022622
022554 012761 000001 000004
022562 104414
022564 121110
022566 000240
022570 012705 177777
022574 104414
022576 021004
022600 104414
022602 021025
022604 016104 000004
022610 020504
022612 001401
022614 104012
022616 104400
022620 000000

4\$:
2\$:

MOV #61,TSTNO
MOV #TST62,NEXT

MSTCLR
CLR 4(R1)
JSR R5,NPRSET
0
-1
3\$
2\$
MOV #-1,3\$
MOV #1,4(R1)
ROMCLK
121110
NOP
MOV #-1,R5
ROMCLK
021004
ROMCLK
021025
MOV 4(R1),R4
CMP R5,R4
BEQ 4\$
EMT 12
SCOPE
0

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;CLR PORT4
;SET UP IBUS REG 0-7
;IN DATA
;OUT DATA
;IN BA
;OUT BA
;PUT DATA IN 3\$
;WRITE PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET NPR BITS IN IBUS* REG 11

;PUT "EXPECTED" IN R5
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOVE IN DATA LOW BYTE TO PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOVE IN DATA HIGH BYTE TO PORT5
;PUT "FOUND" IN R4
;DATA CORRECT?
;BR IF YES
;ERROR NPR FAILED
;SCOPE THIS TEST
;OUT BA

56 022622 000000

3\$: 0 ;IN BA

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

; TEST 62

022624 012737 000062 001226 TST62:
022632 012737 022726 001216MOV #62,TSTNO
MOV #TST63,NEXT022640 104412
022642 005061 000004
022646 004537 034664MSTCLR
CLR 4(R1)
JSR R5,NPRSET;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;CLR PORT4022652 000000
022654 1777770
-1;SET UP IBUS REG 0-7
;IN DATA
;OUT DATA

022656 022724

3\$

;IN BA

022660 022723

2\$+1

;OUT BA

022662 005037 022722

CLR 2\$

;CLEAR 2\$

022666 012761 000221 000004

MOV #221,4(R1)

;WRITE PORT4

022674 104414

ROMCLK

;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

022676 121110

121110

;SET NPR BITS IN IBUS* REG 11

022700 000240

NOP

022702 012705 177400

MOV #177400,R5

;PUT "EXPECTED" IN R5

022706 013704 022722

MOV 2\$,R4

;PUT "FOUND" IN R4

022712 020504

CMP R5,R4

;DATA CORRECT?

022714 001401

BEQ 4\$

;BR IF YES

022716 104012

EMT 12

;ERROR NPR FAILED

022720 104400

4\$: SCOPE

;SCOPE THIS TEST

022722 000000

2\$: 0

;OUT BA

022724 000000

3\$: 0

;IN BA

57

***** 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

022726 012737 000063 001226 TST63:
022734 012737 023064 001216MOV #63,TSTNO
MOV #TST64,NEXT022742 104412
022744 013737 001412 022772
022752 013737 001412 022770
022760 004537 034664MSTCLR
MOV DMP04,1\$
MOV DMP04,2\$
JSR R5,NPRSET;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;USE SEL4 FOR ADDRESS

022764 000000

0

;USE SEL4 FOR ADDRESS

022766 125252

125252

;LOAD BA AND DATA

022770 000000

0

;IN DATA

022772 000000

2\$: 0

;OUT DATA

022774 012761 000014 000004

1\$: 0

;IN BA

023002 104414

MOV #14,4(R1)

;OUT BA

023004 121111

ROMCLK

;LOAD SEL 4 WITH OUT BA16 AND 17

121111

;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

;SET OUTBA 16 AND 17

58

023006	012761	000021	000004	MOV	#21,4(R1)	;LOAD SEL4
023014	012761	121110	000006	MOV	#121110,6(R1)	;PUT INSTRUCTION IN SEL6
023022	012711	003000		MOV	#BIT9!BIT10,(R1)	;SET CROMI AND CROMO!!
023026	052711	000400		BIS	#BIT8,(R1)	;CLOCK IT!
023032	000240			NOP		;WAIT FOR NPR
023034	012705	121110		MOV	#121110,R5	;PUT "EXPECTED" IN R5
023040	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023042	021044			021044		;MOVE OUT DATA LB TO SEL4
023044	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023046	021065			021065		;MOVE OUT DATA HB TO SEL5
023050	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
023054	020504			CMP	R5,R4	;CORRECT RESULTS ?
023056	001401			BEQ	3\$	;BR IF YES
023060	104012			EMT	12	;ERROR BA 16 AND 17 FAILED
023062	104400			3\$: SCOPE		;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

59

023064	012737	000064	001226	TST64:	MOV	#64,TSTNO	
023072	012737	023210	001216		MOV	#TST65,NEXT	
023100	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
023102	013737	001412	023130		MOV	DMP04,1\$	;MASTER CLEAR DMC11
023110	013737	001412	023126		MOV	DMP04,2\$	;USE SEL4 FOR ADDRESS
023116	004537	034664			JSR	R5,NPRSET	;USE SEL4 FOR ADDRESS
023122	000000				0		;LOAD BA AND DATA
023124	125252				125252		;IN DATA
023126	000000			2\$: 0			;OUT DATA
023130	000000			1\$: 0			;IN BA
023132	012761	000015	000004		MOV	#15,4(R1)	;OUT BA
023140	012761	121110	000006		MOV	#121110,6(R1)	;LOAD SEL4
023146	012711	003000			MOV	#BIT9!BIT10,(R1)	;PUT INSTRUCTION IN SEL6
023152	052711	000400			BIS	#BIT8,(R1)	;SET CROMI AND CROMO!!
023156	000240				NOP		;CLOCK IT!
023160	012705	121110			MOV	#121110,R5	;WAIT FOR NPR
023164	104414				ROMCLK		;PUT "EXPECTED" IN R5
023166	021004				021004		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023170	104414				ROMCLK		;MOVE IN DATA LB TO SEL4
023172	021025				021025		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023174	016104	000304			MOV	4(R1),R4	;MOVE IN DATA HB TO SEL5
023200	020504				CMP	R5,R4	;PUT "FOUND" IN R4
023202	001401				BEQ	3\$	;CORRECT RESULTS ?
023204	104012				EMT	12	;BR IF YES
023206	104400			3\$: SCOPE			;ERROR BA 16 AND 17 FAILED
							;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

023210	012737	000065	001226	TST65:	MOV	#65,TSTNO	
023216	012737	023320	001216		MOV	#TST66,NEXT	
023224	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
023226	004537	034664			JSR	R5,NPRSET	;MASTER CLEAR DMC11
023232	000000				0		;LOAD IBUS REGISTERS 0-7
023234	000000				0		;IN DATA
023236	177320				177320		;OUT DATA
023240	177320				177320		;IN BA
023242	012761	000014	000004		MOV	#14,4(R1)	;OUT BA
023250	104414				ROMCLK		;SET OUT BA BITS 16+17 IN PORT4
023252	121111				121111		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023254	012761	000021	000004		MOV	#21,4(R1)	;SET OUTBA 16 AND 17
023262	104414				ROMCLK		;SET NPR REQUEST BITS IN PORT4
023264	121110				121110		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023266	000240				NOP		;MOV IBUS* 4 TO IBUS* 10
023270	104414				ROMCLK		
023272	121225				121225		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023274	012705	000001			MOV	#1,R5	;MOV IBUS*11 TO IBUS*5
023300	116104	000005			MOVB	5(R1),R4	;PUT "EXPECTED" IN R5
023304	042704	177776			BIC	#177776,R4	;PUT "FOUND" IN R4
023310	020504				CMP	R5,R4	;CLEAR UNWANTED BITS
023312	001401				BEQ	14	;DATA CORRECT?
023314	104012				EMT	12	;BR IF YES
023316	104400			14:	SCOPE		;ERROR NON-EXISTENT MEM BIT FAILED TO SET

60

;***** 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

023320	012737	000066	001226	TST66:	MOV	#66,TSTNO	
023326	012737	023426	001216		MOV	#TST67,NEXT	
023334	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
023336	004537	034664			JSR	R5,NPRSET	;MASTER CLEAR DMC11
023342	000000				0		;LOAD IBUS REGISTERS 0-7
023344	000000				0		;IN DATA
023346	177320				177320		;OUT DATA
023350	177320				177320		;IN BA
023352	005061	000004			CLR	4(R1)	;OUT BA
023356	104414				ROMCLK		
023360	121111				121111		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023362	012761	000015	000004		MOV	#15,4(R1)	;CLEAR NON-EXISTENT BIT
023370	104414				ROMCLK		;SET NPR REQUEST BITS IN PORT4
023372	121110				121110		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023374	000240				NOP		;MOV IBUS* 4 TO IBUS* 10
023376	104414				ROMCLK		
023400	121225				121225		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

;MOV IBUS*11 TO IBUS*5

023402 012705 000001
023406 116104 000005
023412 042704 177776
023416 020504
023420 001401
023422 104012
023424 104400

1\$:

MOV #1,R5
MOVB 5(R1),R4
BIC #177776,R4
CMP R5,R4
BEQ 1\$
EMT 12
SCOPE

;PUT "EXPECTED" IN R5
;PUT "FOUND" IN R4
;CLEAR UNWANTED BITS
;DATA CORRECT?
;BR IF YES
;ERROR NON-EXISTENT MEM BIT FAILED TO SET
;SCOPE THIS TEST

61

***** TEST 67 *****
;NPR TEST
;USING DAT0, NPR A BINARY COUNT (0-377)
;FROM MICRO-PROCESSOR TO ALL AVAILABLE MEMORY
;*****

; TEST 67

023426 012737 000067 001226 TST67:
023434 012737 000003 001222
023442 012737 023624 001216

MOV #67,TSTNO
MOV #3,ICOUNT
MOV #TST70,NEXT

023450 104412
023452 005037 023622
023456 005000
023460 012702 036566
023464

MSTCLR
CLR 5\$
CLR R0
MOV #CORMAX,R2

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;START FLAG AT 0
;DATA
;ADDRESS

023464 010037 023514
023470 010237 023520
023474 032702 000001
023500 001402
023502 000337 023514
023506 004537 034664
023512 000000
023514 000000
023516 000000
023520 000000
023522 105012
023524 012761 000221 000004
023532 104414
023534 121110
023536 000240
023540 010005
023542 111204
023544 120504
023546 001401
023550 104021
023552 104401
023554 005200
023556 042700 177400
023562 005737 023622
023566 001402
023570 005700
023572 001412
023574 005202
023576 023702 001304
023602 001330
023604 012702 036566

1\$:

MOV R0,2\$
MOV R2,4\$
BIT #BIT0,R2
BEQ .+6
SWAB 2\$
JSR R5,NPRSET

;LOAD DATA
;LOAD BA
;IS BA ODD?
;BR IF NO
;IF ODD PUT DATA IN HI-BYTE
;LOAD NPR REGISTERS

2\$:

0
0
0
0
CLR8 (R2)
MOV #221,4(R1)
ROMCLK
121110
NOP

;IN DATA
;OUT DATA
;IN BA
;OUT BA
;CLEAR MEMORY LOCATION
;LOAD PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;DO THE NPR

4\$:

MOV R0,R5
MOVB (R2),R4
CMPB R5,R4
BEQ 3\$
EMT 21
SCOP1
INC R0
BIC #177400,R0
TST 5\$
BEQ 6\$
TST R0
BEQ 7\$
INC R2
CMP MEMLIM,R2
BNE 1\$
MOV #CORMAX,R2

;PUT "EXPECTED" IN R5
;PUT "FOUND" IN R4
;IS DATA CORRECT?
;BR IF YES
;ERROR, DATA INCORRECT

3\$:

;NEXT CHARACTER
;USE ONLY LOW BYTE
;HAS MAX MEMORY BEEN REACHED YET?

6\$:

;BR IF NO
;DONE PATTERN?
;BR IF YES
;INC BA
;REACHED MEMORY LIMIT YET?
;BR IF NOT
;RESTART BA AT FIRST ADDRESS

023610 012737 177777 023622
023616 000722
023620 104400
023622 000000

MOV # -1.5\$
BR 1\$
SCOPE
0

;SET FLAG TO END TEST AT END OF DATA PATTERN
;CONTINUE
;SCOPE THIS TEST
;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

65

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

; TEST 70

023624 012737 000070 001226 TST70:
023632 012737 023752 001216
023640 012737 023656 001220

MOV #70,TSTNO
MOV #TST71,NEXT
MOV #65\$,LOCK

023646 104412
023650 005002
023652 012700 000001 1\$:
023656 042737 000377 023672 65\$:
023664 050237 023672
023670 104414
023672 010000 66\$:
023674 010061 000004
023700 104414
023702 122500
023704 104414
023706 040620
023710 104414
023712 061225
023714 010005
023716 116104 000005
023722 120504
023724 001401
023726 104013
023730 104401 67\$:
023732 000241
023734 106100
023736 001347
023740 005202
023742 022702 000400
023746 001341
023750 104400

MSTCLR
CLR R2
MOV #1,R0
BIC #377,66\$
BIS R2,66\$
ROMCLK
010000
MOV R0,4(R1)
ROMCLK
122500
ROMCLK
040620
ROMCLK
61225
MOV R0,R5
MOVB 5(R1),R4
CMPB R5,R4
BEQ 67\$
EOT 13
SCOP1
CLC
ROLB R0
BNE 65\$
INC R2
CMP #400,R2
BNE 1\$
SCOPE

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;START WITH ADDRESS 0
;START WITH BIT 0
;CLEAR ADDRESS FIELD OF INSTRUCTION
;ADD ADDRESS TO INSTRUCTION
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD MAR WITH ADDRESS IN R2
;WRITE PATTERN IN PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOVE PORT4 TO MEMORY
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOVE MEMORY 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 R0
;DONE IF R0=0
;NEXT ADDRESS
;LAST ADDRESS
;BR IF NO
;SCOPE THIS TEST

66

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

; TEST 71

023752 012737 000071 001226 TST71:
023760 012737 024104 001216

MOV #71,TSTNO
MOV #TST72,NEXT

023766 012737 024006 001220 MOV #65\$,LOCK
023774 104412 MSTCLR
023776 005002 CLR R2
024000 012700 000001 1\$: MOV #1,R0
024004 005100 64\$: COM R0
024006 042737 000377 024022 65\$: BIC #377,66\$
024014 050237 024022 BIS R2,66\$
024020 104414 ROMCLK
024022 010000 66\$: 010000
024024 010061 000004 MOV R0,4(R1)
024030 104414 ROMCLK
024032 122500 122500
024034 104414 ROMCLK
024036 040620 040620
024040 104414 ROMCLK
024042 061225 61225
024044 010005 MOV R0,R5
024046 116104 000005 MOV 5(R1),R4
024052 120504 CMPB R5,R4
024054 001401 BEQ 67\$
024056 104013 EMT 13
024060 104401 67\$: SCOP1
024062 005100 COM R0
024064 000241 CLC
024066 106100 ROLB R0
024070 001345 BNE 64\$
024072 005202 INC R2
024074 022702 000400 CMP #400,R2
024100 001337 BNE 1\$
024102 104400 SCOPE
;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;START WITH ADDRESS 0
;START WITH BIT 0
;CHANGE TO FLOATING 0
;CLEAR ADDRESS FIELD OF INSTRUCTION
;ADD ADDRESS TO INSTRUCTION
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD MAR WITH ADDRESS IN R2
;WRITE PATTERN IN PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOVE PORT4 TO MEMORY
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOVE MEMORY 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?
;CHANGE TO FLOATING 1
;CLEAR CARRY
;SHIFT BIT IN R0
;DONE IF R0=0
;NEXT ADDRESS
;LAST ADDRESS
;BR IF NO
;SCOPE THIS TEST

67

***** TEST 72 *****
;*MAIN MEMORY DUAL ADDRESSING TEST
;*LOAD EACH MEMORY LOCATION WITH ITS OWN ADDRESS
;*READ BACK EACH LOCATION TO VERIFY CORRECT ADDRESSING
;*****

: TEST 72

024104 012737 000072 001226 TST72: MOV #72,TSTNO
024112 012737 024304 001216 MOV #TST73,NEXT
024120 012737 024132 001220 MOV #1\$,LOCK
024126 104412 MSTCLR
024130 005002 CLR R2
024132 042737 000377 024146 1\$: BIC #377,2\$
024140 050237 024146 BIS R2,2\$
024144 104414 ROMCLK
024146 010000 2\$: 010000
024150 010261 000004 MOV R2,4(R1)
024154 104414 ROMCLK
024156 122500 122500
024160 104414 ROMCLK
024162 040620 040620
024164 104414 ROMCLK
;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;START AT ADDRESS 0
;CLEAR ADDRESS FIELD OF INSTRUCTION
;ADD ADDRESS TO INSTRUCTION
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD MAR
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOVE PORT4 TO MEMORY
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOVE MEMORY TO THE BR
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

024166 061225 61225 ;MOV BR TO PORT5
024170 010205 MOV R2,R5 ;PUT "EXPECTED" IN R5
024172 116104 000005 MOV 5(R1),R4 ;PUT "FOUND" IN R4
024176 120504 CMPB R5,R4 ;DATA CORRECT?
024200 001401 BEQ 3\$;BR IF YES
024202 104013 EMT 13 ;DATA ERROR
024204 104401 3\$: SCOP1 ;SW09=1?
024206 005202 INC R2 ;NEXT ADDRESS
024210 022702 000400 CMP #400,R2 ;LAST ADDRESS
024214 001346 BNE 1\$;BR IF NO
024216 012737 024226 001220 MOV #4\$,LOCK ;NEW SCOPE 1
024224 005002 CLR R2 ;RESTART AT ADDRESS 0
024226 042737 000377 024242 4\$: BIC #377,5\$;CLEAR ADDRESS FIELD OF INSTRUCTION
024234 050237 024242 BIS R2,5\$;ADD ADDRESS TO INSTRUCTION
024240 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024242 010000 5\$: 010000 ;LOAD THE MAR
024244 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024246 040620 040620 ;MOVE MEMORY TO THE BR
024250 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024252 061225 61225 ;MOV BR TO PORT5
024254 010205 MOV R2,R5 ;PUT "EXPECTED" IN R5
024256 116104 000005 MOV 5(R1),R4 ;PUT "FOUND" IN R4
024262 120504 CMPB R5,R4 ;DATA CORRECT?
024264 001401 BEQ 6\$;BR IF YES
024266 104013 EMT 13 ;ADDRESSING ERROR
024270 104401 6\$: SCOP1 ;SW09=1?
024272 005202 INC R2 ;NEXT ADDRESS
024274 022702 000400 CMP #400,R2 ;IS IT THE LAST
024300 001352 BNE 4\$;BR IF NO
024302 104400 SCOPE ;SCOPE THIS TEST

68

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

; TEST 73

024304 012737 000073 001226 TST73: MOV #73,TSTNO
024312 012737 024440 001216 MOV #TST74,NEXT
024320 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
024322 005002 CLR R2 ;MASTER CLEAR DMC11
024324 104414 ROMCLK ;START WITH A ZERO
024326 010000 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024330 032737 100000 001366 BIT #BIT15,STAT1 ;LOAD MAR
024336 001402 BEQ .+6 ;DMC?
024340 104414 ROMCLK ;BR IF YES
024342 004000 4000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024344 010261 000004 1\$: MOV R2,4(R1) ;MAR HI = 0 (KMC ONLY)
024350 104414 ROMCLK ;WRITE DATA TO PORT4
024352 136500 136500 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024354 005202 INC R2 ;LOAD MEM AUTO-INC MAR
024356 022702 000400 CMP #400,R2 ;INCREMENT DATA
024362 001370 BNE 1\$;DONE YET?
;BR IF NO

72

024364	005002			CLR	R2	;RESTART WITH A ZERO
024366	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024370	010000			010000		;LOAD MAR
024372	032737	100000	001366	BIT	#BIT15,STAT1	;DMC?
024400	001402			BEQ	.+6	;BR IF YES
024402	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024404	004000			4000		;MAR HI _ 0 (KMC ONLY)
024406						
024406	104414		2\$:	ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024410	055224			055224		;MOVE MEM TO PORT4
024412	010205			MOV	R2,R5	;PUT "EXPECTED" IN R5
024414	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
024420	120504			CMPB	R5,R4	;DATA CORRECT?
024422	001401			BEQ	3\$	;BR IF YES
024424	104014			EMT	14	;MAR ERROR
024426	005202			INC	R2	;NEXT ADDRESS
024430	022702	000400	3\$:	CMP	#400,R2	;DONE YET?
024434	001364			BNE	2\$	;BR IF NO
024436	104400			SCOPE		;SCOPE THIS TEST

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

; TEST 74
;-----
024440 012737 000074 001226 TST74: MOV #74,TSTNO
024446 012737 024552 001216 MOV #TST75,NEXT
024454 012737 024500 001220 MOV #1\$,LOCK

024462 104412 MSTCLR
024464 004737 034726 JSR PC,MEMLD
024470 024542 TDATA
024472 004737 034762 JSR PC,SPLD
024476 024542 TDATA

024500 1\$:
024500 104414 ROMCLK
024502 010000 010000
024504 104414 ROMCLK
024506 054400 054400!<0*20>
024510 104414 ROMCLK
024512 040421 040401!<1*20>
024514 104414 ROMCLK
024516 061224 61224
024520 012705 000001 MOV #1,R5
024524 016104 000004 MOV 4(R1),R4
024530 120504 CMPB R5,R4
024532 001401 BEQ 2\$
024534 104015 EMT 15
024536 104401 2\$:
024540 104400 SCOP1
024542 377 000 TDATA: .BYTE -1,0,0,0,0,0,0,0
024545 000 000
024550 000 000

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;LOAD MAINMEM DATA
;POINTER TO DATA
;LOAD SP DATA
;POINTER TO DATA

;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MAR_0
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;ADD 377 AND 377, TO SET C BIT
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;ADD 0 AND 0 AND THE C BIT
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PUT RESULTS IN PORT4
;PUT "EXPECTED" IN R5
;PUT "FOUND" IN R4
;DATA CORRECT?
;BR IF YES
;ERROR C BIT NOT SET
;SW09=1?
;SCOPE THIS TEST

.EVEN

73

```
***** 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
;PERFORM THE FUNCTION, VERIFY THE RESULTS
;*****
```

; TEST 75

```
024552 012737 000075 001226 TST75: MOV #75,TSTNO
024560 012737 024726 001216 MOV #TST76,NEXT
024566 012737 024620 001220 MOV #1$,LOCK

024574 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
024576 005000 CLR R0 ;MASTER CLEAR DMC11
024600 012702 024716 MOV #5$,R2 ;MEM + SP ADDRESS
024604 004737 034726 JSR PC,MEMLD ;POINTER TO CORRECT DATA
024610 035052 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
024612 004737 034762 JSR PC,SPLD ;POINTER TO DATA
024616 035062 SPDAT ;LOAD 8 WORDS OF SP
024620 004737 035026 1$: JSR PC,CLRC ;POINTER TO DATA
024624 042737 000017 024640 BIC #17,2$ ;CLEAR C BIT!
024632 050037 024640 BIS R0,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
024636 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
024640 010000 2$: 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024642 042737 000017 024656 BIC #17,3$ ;LOAD MAR
024650 050037 024656 BIS R0,3$ ;CLEAR ADDRESS OF INSTRUCTION
024654 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
024656 040620 3$: 040400!<11*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024660 104414 ROMCLK ;BR - SEL B
024662 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
024664 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
024666 116104 000004 MOVB 4(R1),R4 ;PUT "EXPECTED" IN R5
024672 120504 CMPB R5,R4 ;PUT "FOUND" IN R4
024674 001401 BEQ 4$ ;DATA CORRECT?
024676 104015 EMT 15 ;BR IF YES
024700 104401 4$: SCOP1 ;ALU ERROR
024702 005202 INC R2 ;SW09=1?
024704 005200 INC R0 ;NEXT DATA
024706 022700 000010 CMP #10,R0 ;NEXT ADDRESS
024712 001342 BNE 1$ ;DONE YET?
024714 104400 SCOPE ;BR IF NO
024716 000 377 000 5$: .BYTE 0,-1,0,-1,125,252,125,252 ;SCOPE THIS TEST
024721 377 125 252
024724 125 252

.EVEN
```

74

```
***** TEST 76 *****
;ALU TEST
;TEST OF ALU FUNCTION SEL A WITH C BIT CLEARED
;ALU FUNCTION (A) CODE=10
;LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
;PERFORM THE FUNCTION, VERIFY THE RESULTS
```

;*****

; TEST 76

024726	012737	000076	001226	TST76:	MOV	#76,TSTNO	
024734	012737	025102	001216		MOV	#TST77,NEXT	
024742	012737	024774	001220		MOV	#1\$,LOCK	
024750	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
024752	005000				CLR	R0	;MASTER CLEAR DMC11
024754	012702	025072			MOV	#5\$,R2	;MEM + SP ADDRESS
024760	004737	034726			JSR	PC,MEMLD	;POINTER TO CORRECT DATA
024764	035052				MEMDAT		;LOAD 8 WORDS OF MAIN MEMORY
024766	004737	034762			JSR	PC,SPLD	;POINTER TO DATA
024772	035062				SPDAT		;LOAD 8 WORDS OF SP
024774	004737	035026			JSR	PC,CLRC	;POINTER TO DATA
025000	042737	000017	025014	1\$:	BIC	#17,2\$	;CLEAR C BIT!
025006	050037	025014			BIS	R0,2\$	;CLEAR ADDRESS FIELD OF INSTRUCTION
025012	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
025014	010000			2\$:	010000		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025016	042737	000017	025032		BIC	#17,3\$	;LOAD MAR
025024	050037	025032			BIS	R0,3\$	;CLEAR ADDRESS OF INSTRUCTION
025030	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
025032	040600			3\$:	040400!<10*20>		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025034	104414				ROMCLK		;BR SEL A
025036	061224				61224		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025040	111205				MOVB	(R2),R5	;MOVE BR TO PORT4
025042	116104	000004			MOVB	4(R1),R4	;PUT "EXPECTED" IN R5
025046	120504				CMPB	R5,R4	;PUT "FOUND" IN R4
025050	001401				BEQ	4\$	;DATA CORRECT?
025052	104015				EMT	15	;BR IF YES
025054	104401			4\$:	SCOP1		;ALU ERROR
025056	005202				INC	R2	;SW09=1?
025060	005200				INC	R0	;NEXT DATA
025062	022700	000010			CMP	#10,R0	;NEXT ADDRESS
025066	001342				BNE	1\$	;DONE YET?
025070	104400				SCOPE		;BR IF NO
025072	000	000	377	5\$:	.BYTE	0,0,-1,-1,125,125,252,252	;SCOPE THIS TEST
025075	377	125	125				
025100	252	252					

.EVEN

75

;***** TEST 77 *****

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

;*****

; TEST 77

025102	012737	000077	001226	TST77:	MOV	#77,TSTNO
025110	012737	025256	001216		MOV	#TST100,NEXT
025116	012737	025150	001220		MOV	#1\$,LOCK

;R1 CONTAINS BASE DMC11 ADDRESS

025124	104412				MSTCLR				;MASTER CLEAR DMC11
025126	005000				CLR	R0			;MEM + SP ADDRESS
025130	012702	025246			MOV	#5,R2			;POINTER TO CORRECT DATA
025134	004737	034726			JSR	PC,MEMLD			;LOAD 8 WORDS OF MAIN MEMORY
025140	035052				MEMDAT				;POINTER TO DATA
025142	004737	034762			JSR	PC,SPLD			;LOAD 8 WORDS OF SP
025146	035062				SPDAT				;POINTER TO DATA
025150	004737	035026		1#:	JSR	PC,CLRC			;CLEAR C BIT!
025154	042737	000017	025170		BIC	#17,2#			;CLEAR ADDRESS FIELD OF INSTRUCTION
025162	050037	025170			BIS	R0,2#			;ADD ADDRESS TO INSTRUCTION
025166	104414				ROMCLK				;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025170	010000			2#:	010000				;LOAD MAR
025172	042737	000017	025206		BIC	#17,3#			;CLEAR ADDRESS OF INSTRUCTION
025200	050037	025206			BIS	R0,3#			;ADD ADDRESS TO INSTRUCTION
025204	104414				ROMCLK				;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025206	040640			3#:	040400!<12*20>				;BR A OR NOTB
025210	104414				ROMCLK				;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025212	061224				61224				;MOVE BR TO PORT4
025214	111205				MOVB	(R2),R5			;PUT "EXPECTED" IN R5
025216	116104	000004			MOVB	4(R1),R4			;PUT "FOUND" IN R4
025222	120504				CMPB	R5,R4			;DATA CORRECT?
025224	001401				BEQ	4#			;BR IF YES
025226	104015				EMT	15			;ALU ERROR
025230	104401			4#:	SCOP1				;SW09=1?
025232	005202				INC	R2			;NEXT DATA
025234	005200				INC	R0			;NEXT ADDRESS
025236	022700	000010			CMP	#10,R0			;DONE YET?
025242	001342				BNE	1#			;BR IF NO
025244	104400				SCOPE				;SCOPE THIS TEST
025246	377	000	377	5#:	.BYTE	-1,0,-1,-1,-1,125,252,-1			
025251	377	377	125						
025254	252	377							

.EVEN

76

***** 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

025256	012737	000100	001226	TST100:	MOV	#100,TSTNO			
025264	012737	025432	001216		MOV	#TST101,NEXT			
025272	012737	025324	001220		MOV	#1#,LOCK			
025300	104412				MSTCLR				;R1 CONTAINS BASE DMC11 ADDRESS
025302	005000				CLR	R0			;MASTER CLEAR DMC11
025304	012702	025422			MOV	#5,R2			;MEM + SP ADDRESS
025310	004737	034726			JSR	PC,MEMLD			;POINTER TO CORRECT DATA
025314	035052				MEMDAT				;LOAD 8 WORDS OF MAIN MEMORY
025316	004737	034762			JSR	PC,SPLD			;POINTER TO DATA
025322	035062				SPDAT				;LOAD 8 WORDS OF SP
025324	004737	035026		1#:	JSR	PC,CLRC			;POINTER TO DATA

;CLEAR C BIT!

.EVEN

```

;***** TEST 101 *****
;ALU TEST
;TEST OF ALU FUNCTION A OR B WITH C BIT CLEARED
;ALU FUNCTION (A OR B)      CODE=14
;LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
;PERFORM THE FUNCTION, VERIFY THE RESULTS
;:*****

```

: TEST 101

025432	012737	000101	001226	TST101:	MOV	#101,TSTNO			
025440	012737	025606	001216		MOV	#TST102,NEXT			
025446	012737	025500	001220		MOV	#1\$,LOCK			
									:R1 CONTAINS BASE DMC11 ADDRESS
025454	104412				MSTCLR				:MASTER CLEAR DMC11
025456	005000				CLR	R0			:MEM + SP ADDRESS
025460	012702	025576			MOV	#5\$,R2			:POINTER TO CORRECT DATA
025464	004737	034726			JSR	PC,MEMLD			:LOAD 8 WORDS OF MAIN MEMORY
025470	035052				MEMDAT				:POINTER TO DATA
025472	004737	034762			JSR	PC,SPLD			:LOAD 8 WORDS OF SP
025476	035062				SPDAT				:POINTER TO DATA
025500	004737	035026		1\$:	JSR	PC,CLRC			:CLEAR C BIT!
025504	042737	000017	025520		BIC	#17,2\$			:CLEAR ADDRESS FIELD OF INSTRUCTION
025512	050037	025520			BIS	R0,2\$			:ADD ADDRESS TO INSTRUCTION
025516	104414				ROMCLK				:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025520	010000			2\$:	010000				:LOAD MAR
025522	042737	000017	025536		BIC	#17,3\$			:CLEAR ADDRESS OF INSTRUCTION
025530	050037	025536			BIS	R0,3\$			:ADD ADDRESS TO INSTRUCTION
025534	104414				ROMCLK				:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025536	040700			3\$:	040400!<14*20>				:BR A OR B

025540	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025542	061224				61224		;MOVE BR TO PORT4
025544	111205				MOVB	(R2),R5	;PUT "EXPECTED" IN R5
025546	116104	000004			MOVB	4(R1),R4	;PUT "FOUND" IN R4
025552	120504				CMPB	R5,R4	;DATA CORRECT?
025554	001401				BEQ	4\$	;BR IF YES
025556	104015				EMT	15	;ALU ERROR
025560	104401		4\$:		SCOP1		;SW09=1?
025562	005202				INC	R2	;NEXT DATA
025564	005200				INC	R0	;NEXT ADDRESS
025566	022700	000010			CMP	#10,R0	;DONE YET?
025572	001342				BNE	1\$	;BR IF NO
025574	104400				SCOPE		;SCOPE THIS TEST
025576	000	377	377	5\$:	.BYTE	0,-1,-1,-1,125,-1,-1,252	
025601	377	125	377				
025604	377	252					

.EVEN

78

***** TEST 102 *****
;ALU TEST
;TEST OF ALU FUNCTION A XOR B WITH C BIT CLEARED
;ALU FUNCTION (A XOR B) CODE=15
;LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
;PERFORM THE FUNCTION, VERIFY THE RESULTS
:*****

; TEST 102

025606	012737	000102	001226	TST102:	MOV	#102,TSTNO	
025614	012737	025762	001216		MOV	#TST103,NEXT	
025622	012737	025654	001220		MOV	#1\$,LOCK	

025630	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
025632	005000				CLR	R0	;MASTER CLEAR DMC11
025634	012702	025752			MOV	#5\$,R2	;MEM + SP ADDRESS
025640	004737	034726			JSR	PC,MEMLD	;POINTER TO CORRECT DATA
025644	035052				MEMDAT		;LOAD 8 WORDS OF MAIN MEMORY
025646	004737	034762			JSR	PC,SPLD	;POINTER TO DATA
025652	035062				SPDAT		;LOAD 8 WORDS OF SP
025654	004737	035026		1\$:	JSR	PC,CLRC	;POINTER TO DATA
025660	042737	000017	025674		BIC	#17,2\$	;CLEAR C BIT!
025666	050037	025674			BIS	R0,2\$	;CLEAR ADDRESS FIELD OF INSTRUCTION
025672	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
025674	010000			2\$:	010000		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025676	042737	000017	025712		BIC	#17,3\$	;LOAD MAR
025704	050037	025712			BIS	R0,3\$	;CLEAR ADDRESS OF INSTRUCTION
025710	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
025712	040720			3\$:	040400!<15*20>		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025714	104414				ROMCLK		;BR A XOR B
025716	061224				61224		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025720	111205				MOVB	(R2),R5	;MOVE BR TO PORT4
025722	116104	000004			MOVB	4(R1),R4	;PUT "EXPECTED" IN R5
025726	120504				CMPB	R5,R4	;PUT "FOUND" IN R4
025730	001401				BEQ	4\$	;DATA CORRECT?
025732	104015				EMT	15	;BR IF YES
025734	104401			4\$:	SCOP1		;ALU ERROR

;SW09=1?

025736	005202				INC	R2		;NEXT DATA
025740	005200				INC	R0		;NEXT ADDRESS
025742	022700	000010			CMP	#10,R0		;DONE YET?
025746	001342				BNE	1\$		;BR IF NO
025750	104400				SCOPE			;SCOPE THIS TEST
025752	000	377	377	5\$:	.BYTE	0,-1,-1,0,0,-1,-1,0		
025755	000	000	377					
025760	377	000						

.EVEN

79

```
***** 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

025762	012737	000103	001226	TST103:	MOV	#103,TSTNO		
025770	012737	026136	001216		MOV	#TST104,NEXT		
025776	012737	026030	001220		MOV	#1\$,LOCK		
;R1 CONTAINS BASE DMC11 ADDRESS								
026004	104412				MSTCLR			;MASTER CLEAR DMC11
026006	005000				CLR	R0		;MEM + SP ADDRESS
026010	012702	026126			MOV	#5\$,R2		;POINTER TO CORRECT DATA
026014	004737	034726			JSR	PC,MEMLD		;LOAD 8 WORDS OF MAIN MEMORY
026020	035052				MEMDAT			;POINTER TO DATA
026022	004737	034762			JSR	PC,SPLD		;LOAD 8 WORDS OF SP
026026	035062				SPDAT			;POINTER TO DATA
026030	004737	035026		1\$:	JSR	PC,CLRC		;CLEAR C BIT!
026034	042737	000017	026050		BIC	#17,2\$		;CLEAR ADDRESS FIELD OF INSTRUCTION
026042	050037	026050			BIS	R0,2\$		;ADD ADDRESS TO INSTRUCTION
026046	104414				ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026050	010000			2\$:	010000			;LOAD MAR
026052	042737	000017	026066		BIC	#17,3\$		;CLEAR ADDRESS OF INSTRUCTION
026060	050037	026066			BIS	R0,3\$		;ADD ADDRESS TO INSTRUCTION
026064	104414				ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026066	040400			3\$:	040400!<00*20>			;BR ADD
026070	104414				ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026072	061224				61224			;MOVE BR TO PORT4
026074	111205				MOVB	(R2),R5		;PUT "EXPECTED" IN R5
026076	116104	000004			MOVB	4(R1),R4		;PUT "FOUND" IN R4
026102	120504				CMPB	R5,R4		;DATA CORRECT?
026104	001401				BEQ	4\$		;BR IF YES
026106	104015				EMT	15		;ALU ERROR
026110	104401			4\$:	SCOP1			;SW09=1?
026112	005202				INC	R2		;NEXT DATA
026114	005200				INC	R0		;NEXT ADDRESS
026116	022700	000010			CMP	#10,R0		;DONE YET?
026122	001342				BNE	1\$		;BR IF NO
026124	104400				SCOPE			;SCOPE THIS TEST
026126	000	377	377	5\$:	.BYTE	0,-1,-1,376,252,-1,-1,124		
026131	376	252	377					
026134	377	124						

80

.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

```
026136 012737 000104 001226 TST104: MOV #104,TSTNO
026144 012737 026312 001216 MOV #TST105,NEXT
026152 012737 026204 001220 MOV #1$,LOCK

026160 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
026162 005000 CLR R0 ;MASTER CLEAR DMC11
026164 012702 026302 MOV #5$,R2 ;MEM + SP ADDRESS
026170 004737 034726 JSR PC,MEMLD ;POINTER TO CORRECT DATA
026174 035052 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
026176 004737 034762 JSR PC,SPLD ;POINTER TO DATA
026202 035062 SPDAT ;LOAD 8 WORDS OF SP
026204 004737 035026 1$: JSR PC,CLRC ;POINTER TO DATA
026210 042737 000017 026224 BIC #17,2$ ;CLEAR C BIT!
026216 050037 026224 BIS R0,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
026222 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
026224 010000 2$: 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026226 042737 000017 026242 BIC #17,3$ ;LOAD MAR
026234 050037 026242 BIS R0,3$ ;CLEAR ADDRESS OF INSTRUCTION
026240 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
026242 040540 3$: 040400!<6*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026244 104414 ROMCLK ;BR 2A W/C
026246 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026250 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
026252 116104 000004 MOVB 4(R1),R4 ;PUT "EXPECTED" IN R5
026256 120504 CMPB R5,R4 ;PUT "FOUND" IN R4
026260 001401 BEQ 4$ ;DATA CORRECT?
026262 104015 EMT 15 ;BR IF YES
026264 104401 4$: SCOP1 ;ALU ERROR
026266 005202 INC R2 ;SW09=1?
026270 005200 INC R0 ;NEXT DATA
026272 022700 000010 CMP #10,R0 ;NEXT ADDRESS
026276 001342 BNE 1$ ;DONE YET?
026300 104400 SCOPE ;BR IF NO
026302 000 000 376 5$: .BYTE 0,0,376,376,252,252,124,124 ;SCOPE THIS TEST
026305 376 252 252
026310 124 124
```

.EVEN

81

```
***** 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

```
026312 012737 000105 001226 TST105: MOV #105,TSTNO
026320 012737 026466 001216 MOV #TST106,NEXT
026326 012737 026360 001220 MOV #1$,LOCK

026334 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
026336 005000 CLR R0 ;MASTER CLEAR DMC11
026340 012702 026456 MOV #5$,R2 ;MEM + SP ADDRESS
026344 004737 034726 JSR PC,MEMLD ;POINTER TO CORRECT DATA
026350 035052 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
026352 004737 034762 JSR PC,SPLD ;POINTER TO DATA
026356 035062 SPDAT ;LOAD 8 WORDS OF SP
026360 004737 035026 1$: JSR PC,CLRC ;POINTER TO DATA
026364 042737 000017 026400 BIC #17,2$ ;CLEAR C BIT!
026372 050037 026400 BIS R0,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
026376 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
026400 010000 2$: 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026402 042737 000017 026416 BIC #17,3$ ;LOAD MAR
026410 050037 026416 BIS R0,3$ ;CLEAR ADDRESS OF INSTRUCTION
026414 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
026416 040740 3$: 040400!<16*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026420 104414 ROMCLK ;BR SUB
026422 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026424 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
026426 116104 000004 MOVB 4(R1),R4 ;PUT "EXPECTED" IN R5
026432 120504 CMPB R5,R4 ;PUT "FOUND" IN R4
026434 001401 BEQ 4$ ;DATA CORRECT?
026436 104015 EMT 15 ;BR IF YES
026440 104401 4$: SCOP1 ;ALU ERROR
026442 005202 INC R2 ;SW09=1?
026444 005200 INC R0 ;NEXT DATA
026446 022700 000010 CMP #10,R0 ;NEXT ADDRESS
026452 001342 BNE 1$ ;DONE YET?
026454 104400 SCOPE 1$ ;BR IF NO
026456 000 001 377 5$: .BYTE 0,1,-1,0,0,253,125,0
026461 000 000 253
026464 125 000

.EVEN
```

82

;***** 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

```
026466 012737 000106 001226 TST106: MOV #106,TSTNO
026474 012737 026642 001216 MOV #TST107,NEXT
026502 012737 026534 001220 MOV #1$,LOCK
```


83

; TEST 107

[illegible]

026710	004737	035026		1\$:	JSR	PC,CLRC	;CLEAR C BIT!
026714	042737	000017	026730		BIC	#17,2\$	;CLEAR ADDRESS FIELD OF INSTRUCTION
026722	050037	026730			BIS	R0,2\$	;ADD ADDRESS TO INSTRUCTION
026726	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026730	010000			2\$:	010000		;LOAD MAR
026732	042737	000017	026746		BIC	#17,3\$	;CLEAR ADDRESS OF INSTRUCTION
026740	050037	026746			BIS	R0,3\$	;ADD ADDRESS TO INSTRUCTION
026744	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026746	040440			3\$:	040400!<2*20>		;BR SUB W/C
026750	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026752	061224				61224		;MOVE BR TO PORT4
026754	111205				MOVB	(R2),R5	;PUT "EXPECTED" IN R5
026756	116104	000004			MOVB	4(R1),R4	;PUT "FOUND" IN R4
026762	120504				CMPB	R5,R4	;DATA CORRECT?
026764	001401				BEQ	4\$	;BR IF YES
026766	104015				EMT	15	;ALU ERROR
026770	104401			4\$:	SCOP1		;SW09=1?
026772	005202				INC	R2	;NEXT DATA
026774	005200				INC	R0	;NEXT ADDRESS
026776	022700	000010			CMP	#10,R0	;DONE YET?
027002	001342				BNE	1\$	;BR IF NO
027004	104400				SCOPE		;SCOPE THIS TEST
027006	377	000	376	5\$:	.BYTE	-1,0,376,-1,-1,252,124,-1	
027011	377	377	252				
027014	124	377					

.EVEN

84

***** TEST 110 *****
;ALU TEST
;TEST OF ALU FUNCTION INC A WITH C BIT CLEARED
;ALU FUNCTION (A PLUS 1) CODE=3
;LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
;PERFORM THE FUNCTION, VERIFY THE RESULTS
;*****

; TEST 110

027016	012737	000110	001226	TST110:	MOV	#110,TSTNO	
027024	012737	027172	001216		MOV	#TST111,NEXT	
027032	012737	027064	001220		MOV	#1\$,LOCK	
027040	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
027042	005000				CLR	R0	;MASTER CLEAR DMC11
027044	012702	027162			MOV	#5\$,R2	;MEM + SP ADDRESS
027050	004737	034726			JSR	PC,MEMLD	;POINTER TO CORRECT DATA
027054	035052				MEMDAT		;LOAD 8 WORDS OF MAIN MEMORY
027056	004737	034762			JSR	PC,SPLD	;POINTER TO DATA
027062	035062				SPDAT		;LOAD 8 WORDS OF SP
027064	004737	035026			JSR	PC,CLRC	;POINTER TO DATA
027070	042737	000017	027104	1\$:	BIC	#17,2\$	;CLEAR C BIT!
027076	050037	027104			BIS	R0,2\$	;CLEAR ADDRESS FIELD OF INSTRUCTION
027102	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
027104	010000			2\$:	010000		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027106	042737	000017	027122		BIC	#17,3\$	;LOAD MAR
027114	050037	027122			BIS	R0,3\$	;CLEAR ADDRESS OF INSTRUCTION
027120	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
							;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

027122	040460			3\$:	040400!<3*20>		;BR INC A
027124	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027126	061224				61224		;MOVE BR TO PORT4
027130	111205				MOVB (R2),R5		;PUT "EXPECTED" IN R5
027132	116104	000004			MOVB 4(R1),R4		;PUT "FOUND" IN R4
027136	120504				CMPB R5,R4		;DATA CORRECT?
027140	001401				BEQ 4\$		;BR IF YES
027142	104015				EMT 15		;ALU ERROR
027144	104401			4\$:	SCOP1		;SW09=1?
027146	005202				INC R2		;NEXT DATA
027150	005200				INC R0		;NEXT ADDRESS
027152	022700	000010			CMP #10,R0		;DONE YET?
027156	001342				BNE 1\$		;BR IF NO
027160	104400				SCOPE		;SCOPE THIS TEST
027162	001	001	000	5\$:	.BYTE 1,1,0,0,126,126,253,253		
027165	000	126	126				
027170	253	253					

.EVEN

85

***** TEST 111 *****
;ALU TEST
;TEST OF ALU FUNCTION 2A WITH C BIT CLEARED
;ALU FUNCTION (A PLUS A) CODE=5
;LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
;PERFORM THE FUNCTION, VERIFY THE RESULTS

; TEST 111

027172	012737	000111	001226	TST111:	MOV #111,TSTNO		
027200	012737	027346	001216		MOV #TST112,NEXT		
027206	012737	027240	001220		MOV #1\$,LOCK		
027214	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
027216	005000				CLR R0		;MASTER CLEAR DMC11
027220	012702	027336			MOV #5\$,R2		;MEM + SP ADDRESS
027224	004737	034726			JSR PC,MEMLD		;POINTER TO CORRECT DATA
027230	035052				MEMDAT		;LOAD 8 WORDS OF MAIN MEMORY
027232	004737	034762			JSR PC,SPLD		;POINTER TO DATA
027236	035062				SPDAT		;LOAD 8 WORDS OF SP
027240	004737	035026			JSR PC,CLRC		;POINTER TO DATA
027244	042737	000017	027260	1\$:	BIC #17,2\$		;CLEAR C BIT!
027252	050037	027260			BIS R0,2\$		;CLEAR ADDRESS FIELD OF INSTRUCTION
027256	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
027260	010000				010000		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027262	042737	000017	027276	2\$:	BIC #17,3\$		;LOAD MAR
027270	050037	027276			BIS R0,3\$		;CLEAR ADDRESS OF INSTRUCTION
027274	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
027276	040520			3\$:	040400!<5*20>		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027300	104414				ROMCLK		;BR 2A
027302	061224				61224		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027304	111205				MOVB (R2),R5		;MOVE BR TO PORT4
027306	116104	000004			MOVB 4(R1),R4		;PUT "EXPECTED" IN R5
027312	120504				CMPB R5,R4		;PUT "FOUND" IN R4
027314	001401				BEQ 4\$		;DATA CORRECT?
027316	104015				EMT 15		;BR IF YES
							;ALU ERROR

4\$:
5\$:
.EVEN

86

```
***** TEST 112 *****
;*ALU TEST
;*TEST OF ALU FUNCTION A PLUS C WITH C BIT CLEARED
;*ALU FUNCTION (A PLUS C)    CODE=4
;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
;*PERFORM THE FUNCTION, VERIFY THE RESULTS
: *****
```

; TEST 112

Address	Hex	Hex	Hex	Label	Operation	Comments
027346	012737	000112	001226	TST112:	MOV	#112,TSTNO
027354	012737	027522	001216		MOV	#TST113,NEXT
027362	012737	027414	001220		MOV	#1\$,LOCK
027370	104412				MSTCLR	;R1 CONTAINS BASE DMC11 ADDRESS
027372	005000				CLR	;MASTER CLEAR DMC11
027374	012702	027512			MOV	;MEM + SP ADDRESS
027400	004737	034726			JSR	;POINTER TO CORRECT DATA
027404	035052				MEMDAT	;LOAD 8 WORDS OF MAIN MEMORY
027406	004737	034762			JSR	;POINTER TO DATA
027412	035062				SPDAT	;LOAD 8 WORDS OF SP
027414	004737	035026		1\$:	JSR	;POINTER TO DATA
027420	042737	000017	027434		BIC	;CLEAR C BIT!
027426	050037	027434			BIS	;CLEAR ADDRESS FIELD OF INSTRUCTION
027432	104414				ROMCLK	;ADD ADDRESS TO INSTRUCTION
027434	010000			2\$:	010000	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027436	042737	000017	027452		BIC	;LOAD MAR
027444	050037	027452			BIS	;CLEAR ADDRESS OF INSTRUCTION
027450	104414				ROMCLK	;ADD ADDRESS TO INSTRUCTION
027452	040500			3\$:	040400!<4*20>	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027454	104414				ROMCLK	;BR - A PLUS C
027456	061224				61224	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027460	111205				MOVB	;MOVE BR TO PORT4
027462	116104	000004			MOVB	;PUT "EXPECTED" IN R5
027466	120504				CMPB	;PUT "FOUND" IN R4
027470	001401				BEQ	;DATA CORRECT?
027472	104015				EMT	;BR IF YES
027474	104401			4\$:	SCOPI	;ALU ERROR
027476	005202				INC	;SW09=1?
027500	005200				INC	;NEXT DATA
027502	022700	000010			CMP	;NEXT ADDRESS
027506	001342				BNE	;DONE YET?
027510	104400				SCOPE	;BR IF NO
027512	000	000	377	5\$:	.BYTE	;SCOPE THIS TEST
027515	377	125	125			0,0,-1,-1,125,125,252,252

027520 252 252

.EVEN

87

```
***** 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

```
027522 012737 000113 001226 TST113: MOV #113,TSTNO
027530 012737 027676 001216 MOV #TST114,NEXT
027536 012737 027570 001220 MOV #1$,LOCK

027544 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
027546 005000 CLR R0 ;MASTER CLEAR DMC11
027550 012702 027666 MOV #5$,R2 ;MEM + SP ADDRESS
027554 004737 034726 JSR PC,MEMLD ;POINTER TO CORRECT DATA
027560 035052 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
027562 004737 034762 JSR PC,SPLD ;POINTER TO DATA
027566 035062 SPDAT ;LOAD 8 WORDS OF SP
027570 004737 035026 1$: JSR PC,CLRC ;POINTER TO DATA
027574 042737 000017 027610 BIC #17,2$ ;CLEAR C BIT!
027602 050037 027610 BIS R0,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
027606 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
027610 010000 2$: 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027612 042737 000017 027626 BIC #17,3$ ;LOAD MAR
027620 050037 027626 BIS R0,3$ ;CLEAR ADDRESS OF INSTRUCTION
027624 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
027626 040760 3$: 040400!<17*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027630 104414 ROMCLK ;BR 2'S COMP SUB
027632 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027634 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
027636 116104 000004 MOVB 4(R1),R4 ;PUT "EXPECTED" IN R5
027642 120504 CMPB R5,R4 ;PUT "FOUND" IN R4
027644 001401 BEQ 4$ ;DATA CORRECT?
027646 104015 EMT 15 ;BR IF YES
027650 104401 4$: SCOP1 ;ALU ERROR
027652 005202 INC R2 ;SW09=1?
027654 005200 INC R0 ;NEXT DATA
027656 022700 000010 CMP #10,R0 ;NEXT ADDRESS
027662 001342 BNE 1$ ;DONE YET?
027664 104400 SCOPE ;BR IF NO
027666 377 000 376 5$: .BYTE -1,0,376,-1,-1,252,124,-1 ;SCOPE THIS TEST
027671 377 377 252
027674 124 377
```

.EVEN

88

```
***** 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

```
027676 012737 000114 001226 TST114: MOV #114,TSTNO
027704 012737 030052 001216 MOV #TST115,NEXT
027712 012737 027744 001220 MOV #1$,LOCK

027720 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
027722 005000 CLR R0 ;MASTER CLEAR DMC11
027724 012702 030042 MOV #5$,R2 ;MEM + SP ADDRESS
027730 004737 034726 JSR PC,MEMLD ;POINTER TO CORRECT DATA
027734 035052 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
027736 004737 034762 JSR PC,SPLD ;POINTER TO DATA
027742 035062 SPDAT ;LOAD 8 WORDS OF SP
027744 004737 035026 1$: JSR PC,CLRC ;POINTER TO DATA
027750 042737 000017 027764 BIC #17,2$ ;CLEAR C BIT!
027756 050037 027764 BIS R0,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
027762 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
027764 010000 2$: ROMCLK 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
027766 042737 000017 030002 BIC #17,3$ ;LOAD MAR
027774 050037 030002 BIS R0,3$ ;CLEAR ADDRESS OF INSTRUCTION
030000 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
030002 040560 3$: ROMCLK 040400!<7*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
030004 104414 ROMCLK ;BR DEC A
030006 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
030010 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
030012 116104 000004 MOVB 4(R1),R4 ;PUT "EXPECTED" IN R5
030016 120504 CMPB R5,R4 ;PUT "FOUND" IN R4
030020 001401 BEQ 4$ ;DATA CORRECT?
030022 104015 EMT 15 ;BR IF YES
030024 104401 4$: SCOP1 ;ALU ERROR
030026 005202 INC R2 ;SW09=1?
030030 005200 INC R0 ;NEXT DATA
030032 022700 000010 CMP #10,R0 ;NEXT ADDRESS
030036 001342 BNE 1$ ;DONE YET?
030040 104400 SCOPE ;BR IF NO
030042 377 377 376 5$: .BYTE -1,-1,376,376,124,124,251,251
030045 376 124 124
030050 251 251
```

.EVEN

89

```
;***** 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

```
030052 012737 000115 001226 TST115: MOV #115,TSTNO
030060 012737 030226 001216 MOV #TST116,NEXT
```



```
030066 012737 030120 001220      MOV      #1$,LOCK
030074 104412                      MSTCLR
030076 005000                      CLR      R0
030100 012702 030216      MOV      #5$,R2
030104 004737 034726      JSR      PC,MEMLD
030110 035052                      MEMDAT
030112 004737 034762      JSR      PC,SPLD
030116 035062                      SPDAT
030120 004737 035040      JSR      PC,SETC
030124 042737 000017 030140      BIC      #17,2$
030132 050037 030140      BIS      R0,2$
030136 104414                      ROMCLK
030140 010000      2$: 010000
030142 042737 000017 030156      BIC      #17,3$
030150 050037 030156      BIS      R0,3$
030154 104414                      ROMCLK
030156 040620      3$: 040400!<11*20>
030160 104414                      ROMCLK
030162 061224                      61224
030164 111205                      MOVB      (R2),R5
030166 116104 000004                      MOVB      4(R1),R4
030172 120504                      CMPB      R5,R4
030174 001401                      BEQ      4$
030176 104015                      EMT      15
030200 104401      4$: SCOP1
030202 005202                      INC      R2
030204 005200                      INC      R0
030206 022700 000010                      CMP      #10,R0
030212 001342                      BNE      1$
030214 104400                      SCOPE
030216 000      377      000 5$: .BYTE 0,-1,0,-1,125,252,125,252
030221 377      125      252
030224 125      252
```

.EVEN

90

```
***** 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

```
030226 012737 000116 001226 TST116: MOV      #116,TSTNO
030234 012737 030402 001216      MOV      #TST117,NEXT
030242 012737 030274 001220      MOV      #1$,LOCK
030250 104412                      MSTCLR
030252 005000                      CLR      R0
030254 012702 030372      MOV      #5$,R2
030260 004737 034726      JSR      PC,MEMLD
030264 035052                      MEMDAT
030266 004737 034762      JSR      PC,SPLD
;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;MEM + SP ADDRESS
;POINTER TO CORRECT DATA
;LOAD 8 WORDS OF MAIN MEMORY
;POINTER TO DATA
;LOAD 8 WORDS OF SP
```

030272	035062				SPDAT						
030274	004737	035040		1#:	JSR	PC,SETC					
030300	042737	000017	030314		BIC	#17,2#					
030306	050037	030314			BIS	R0,2#					
030312	104414				ROMCLK						
030314	010000			2#:	010000						
030316	042737	000017	030332		BIC	#17,3#					
030324	050037	030332			BIS	R0,3#					
030330	104414				ROMCLK						
030332	040600			3#:	040400!<10*20>						
030334	104414				ROMCLK						
030336	061224				61224						
030340	111205				MOVB	(R2),R5					
030342	116104	000004			MOVB	4(R1),R4					
030346	120504				CMPB	R5,R4					
030350	001401				BEQ	4#					
030352	104015				EMT	15					
030354	104401			4#:	SCOP1						
030356	005202				INC	R2					
030360	005200				INC	R0					
030362	022700	000010			CMP	#10,R0					
030366	001342				BNE	1#					
030370	104400				SCOPE						
030372	000	000	377	5#:	.BYTE	0,0,-1,-1,125,125,252,252					
030375	377	125	125								
030400	252	252									

.EVEN

91

***** 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
;*****

; TEST 117

030402	012737	000117	001226	TST117:	MOV	#117,TSTNO					
030410	012737	030556	001216		MOV	#TST120,NEXT					
030416	012737	030450	001220		MOV	#1#,LOCK					
030424	104412				MSTCLR						
030426	005000				CLR	R0					
030430	012702	030546			MOV	#5#,R2					
030434	004737	034726			JSR	PC,MEMLD					
030440	035052				MEMDAT						
030442	004737	034762			JSR	PC,SPLD					
030446	035062				SPDAT						
030450	004737	035040		1#:	JSR	PC,SETC					
030454	042737	000017	030470		BIC	#17,2#					
030462	050037	030470			BIS	R0,2#					
030466	104414				ROMCLK						
030470	010000			2#:	010000						
030472	042737	000017	030506		BIC	#17,3#					
030500	050037	030506			BIS	R0,3#					

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;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

030504	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
030506	040640			3\$:	040400!<12*20>		;BR A OR NOTB
030510	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
030512	061224				61224		;MOVE BR TO PORT4
030514	111205				MOVB (R2),R5		;PUT "EXPECTED" IN R5
030516	116104	000004			MOVB 4(R1),R4		;PUT "FOUND" IN R4
030522	120504				CMPB R5,R4		;DATA CORRECT?
030524	001401				BEQ 4\$		;BR IF YES
030526	104015				EMT 15		;ALU ERROR
030530	104401			4\$:	SCOP1		;SW09=1?
030532	005202				INC R2		;NEXT DATA
030534	005200				INC R0		;NEXT ADDRESS
030536	022700	000010			CMP #10,R0		;DONE YET?
030542	001342				BNE 1\$		;BR IF NO
030544	104400				SCOPE		;SCOPE THIS TEST
030546	377	000	377	5\$:	.BYTE -1.0,-1,-1,-1.125,252,-1		
030551	377	377	125				
030554	252	377					

.EVEN

92

***** 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
:*****

; TEST 120

030556	012737	000120	001226	TST120:	MOV	#120,TSTNO	
030564	012737	030732	001216		MOV	#TST121,NEXT	
030572	012737	030624	001220		MOV	#1\$,LOCK	
030600	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
030602	005000				CLR R0		;MASTER CLEAR DMC11
030604	012702	030722			MOV #5\$,R2		;MEM + SP ADDRESS
030610	004737	034726			JSR PC,MEMLD		;POINTER TO CORRECT DATA
030614	035052				MEMDAT		;LOAD 8 WORDS OF MAIN MEMORY
030616	004737	034762			JSR PC,SPLD		;POINTER TO DATA
030622	035062				SPDAT		;LOAD 8 WORDS OF SP
030624	004737	035040		1\$:	JSR PC,SETC		;POINTER TO DATA
030630	042737	000017	030644		BIC #17,2\$		;SET C BIT!
030636	050037	030644			BIS R0,2\$		;CLEAR ADDRESS FIELD OF INSTRUCTION
030642	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
030644	010000			2\$:	010000		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
030646	042737	000017	030662		BIC #17,3\$		;LOAD MAR
030654	050037	030662			BIS R0,3\$		;CLEAR ADDRESS OF INSTRUCTION
030660	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
030662	040660			3\$:	040400!<13*20>		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
030664	104414				ROMCLK		;BR A AND B
030666	061224				61224		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
030670	111205				MOVB (R2),R5		;MOVE BR TO PORT4
030672	116104	000004			MOVB 4(R1),R4		;PUT "EXPECTED" IN R5
030676	120504				CMPB R5,R4		;PUT "FOUND" IN R4
030700	001401				BEQ 4\$		;DATA CORRECT?
							;BR IF YES

```
030702 104015          EMT      15          ;ALU ERROR
030704 104401          SCOP1          ;SW09=1?
030706 005202          INC      R2          ;NEXT DATA
030710 005200          INC      R0          ;NEXT ADDRESS
030712 022700 000010    CMP      #10,R0     ;DONE YET?
030716 001342          BNE      1$          ;BR IF NO
030720 104400          SCOPE          ;SCOPE THIS TEST
030722 000 000 000 5$: .BYTE 0,0,0,-1,125,0,0,252
030725 377 125 000
030730 000 252
```

.EVEN

93

```
***** 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

```
030732 012737 000121 001226 TST121: MOV      #121,TSTNO
030740 012737 031106 001216      MOV      #TST122,NEXT
030746 012737 031000 001220      MOV      #1$,LOCK

030754 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
030756 005000          CLR      R0          ;MASTER CLEAR DMC11
030760 012702 031076    MOV      #5$,R2     ;MEM + SP ADDRESS
030764 004737 034726    JSR      PC,MEMLD    ;POINTER TO CORRECT DATA
030770 035052          MEMDAT          ;LOAD 8 WORDS OF MAIN MEMORY
030772 004737 034762    JSR      PC,SPLD     ;PCINTER TO DATA
030776 035062          SPDAT          ;LOAD 8 WORDS OF SP
031000 004737 035040    JSR      PC,SETC     ;POINTER TO DATA
031004 042737 000017 031020 1$: BIC      #17,2$ ;SET C BIT!
031012 050037 031020    BIS      R0,2$      ;CLEAR ADDRESS FIELD OF INSTRUCTION
031016 104414          ROMCLK          ;ADD ADDRESS TO INSTRUCTION
031020 010000          010000          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
031022 042737 000017 031036 2$: BIC      #17,3$ ;LOAD MAR
031030 050037 031036    BIS      R0,3$      ;CLEAR ADDRESS OF INSTRUCTION
031034 104414          ROMCLK          ;ADD ADDRESS TO INSTRUCTION
031036 040700          040400!<14*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
031040 104414          ROMCLK          ;BR - A OR B
031042 061224          61224          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
031044 111205          MOVB      (R2),R5    ;MOVE BR TO PORT4
031046 116104 000004    MOVB      4(R1),R4  ;PUT "EXPECTED" IN R5
031052 120504          CMPB      R5,R4      ;PUT "FOUND" IN R4
031054 001401          BEQ      4$          ;DATA CORRECT?
031056 104015          EMT      15          ;BR IF YES
031060 104401          SCOP1          ;ALU ERROR
031062 005202          INC      R2          ;SW09=1?
031064 005200          INC      R0          ;NEXT DATA
031066 022700 000010    CMP      #10,R0     ;NEXT ADDRESS
031072 001342          BNE      1$          ;DONE YET?
031074 104400          SCOPE          ;BR IF NO
031076 000 377 377 5$: .BYTE 0,-1,-1,-1,125,-1,-1,252 ;SCOPE THIS TEST
```


031101 377 125 377
031104 377 252

.EVEN

94

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

; TEST 122

031106	012737	000122	001226	TST122:	MOV	#122,TSTNO	
031114	012737	031262	001216		MOV	#TST123,NEXT	
031122	012737	031154	001220		MOV	#1\$,LOCK	
031130	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
031132	005000				CLR	R0	;MASTER CLEAR DMC11
031134	012702	031252			MOV	#5\$,R2	;MEM + SP ADDRESS
031140	004737	034726			JSR	PC,MEMLD	;POINTER TO CORRECT DATA
031144	035052				MEMDAT		;LOAD 8 WORDS OF MAIN MEMORY
031146	004737	034762			JSR	PC,SPLD	;POINTER TO DATA
031152	035062				SPDAT		;LOAD 8 WORDS OF SP
031154	004737	035040			JSR	PC,SETC	;POINTER TO DATA
031160	042737	000017	031174	1\$:	BIC	#17,2\$	;SET C BIT!
031166	050037	031174			BIS	R0,2\$	;CLEAR ADDRESS FIELD OF INSTRUCTION
031172	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
031174	010000			2\$:	010000		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
031176	042737	000017	031212		BIC	#17,3\$	;LOAD MAR
031204	050037	031212			BIS	R0,3\$	;CLEAR ADDRESS OF INSTRUCTION
031210	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
031212	040720			3\$:	040400!<15*20>		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
031214	104414				ROMCLK		;BR A XOR B
031216	061224				61224		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
031220	111205				MOVB	(R2),R5	;MOVE BR TO PORT4
031222	116104	000004			MOVB	4(R1),R4	;PUT "EXPECTED" IN R5
031226	120504				CMPB	R5,R4	;PUT "FOUND" IN R4
031230	001401				BEQ	4\$	;DATA CORRECT?
031232	104015				EMT	15	;BR IF YES
031234	104401			4\$:	SCOP1		;ALU ERROR
031236	005202				INC	R2	;SW09=1?
031240	005200				INC	R0	;NEXT DATA
031242	022700	000010			CMP	#10,R0	;NEXT ADDRESS
031246	001342				BNE	1\$	;DONE YET?
031250	104400				SCOPE		;BR IF NO
031252	000	377	377	5\$:	.BYTE	0,-1,-1,0,0,-1,-1,0	;SCOPE THIS TEST
031255	000	000	377				
031260	377	000					

.EVEN

95

***** TEST 123 *****
;ALU TEST
;TEST OF ALU FUNCTION ADD WITH C BIT SET

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

; TEST 123

```
031262 012737 000123 001226 TST123: MOV #123,TSTNO
031270 012737 031436 001216 MOV #TST124,NEXT
031276 012737 031330 001220 MOV #1$,LOCK

031304 104412 MSTCLR
031306 005000 CLR R0
031310 012702 031426 MOV #5$,R2
031314 004737 034726 JSR PC,MEMLD
031320 035052 MEMDAT
031322 004737 034762 JSR PC,SPLD
031326 035062 SPDAT
031330 004737 035040 1$: JSR PC,SETC
031334 042737 000017 031350 BIC #17,2$
031342 050037 031350 BIS R0,2$
031346 104414 ROMCLK
031350 010000 2$: 010000
031352 042737 000017 031366 BIC #17,3$
031360 050037 031366 BIS R0,3$
031364 104414 ROMCLK
031366 040400 3$: 040400!<00*20>
031370 104414 ROMCLK
031372 061224 61224
031374 111205 MOVB (R2),R5
031376 116104 000004 MOVB 4(R1),R4
031402 120504 CMPB R5,R4
031404 001401 BEQ 4$
031406 104015 EMT 15
031410 104401 4$: SCOP1
031412 005202 INC R2
031414 005200 INC R0
031416 022700 000010 CMP #10,R0
031422 001342 BNE 1$
031424 104400 SCOPE
031426 000 377 377 5$: .BYTE 0,-1,-1,376,252,-1,-1,124
031431 376 252 377
031434 377 124
```

.EVEN

96

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

; TEST 124

```
031436 012737 000124 001226 TST124: MOV #124,TSTNO
```


97

: TEST 125

ADDRESS	DATA	OPERATION	COMMENT
031612	012737 000125 001226	TST125: MOV #125,TSTNO	
031620	012737 031766 001216	MOV #TST126,NEXT	
031626	012737 031660 001220	MOV #1\$,LOCK	
031634	104412	MSTCLR	;R1 CONTAINS BASE DMC11 ADDRESS
031636	005000	CLR R0	;MASTER CLEAR DMC11
031640	012702 031756	MOV #5\$,R2	;MEM + SP ADDRESS
031644	004737 034726	JSR PC,MEMLD	;POINTER TO CORRECT DATA
031650	035052	MEMDAT	;LOAD 8 WORDS OF MAIN MEMORY
			;POINTER TO DATA

031652	004737	034762		JSR	PC,SPLD	;LOAD 8 WORDS OF SP
031656	035062			SPDAT		;POINTER TO DATA
031660	004737	035040	1\$:	JSR	PC,SETC	;SET C BIT!
031664	042737	000017	031700	BIC	#17,2\$	;CLEAR ADDRESS FIELD OF INSTRUCTION
031672	050037	031700		BIS	R0,2\$	;ADD ADDRESS TO INSTRUCTION
031676	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
031700	010000		2\$:	010000		;LOAD MAR
031702	042737	000017	031716	BIC	#17,3\$	;CLEAR ADDRESS OF INSTRUCTION
031710	050037	031716		BIS	R0,3\$	;ADD ADDRESS TO INSTRUCTION
031714	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
031716	040740		3\$:	040400!	<16*20>	;BR SUB
031720	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
031722	061224			61224		;MOVE BR TO PORT4
031724	111205			MOVB	(R2),R5	;PUT "EXPECTED" IN R5
031726	116104	000004		MOVB	4(R1),R4	;PUT "FOUND" IN R4
031732	120504			CMPB	R5,R4	;DATA CORRECT?
031734	001401			BEQ	4\$	;BR IF YES
031736	104015			EMT	15	;ALU ERROR
031740	104401		4\$:	SCOP1		;SW09=1?
031742	005202			INC	R2	;NEXT DATA
031744	005200			INC	R0	;NEXT ADDRESS
031746	022700	000010		CMP	#10,R0	;DONE YET?
031752	001342			BNE	1\$	;BR IF NO
031754	104400			SCOPE		;SCOPE THIS TEST
031756	000	001	377	.BYTE	0,1,-1,0,0,253,125,0	
031761	000	000	253			
031764	125	000				

.EVEN

98

***** 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

031766	012737	000126	001226	TST126:	MOV	#126,TSTNO	
031774	012737	032142	001216		MOV	#TST127,NEXT	
032002	012737	032034	001220		MOV	#1\$,LOCK	
032010	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
032012	005000				CLR	R0	;MASTER CLEAR DMC11
032014	012702	032132			MOV	#5\$,R2	;MEM + SP ADDRESS
032020	004737	034726			JSR	PC,MEMLD	;POINTER TO CORRECT DATA
032024	035052				MEMDAT		;LOAD 8 WORDS OF MAIN MEMORY
032026	004737	034762			JSR	PC,SPLD	;POINTER TO DATA
032032	035062				SPDAT		;LOAD 8 WORDS OF SP
032034	004737	035040	1\$:	JSR	PC,SETC		;POINTER TO DATA
032040	042737	000017	032054	BIC	#17,2\$		;SET C BIT!
032046	050037	032054		BIS	R0,2\$		;CLEAR ADDRESS FIELD OF INSTRUCTION
032052	104414			ROMCLK			;ADD ADDRESS TO INSTRUCTION
032054	010000		2\$:	010000			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
032056	042737	000017	032072	BIC	#17,3\$		;LOAD MAR
							;CLEAR ADDRESS OF INSTRUCTION

99

```

;***** 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

PC	OP	OP2	OP3	OP4	OP5	OP6	OP7	OP8	OP9	OP10	OP11	OP12	OP13	OP14	OP15	OP16	OP17	OP18	OP19	OP20	OP21	OP22	OP23	OP24	OP25	OP26	OP27	OP28	OP29	OP30	OP31	OP32	OP33	OP34	OP35	OP36	OP37	OP38	OP39	OP40	OP41	OP42	OP43	OP44	OP45	OP46	OP47	OP48	OP49	OP50	OP51	OP52	OP53	OP54	OP55	OP56	OP57	OP58	OP59	OP60	OP61	OP62	OP63	OP64	OP65	OP66	OP67	OP68	OP69	OP70	OP71	OP72	OP73	OP74	OP75	OP76	OP77	OP78	OP79	OP80	OP81	OP82	OP83	OP84	OP85	OP86	OP87	OP88	OP89	OP90	OP91	OP92	OP93	OP94	OP95	OP96	OP97	OP98	OP99	OP100	OP101	OP102	OP103	OP104	OP105	OP106	OP107	OP108	OP109	OP110	OP111	OP112	OP113	OP114	OP115	OP116	OP117	OP118	OP119	OP120	OP121	OP122	OP123	OP124	OP125	OP126	OP127	OP128	OP129	OP130	OP131	OP132	OP133	OP134	OP135	OP136	OP137	OP138	OP139	OP140	OP141	OP142	OP143	OP144	OP145	OP146	OP147	OP148	OP149	OP150	OP151	OP152	OP153	OP154	OP155	OP156	OP157	OP158	OP159	OP160	OP161	OP162	OP163	OP164	OP165	OP166	OP167	OP168	OP169	OP170	OP171	OP172	OP173	OP174	OP175	OP176	OP177	OP178	OP179	OP180	OP181	OP182	OP183	OP184	OP185	OP186	OP187	OP188	OP189	OP190	OP191	OP192	OP193	OP194	OP195	OP196	OP197	OP198	OP199	OP200	OP201	OP202	OP203	OP204	OP205	OP206	OP207	OP208	OP209	OP210	OP211	OP212	OP213	OP214	OP215	OP216	OP217	OP218	OP219	OP220	OP221	OP222	OP223	OP224	OP225	OP226	OP227	OP228	OP229	OP230	OP231	OP232	OP233	OP234	OP235	OP236	OP237	OP238	OP239	OP240	OP241	OP242	OP243	OP244	OP245	OP246	OP247	OP248	OP249	OP250	OP251	OP252	OP253	OP254	OP255	OP256	OP257	OP258	OP259	OP260	OP261	OP262	OP263	OP264	OP265	OP266	OP267	OP268	OP269	OP270	OP271	OP272	OP273	OP274	OP275	OP276	OP277	OP278	OP279	OP280	OP281	OP282	OP283	OP284	OP285	OP286	OP287	OP288	OP289	OP290	OP291	OP292	OP293	OP294	OP295	OP296	OP297	OP298	OP299	OP300	OP301	OP302	OP303	OP304	OP305	OP306	OP307	OP308	OP309	OP310	OP311	OP312	OP313	OP314	OP315	OP316	OP317	OP318	OP319	OP320	OP321	OP322	OP323	OP324	OP325	OP326	OP327	OP328	OP329	OP330	OP331	OP332	OP333	OP334	OP335	OP336	OP337	OP338	OP339	OP340	OP341	OP342	OP343	OP344	OP345	OP346	OP347	OP348	OP349	OP350	OP351	OP352	OP353	OP354	OP355	OP356	OP357	OP358	OP359	OP360	OP361	OP362	OP363	OP364	OP365	OP366	OP367	OP368	OP369	OP370	OP371	OP372	OP373	OP374	OP375	OP376	OP377	OP378	OP379	OP380	OP381	OP382	OP383	OP384	OP385	OP386	OP387	OP388	OP389	OP390	OP391	OP392	OP393	OP394	OP395	OP396	OP397	OP398	OP399	OP400	OP401	OP402	OP403	OP404	OP405	OP406	OP407	OP408	OP409	OP410	OP411	OP412	OP413	OP414	OP415	OP416	OP417	OP418	OP419
----	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

100

: TEST 130

```

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;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 INC A
;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

```


032462	001	001	000	5\$:	.BYTE	1,1,0,0,126,126,253,253
032465	000	126	126			
032470	253	253				

.EVEN

101

```
***** TEST 131 *****
;ALU TEST
;TEST OF ALU FUNCTION 2A WITH C BIT SET
;ALU FUNCTION (A PLUS A) CODE=5
;LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
;PERFORM THE FUNCTION, VERIFY THE RESULTS
;*****
```

; TEST 131

032472	012737	000131	001226	TST131:	MOV	#131,TSTNO	
032500	012737	032646	001216		MOV	#TST132,NEXT	
032506	012737	032540	001220		MOV	#1\$,LOCK	
032514	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
032516	005000				CLR	R0	;MASTER CLEAR DMC11
032520	012702	032636			MOV	#5\$,R2	;MEM + SP ADDRESS
032524	004737	034726			JSR	PC,MEMLD	;POINTER TO CORRECT DATA
032530	035052				MEMDAT		;LOAD 8 WORDS OF MAIN MEMORY
032532	004737	034762			JSR	PC,SPLD	;POINTER TO DATA
032536	035062				SPDAT		;LOAD 8 WORDS OF SP
032540	004737	035040			JSR	PC,SETC	;POINTER TO DATA
032544	042737	000017	032560	1\$:	BIC	#17,2\$	;SET C BIT!
032552	050037	032560			BIS	R0,2\$	;CLEAR ADDRESS FIELD OF INSTRUCTION
032556	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
032560	010000			2\$:	010000		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
032562	042737	000017	032576		BIC	#17,3\$	;LOAD MAR
032570	050037	032576			BIS	R0,3\$	;CLEAR ADDRESS OF INSTRUCTION
032574	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
032576	040520			3\$:	040400!<5*20>		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
032600	104414				ROMCLK		;BR 2A
032602	061224				61224		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
032604	111205				MOVB	(R2),R5	;MOVE BR TO PORT4
032606	116104	000004			MOVB	4(R1),R4	;PUT "EXPECTED" IN R5
032612	120504				CMPB	R5,R4	;PUT "FOUND" IN R4
032614	001401				BEQ	4\$	;DATA CORRECT?
032616	104015				EMT	15	;BR IF YES
032620	104401			4\$:	SCOP1		;ALU ERROR
032622	005202				INC	R2	;SW09=1?
032624	005200				INC	R0	;NEXT DATA
032626	022700	000010			CMP	#10,R0	;NEXT ADDRESS
032632	001342				BNE	1\$	;DONE YET?
032634	104400				SCOPE		;BR IF NO
032636	000	000	376	5\$:	.BYTE	0,0,376,376,252,252,124,124	;SCOPE THIS TEST
032641	376	252	252				
032644	124	124					

.EVEN

102

```
***** 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

```
032646 012737 000132 001226 TST132: MOV #132,TSTNO
032654 012737 033022 001216 MOV #TST133,NEXT
032662 012737 032714 001220 MOV #1$,LOCK

032670 104412 MSTCLR
032672 005000 CLR R0
032674 012702 033012 MOV #5$,R2
032700 004737 034726 JSR PC,MEMLD
032704 035052 MEMDAT
032706 004737 034762 JSR PC,SPLD
032712 035062 SPDAT
032714 004737 035040 1$: JSR PC,SETC
032720 042737 000017 032734 BIC #17,2$
032726 050037 032734 BIS R0,2$
032732 104414 ROMCLK
032734 010000 2$: 010000
032736 042737 000017 032752 BIC #17,3$
032744 050037 032752 BIS R0,3$
032750 104414 ROMCLK
032752 040500 3$: 040400!<4*20>
032754 104414 ROMCLK
032756 061224 61224
032760 111205 MOVB (R2),R5
032762 116104 000004 MOVB 4(R1),R4
032766 120504 CMPB R5,R4
032770 001401 BEQ 4$
032772 104015 EMT 15
032774 104401 4$: SCOP1
032776 005202 INC R2
033000 005200 INC R0
033002 022700 000010 CMP #10,R0
033006 001342 BNE 1$
033010 104400 SCOPE
033012 001 001 000 5$: .BYTE 1,1,0,0,126,126,253,253
033015 000 126 126
033020 253 253
```

.EVEN

```
***** 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

;-----

033022	012737	000133	001226	TST133:	MOV	#133,TSTNO	
033030	012737	033176	001216		MOV	#TST134,NEXT	
033036	012737	033070	001220		MOV	#1\$,LOCK	
033044	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
033046	005000				CLR	R0	;MASTER CLEAR DMC11
033050	012702	033166			MOV	#5\$,R2	;MEM + SP ADDRESS
033054	004737	034726			JSR	PC,MEMLD	;POINTER TO CORRECT DATA
033060	035052				MEMDAT		;LOAD 8 WORDS OF MAIN MEMORY
033062	004737	034762			JSR	PC,SPLD	;POINTER TO DATA
033066	035062				SPDAT		;LOAD 8 WORDS OF SP
033070	004737	035040			JSR	PC,SETC	;POINTER TO DATA
033074	042737	000017	033110	1\$:	BIC	#17,2\$	;SET C BIT!
033102	050037	033110			BIS	R0,2\$	;CLEAR ADDRESS FIELD OF INSTRUCTION
033106	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
033110	010000			2\$:	010000		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
033112	042737	000017	033126		BIC	#17,3\$	;LOAD MAR
033120	050037	033126			BIS	R0,3\$	;CLEAR ADDRESS OF INSTRUCTION
033124	104414				ROMCLK		;ADD ADDRESS TO INSTRUCTION
033126	040760			3\$:	040400!<17*20>		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
033130	104414				ROMCLK		;BR 2'S COMP SUB
033132	061224				61224		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
033134	111205				MOVB	(R2),R5	;MOVE BR TO PORT4
033136	116104	000004			MOVB	4(R1),R4	;PUT "EXPECTED" IN R5
033142	120504				CMPB	R5,R4	;PUT "FOUND" IN R4
033144	001401				BEQ	4\$	;DATA CORRECT?
033146	104015				EMT	15	;BR IF YES
033150	104401			4\$:	SCOP1		;ALU ERROR
033152	005202				INC	R2	;SW09=1?
033154	005200				INC	R0	;NEXT DATA
033156	022700	000010			CMP	#10,R0	;NEXT ADDRESS
033162	001342				BNE	1\$	;DONE YET?
033164	104400				SCOPE		;BR IF NO
033166	377	000	376	5\$:	.BYTE	-1,0,376,-1,-1,252,124,-1	;SCOPE THIS TEST
033171	377	377	252				
033174	124	377					

.EVEN

104

***** 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

033176	012737	000134	001226	TST134:	MOV	#134,TSTNO	
033204	012737	033352	001216		MOV	#TST135,NEXT	
033212	012737	033244	001220		MOV	#1\$,LOCK	
033220	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
033222	005000				CLR	R0	;MASTER CLEAR DMC11
033224	012702	033342			MOV	#5\$,R2	;MEM + SP ADDRESS
033230	004737	034726			JSR	PC,MEMLD	;POINTER TO CORRECT DATA
							;LOAD 8 WORDS OF MAIN MEMORY

105

TEST 135

033352	012737	000135	001226	TST135:	MOV	#135,TSTNO	
033360	012737	033544	001216		MOV	#TST136,NEXT	
							;R1 CONTAINS BASE DMC11 ADDRESS
033366	104412				MSTCLR		;MASTER CLEAR DMC11
033370	005037	001416			CLR	TEMP	;PREPARE FOR
033374	005037	001246			CLR	TEMP1	;DELAY
033400	012702	000011			MOV	#11,R2	;SAVE FOR TYPEOUT
033404	012761	000020	000004	1\$:	MOV	#20,4(R1)	;LOAD PORT 4
033412	152761	000002	000001		BISB	#BIT1,1(R1)	;SET ROMI
033420	012761	121111	000006		MOV	#121111,6(R1)	;SEL6 - INSTRUCTION
033426	152761	000003	000001		BISB	#BIT1!BIT0,1(R1)	;SET CLOCK BIT
033434	012761	121224	000006		MOV	#121224,6(R1)	;LOAD NEXT INSTRUCTION
033442	152761	000003	000001		BISB	#BIT1!BIT0,1(R1)	;READ CLOCK BIT
033450	142761	000003	000001		BICB	#BIT1!BIT0,1(R1)	;CLEAR MAINT BITS
033456	016104	000004			MOV	4(R1),R4	;PUT "FOUND" IN R4
033462	005005				CLR	R5	;PUT "EXPECTED" IN R5
033464	120504				CMPB	R5,R4	;IS PGM CLOCK CLEAR?
033466	001401				BEQ	2\$	

033470	104016		24:	EMT	16	;ERROR, PGM CLOCK IS NOT CLEAR
033472						
033472	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
033474	121224			121224		;PORT4_LU11
033476	122761	000020	000004	CMPB	#20,4(R1)	;IS PGM CLOCK SET?
033504	001416			BEQ	34	;BR IF YES
033506	062737	000001	001416	ADD	#1,TEMP	;INCREMENT DELAY
033514	005537	001246		ADC	TEMP1	;INCREMENT DELAY
033520	022737	000006	001246	CMP	#6,TEMP1	;IS DELAY DONE
033526	001361			BNE	24	;BR IF NO
033530	012705	000020		MOV	#20,R5	;PUT "EXPECTED" IN R5
033534	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
033540	104016			EMT	16	;ERROR PGM CLOCK NOT SET
033542	104400		34:	SCOPE		;SCOPE THIS TEST

106

***** TEST 136 *****
;FORCE POWER FAIL TEST
;SET FORCE POWER FAIL BIT VERIFY THAT PROCESSOR TRAPS TO 24
;GOING DOWN AND COMING UP. VERIFY ALSO THAT BUS INIT WAS
;BLOCKED FROM GETTING TO THE DMC DURING THE POWER FAIL
;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 33362 WAS 33544 SB 33736
;*****

				; TEST 136		
				;-----		
033544	012737	000136	001226	TST136:	MOV	#136,TSTNO
033552	012737	033766	001216		MOV	#TST137,NEXT
						;R1 CONTAINS BASE DMC11 ADDRESS
033560	104412			MSTCLR		;MASTER CLEAR DMC11
033562	005037	001416		CLR	TEMP	;PREPARE FOR DELAY
033566	013746	000024		MOV	#24,-(SP)	;STORE POWER FAIL ADDRESS
033572	012737	033636	000024	MOV	#14,0#24	;SET UP FOR FORCE POWER FAIL
033600	012761	000002	000004	MOV	#2,4(R1)	;LOAD PORT4
033606	012711	001000		MOV	#BIT9,(R1)	;SET ROMI
033612	012761	121111	000006	MOV	#121111,6(R1)	;LOAD INSTRUCTION
033620	012711	001400		MOV	#BIT9!BIT8,(R1)	;CLOCK INSTRUCTION
033624	005237	001416		INC	TEMP	;WAIT FOR POWER FAIL
033630	001375			BNE	54	;BR IF DELAY NOT DONE
033632	104017			EMT	17	;ERROR, NO POWER FAIL
033634	000426			BR	44	
033636	012737	033654	000024	14:	MOV	#34,0#24
033644	010637	033652			MOV	SP,24
033650	000000			HALT		;WAIT FOR POWER UP SEQUENCE
033652	000000			0		
033654	013706	033652		24:	MOV	24,SP
033660	022626			34:	POP2SP	;RESTORE STACK
033662	012637	000024			MOV	(SP)+,0#24
033666	022737	005336	000024		CMP	#.PFail,0#24
033674	001406				BEQ	44
						;RESTORE TRUE POWER FAIL ADDRESS
						;IS IT CORRECT?
						;BR IF YES

```
033676 104017      EMT      17      ;ERROR, STACK IS INCORRECT
033700 012737 005336 000024      MOV      @.PFAIL,@#24 ;RESTORE TRUE POWER FAIL ADDRESS
033706 012706 001200      MOV      @STACK,SP ;RESTORE STACK
033712 005737 000042      TST      @#42      ; CHECK FOR ACT-11 OR DDP
033716 001407      BEQ      7$
033720 012701 000140      MOV      @140, R1
033724 005000      CLR      R0      ; DELAY TO WAIT FOR RK05 TO BECOME
033726 005300      DEC      R0      ; READY AFTER POWER FAIL
033730 001376      BNE      8$
033732 005301      DEC      R1
033734 001373      BNE      6$
033736 013701 001404      MOV      DMCSR,R1 ;RESTORE ADDRESS. R1 CLOBBED ON 11/84 PWRFAIL
033742 012711 003000      MOV      @BIT9:BIT10,(R1);SEL6 = MAINT IR
033746 012705 121111      MOV      @121111,R5 ;R5 = EXPECTED
033752 016104 000004      MOV      4(R1),R4 ;R4 = FOUND
033756 020504      CMP      R5,R4 ;MAINT IR SHOULD = 12111
033760 001401      BEQ      .+4 ;BR IF OK
033762 104025      EMT      25 ;IF = 0 THEN BUS INIT WAS
                                ;NOT BLOCKED FROM CLEARING
                                ;THE DMC-11
                                ;SCOPE THIS TEST

107 033764 104400      SCOPE
```

```
***** 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 PATTERNS TO VERIFY THAT
;READING AND WRITING OF OTHER LOCATIONS AND REGISTERS
;DID NOT CHANGE THE DATA.
*****
```

; TEST 137

```
033766 012737 000137 001226 TST137: MOV      @137,TSTND
033774 012737 003364 001216      MOV      @.EOP,NEXT

034002 104412      MSTCLR
034004 005002      CLR      R2      ;R1 CONTAINS BASE DMC11 ADDRESS
034006 042737 000017 034032 1$: BIC      @17,2$ ;MASTER CLEAR DMC11
034014 156237 034626 034032      BISB     30$(R2),2$ ;R2 IS INDEX REGISTER
034022 116261 034634 000004      MOV      31$(R2),4(R1) ;CLEAR ADDRESS FIELD
034030 104414      ROMCLK ;ADD IBUS* REG ADDRESS TO INSTRUCTION
034032 121100      121100 ;LOAD PORT4
034034 005202      INC      R2 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034036 022702 000005      CMP      @5,R2 ;WRITE IBUS* REGISTER
034042 001361      BNE      1$ ;INC INDEX REGISTER
034044 005002      CLR      R2 ;DONE YET?
034046 042737 000017 034112 3$: BIC      @17,4$ ;BR IF NO
034054 042737 000017 034124      BIC      @17,5$ ;R2 IS IBUS REGISTER ADDRESS
034062 042737 000017 034134      BIC      @17,6$ ;CLEAR ADDRESS FIELD OF INSTRUCTIONS
034070 050237 034112      BIS      R2,4$ ;ADD IBUS REG ADDRESS TO INSTRUCTION
034074 050237 034124      BIS      R2,5$
034100 050237 034134      BIS      R2,6$
034104 105061 000004      CLRB     4(R1) ;CLEAR PORT4
034110 104414      ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034112 122100      122100 ;WRITE 0 TO IBUS REG
```


034114	112761	000377	000004	MOVB	#377,4(R1)	;LOAD PORT4
034122	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034124	122100		5\$:	122100		;WRITE ALL ONES TO IBUS REG
034126	110261	000004		MOVB	R2,4(R1)	;LOAD PORT4
034132	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034134	122100		6\$:	122100		;WRITE ITS OWN ADDRESS TO IBUS REG
034136	005202			INC	R2	;NEXT ADDRESS
034140	022702	000010		CMP	#10,R2	;DONE YET?
034144	001340			BNE	3\$	;BR IF NO
034146	005002			CLR	R2	;START AT SP ADDRESS 0
034150	042737	000017	034214	BIC	#17,8\$	;CLEAR ADDRESS FIELD
034156	042737	000017	034226	BIC	#17,9\$	
034164	042737	000017	034236	BIC	#17,10\$	
034172	050237	034214		BIS	R2,8\$	;ADD ADDRESS TO INSTRUCTION
034176	050237	034226		BIS	R2,9\$	
034202	050237	034236		BIS	R2,10\$	
034206	105061	000004		CLRB	4(R1)	;CLEAR PORT4
034212	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034214	123100		8\$:	123100		;WRITE ZERO TO SP
034216	112761	000377	000004	MOVB	#377,4(R1)	;LOAD PORT4
034224	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034226	123100		9\$:	123100		;WRITE ALL ONES TO SP
034230	110261	000004		MOVB	R2,4(R1)	;LOAD PORT4
034234	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034236	123100		10\$:	123100		;WRITE SP ADDRESS TO ITSELF
034240	005202			INC	R2	;NEXT SP ADDRESS
034242	022702	000020		CMP	#20,R2	;DONE YET?
034246	001340			BNE	7\$	;BR IF NO
034250	005002			CLR	R2	;R2 = MAIN MEM ADDRESS
034252	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034254	010000			010000		;MAR = 0
034256	105061	000004	11\$:	CLRB	4(R1)	;CLEAR PORT4
034262	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034264	122500			122500		;WRITE ZEROS TO MEM
034266	112761	000377	000004	MOVB	#377,4(R1)	;LOAD PORT4
034274	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034276	122500			122500		;WRITE ONES TO MEM
034300	110261	000004		MOVB	R2,4(R1)	;LOAD PORT4
034304	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034306	136500			136500		;WRITE TO MEM IT OWN ADDRESS
034310	005202			INC	R2	;NEXT MEM ADDRESS
034312	022702	000400		CMP	#400,R2	;DONE YET?
034316	001357			BNE	11\$	;BR IF NO
;NOW GO BACK AND READ EVERYTHING						
034320	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034322	010000			010000		;MAR_0
034324	032737	100000	001366	BIT	#BIT15,STAT1	;DMC?
034332	001402			BEQ	.+6	;BR IF YES
034334	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
034336	004000			4000		;MAR HI = 0 (KMC ONLY)
034340	005000			CLR	R0	;R0 IS INDEX REGISTER
034342	042737	000360	034400	BIC	#360,13\$	;CLEAR ADDRESS FIELD
034350	116002	034626	12\$:	MOVB	30\$(R0),R2	;R2 = IBUS* ADDRESS
034354	010203			MOV	R2,R3	;PUT IBUS* ADDRESS IN R3
034356	006303			ASL	R3	;SHIFT ADDRESS TO BITS 4-7

034360	006303			ASL	R3		
034362	006303			ASL	R3		
034364	006303			ASL	R3		
034366	050337	034400		BIS	R3,13\$	;ADD ADDRESS TO INSTRUCTION	
034372	116005	034634		MOVB	31\$(R0),R5	;R5 = "EXPECTED"	
034376	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
034400	121004		13\$:	121004		;PORT4 - IBUS* REGISTER	
034402	016104	000004		MOV	4(R1),R4	;R4 = "FOUND"	
034406	120504			CMPB	R5,R4	;IBUS* CONTENTS OK?	
034410	001401			BEQ	.+4	;BR IF YES	
034412	104004			EMT	4	;IBUS* DATA ERROR	
034414	005200			INC	R0	;INC COUNTER	
034416	022700	000005		CMP	#5,R0	;DONE YET?	
034422	001347			BNE	12\$	;BR IF NO	
034424	005002			CLR	R2	;R2 = IBUS REG ADDRESS	
034426	042737	000360	034456	14\$:	BIC	#360,15\$	;CLEAR ADDRESS FIELD OF INSTRUCTION
034434	010203			MOV	R2,R3	;R3 = IBUS ADDRESS	
034436	006303			ASL	R3	;SHIFT ADDRESS TO BITS 4-7	
034440	006303			ASL	R3		
034442	006303			ASL	R3		
034444	006303			ASL	R3		
034446	050337	034456		BIS	R3,15\$	;ADD ADDRESS TO INSTRUCTION	
034452	010205			MOV	R2,R5	;R5 = "EXPECTED"	
034454	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
034456	021004		15\$:	021004		;PORT4 - IBUS REG	
034460	016104	000004		MOV	4(R1),R4	;R4 = "FOUND"	
034464	120504			CMPB	R5,R4	;IBUS CONTENTS OK?	
034466	001401			BEQ	.+4	;BR IF YES	
034470	104005			EMT	5	;IBUS DATA ERROR	
034472	005202			INC	R2	;NEXT IBUS REGISTER	
034474	022702	000010		CMP	#10,R2	;DONE YET?	
034500	001352			BNE	14\$	;BR IF NO	
034502	005002			CLR	R2	;R2 = SP ADDRESS	
034504	042737	000017	034520	16\$:	BIC	#17,17\$;CLEAR	ADRESS FIELD OF INSTRUCTION
034512	050237	034520		BIS	R2,17\$	;ADD ADDRESS TO INSTRUCTION	
034516	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
034520	040600		17\$:	040600		;BR - SP	
034522	010205			MOV	R2,R5	;R5 = "EXPECTED"	
034524	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
034526	061224			061224		;PORT4 - BR	
034530	016104	000004		MOV	4(R1),R4	;R4 = "FOUND"	
034534	120504			CMPB	R5,R4	;SP CONTENTS OK?	
034536	001401			BEQ	.+4	;BR IF YES	
034540	104007			EMT	7	;SP DATA ERROR	
034542	005202			INC	R2	;NEXT SP LOCATION	
034544	022702	000020		CMP	#20,R2	;DONE YET?	
034550	001355			BNE	16\$	;BR IF NO	
034552	005002			CLR	R2	;R2 = MEMORY ADDRESS	
034554	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
034556	010000			010000		;MAR - 0	
034560	032737	100000	001366	BIT	#BIT15,STAT1	;DMC?	
034566	001402			BEQ	.+6	;BR IF YES	
034570	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
034572	004000			4000		;MAR HI - 0 (KMC ONLY)	
034574	010205		18\$:	MOV	R2,R5	;R5 = "EXPECTED"	
034576	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
034600	055224			055224		;PORT4 - MAIN MEM	


```
034602 016104 000004      MOV    4(R1),R4      ;R4 = "FOUND"
034606 120504      CMPB   R5,R4      ;MAIN MEM CONTENTS OK?
034610 001401      BEQ     .+4      ;BR IF YES
034612 1C4013      EMT     13      ;MAIN MEM DATA ERROR
034614 005202      INC     R2      ;NEXT MEM ADDRESS
034616 022702 000400      CMP     #400,R2      ;DONE YET?
034622 001364      BNE     18$      ;BR IF NO
034624 104400      SCOPE   0,2,3,5,10 ;SCOPE THIS TEST
034626 000 002 003 30$: .BYTE
034631 005 010      .EVEN
034634 001 003 004 31$: .BYTE 1,3,4,6,10
034637 006 010      .EVEN

110
111
112      ;SUBROUTINES
113      ;-----
114
115 034642      SETVEC:
116      ;THIS SUBROUTINE LOADS THE VECTORS AND VECTOR LEVELS
117
118 034642 012577 144526      MOV     (R5)+, @DMRVEC      ;LOAD BASE VECTOR
119 034646 012577 144526      MOV     (R5)+, @DMTVEC      ;LOAD VECTOR + 2
120 034652 112577 144520      MOVB    (R5)+, @DMRLVL      ;LOAD VECTOR + 4
121 034656 112577 144520      MOVB    (R5)+, @DMTLVL      ;LOAD VECTOR + 6
122 034662 000205      RTS     R5      ;RETURN
123
124
125 034664      NPRSET:
126      ;THIS SUBROUTINE LOADS IBUS REGISTERS 0-7
127      ;WITH NPR INFORMATION (INBA, OUTBA, OUT DATA)
128
129 034664 010246      MOV     R2,-(SP)      ;SAVE R2
130 034666 005002      CLR     R2      ;START AT IBUS REG 0
131 034670 112561 000004      MOVB    (R5)+, 4(R1)      ;LOAD PORT4
132 034674 042737 000017 034710 1$: BIC     #17,2$      ;CLEAR ADDRESS FIELD OF INSTRUCTION
133 034702 050237 034710      BIS     R2,2$      ;ADD ADDRESS TO INSTRUCTION
134 034706 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
135 034710 122100 2$: 122100      ;MOVE PORT4 TO IBUS REG
136 034712 005202      INC     R2      ;NEXT ADDRESS
137 034714 022702 000010      CMP     #10,R2      ;ALL DONE?
138 034720 001363      BNE     1$      ;BR IF NO
139 034722 012602      MOV     (SP)+,R2      ;RESTORE R2
140 034724 000205      RTS     R5      ;RETURN
141
142
143 034726      MEMLD:
144      ;THIS SUBROUTINE LOADS THE FIRST 8 LOCATIONS OF MAIN
145      ;MEMORY WITH THIS DATA: 0,-1,,0,-1,125,252,125,252
146
147 034726 013605      MOV     @ (SP)+,R5      ;PUT POINTER TO DATA IN R5
148 034730 062746 000002      ADD     #2,-(SP)      ;ADJUST STACK
149 034734 012704 000010      MOV     #10,R4      ;DO 8 LOADS
150 034740 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
151 034742 010000      010000      ;MAR < 0
152 034744 112577 144442 1$: MOVB    (R5)+, @DMP04      ;LOAD PORT4
```

```
153 034750 104414
154 034752 136500
155 034754 005304
156 034756 001372
157 034760 000207
158
159
162 034762
163
164
165
166 034762 013605
167 034764 062746 000002
168 034770 005004
169 034772 112577 144414
170 034776 042737 000017 035012
171 035004 050437 035012
172 035010 104414
173 035012 123100
174 035014 005204
175 035016 022704 000010
176 035022 001363
177 035024 000207
178
179
180 035026
181
182
183 035026 104414
184 035030 010000
185 035032 104414
186 035034 040400
187 035036 000207
188
189
190 035040
191
192
193 035040 104414
194 035042 010003
195 035044 104414
196 035046 040403
197 035050 000207
198
199
200 035052 000 377 000 MEMDAT: .BYTE 0,-1,0,-1,125,252,125,252
    035055 377 125 252
    035060 125 252
201 035062 000 000 377 SPDAT: .BYTE 0,0,-1,-1,125,125,252,252
    035065 377 125 125
    035070 252 252
202
207 035072 377 125 116 .EVEN
208 035137 377 125 116 EM1: .ASCIZ <377>/UNIBUS REGISTER ADDRESSING TIME-OUT/
209 035201 377 115 111 EM2: .ASCIZ <377>/UNIBUS REGISTER WRITE, READ TEST/
210 035227 377 115 111 EM3: .ASCIZ <377>/MICRO PROCESSOR TEST/
211 035271 377 102 122 EM4: .ASCIZ <377>/MICRO PROCESSOR WRITE, READ TEST/
    EM5: .ASCIZ <377>/BR REGISTER TEST/
```

ROMCLK 136500 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
DEC R4 ;MOV DATA TO MEM, AUTO INC MAR
BNE 1\$;DECREMENT COUNT
RTS PC ;BR IF NOT DONE
;RETURN

SPLD: ;THIS SUBROUTINE LOADS THE FIRST 8 SCRATCH PAD
;LOCATIONS WITH: 0,0,-1,-1,125,125,252,252

1\$: MOV @ (SP)+,R5 ;PUT POINTER TO DATA IN R5
ADD #2,-(SP) ;ADJUST STACK
CLR R4 ;START AT SP ADDRESS 0
MOVB (R5)+,@DMP04 ;LOAD PORT4 WITH DATA
BIC #17,2\$;CLEAR ADDRESS FIELD OF INSTRUCTION
BIS R4,2\$;ADD ADDRESS TO INSTRUCTION
ROMCLK 123100 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2\$: INC R4 ;MOVE DATA TO SP
CMP #10,R4 ;INCREMENT COUNT
BNE 1\$;DONE YET?
RTS PC ;BR IF NO
;RETURN

CLRC: ;THIS SUBROUTINE CLEARS THE MICRO PROCESSOR C BIT

ROMCLK 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
ROMCLK 040400!<0*20> ;MAR_0
RTS PC ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;CLEAR C BIT
;RETURN

SETC: ;THIS SUBROUTINE SETS THE MICRO PROCESSOR C BIT

ROMCLK 010003 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
ROMCLK 040403!<0*20> ;MAR_3
RTS PC ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET C BIT
;RETURN

212	035313	377	123	103	EM6:	.ASCIZ	<377>/SCRATCH PAD TEST/		
213	035335	377	104	105	EM7:	.ASCIZ	<377>/DEVICE FAILED TO INTERRUPT/		
214	035371	377	104	105	EM10:	.ASCIZ	<377>/DEVICE INTERRUPTED TO WRONG VECTOR/		
215	035435	377	116	120	EM11:	.ASCIZ	<377>/NPR TEST/		
216	035447	377	115	101	EM12:	.ASCIZ	<377>/MAIN MEMORY TEST/		
217	035471	377	115	101	EM13:	.ASCIZ	<377>/MAR TEST/		
218	035503	377	101	114	EM14:	.ASCIZ	<377>/ALU TEST/		
219	035515	377	120	122	EM15:	.ASCIZ	<377>/PROGRAM CLOCK TEST/		
220	035541	377	106	117	EM16:	.ASCIZ	<377>/FORCE POWER FAIL ERROR/		
221	035571	377	125	116	EM17:	.ASCIZ	<377>/UNEXPECTED INTERRUPT/		
222	035617	377	104	115	EM20:	.ASCIZ	<377>/DMC11 CONFIGURATION ERROR/		
223	035652	377	115	101	EM21:	.ASCIZ	<377>/MAINTENANCE INSTRUCTION REGISTER TEST/		
224	035721	377	120	117	EM22:	.ASCIZ	<377>/POWER FAIL INITIALIZE FAILURE/		
225									
226	035760	377	122	105	DH1:	.ASCIZ	<377>/REGISTER	TRAPPED FROM/	
227	036021	377	105	130	DH2:	.ASCIZ	<377>/EXPECTED FOUND	REGISTER/	
228	036057	377	105	130	DH3:	.ASCIZ	<377>/EXPECTED FOUND/		
229	036100	377	105	130	DH4:	.ASCIZ	<377>/EXPECTED FOUND	IBUS* REGISTER/	
230	036142	377	105	130	DH5:	.ASCIZ	<377>/EXPECTED FOUND	IBUS REGISTER/	
231	036203	377	105	130	DH6:	.ASCIZ	<377>/EXPECTED FOUND	ADDRESS/	
232					.EVEN				
233									
234	036240	000002			DT1:	2			
235	036242	006	015		.BYTE	6,15			
236	036244	001262			SAVR1				
237	036246	006	002		.BYTE	6,2			
238	036250	001264			SAVR2				
239	036252	000003			DT2:	3			
240	036254	006	004		.BYTE	6,4			
241	036256	001272			SAVR5				
242	036260	006	004		.BYTE	6,4			
243	036262	001270			SAVR4				
244	036264	006	002		.BYTE	6,2			
245	036266	001262			SAVR1				
246	036270	000002			DT3:	2			
247	036272	006	004		.BYTE	6,4			
248	036274	001272			SAVR5				
249	036276	006	002		.BYTE	6,2			
250	036300	001270			SAVR4				
251	036302	000003			DT4:	3			
252	036304	003	007		.BYTE	3,7			
253	036306	001272			SAVR5				
254	036310	003	011		.BYTE	3,11			
255	036312	001270			SAVR4				
256	036314	002	002		.BYTE	2,2			
257	036316	001264			SAVR2				
258	036320	000003			DT5:	3			
259	036322	003	007		.BYTE	3,7			
260	036324	001272			SAVR5				
261	036326	003	007		.BYTE	3,7			
262	036330	001270			SAVR4				
263	036332	006	002		.BYTE	6,2			
264	036334	001264			SAVR2				
265	036336	000002			DT6:	2			
266	036340	003	007		.BYTE	3,7			
267	036342	001272			SAVR5				
268	036344	003	002		.BYTE	3,2			

269	036346	001270			SAVR4		
270	036350	000002			2		
271	036352	006	004		.BYTE	6,4	
272	036354	001262			SAVR1		
273	036356	006	002		.BYTE	6,2	
274	036360	001404			DMCSR		
275	036362				.ERRTAB:		
276	036362	000000			0		
277	036364	000000			0		
278	036366	000000			0		
279	036370	035072			EM1		
280	036372	035760			DH1	:EMT	1
281	036374	036240			DT1		
282	036376	035137			EM2		
283	036400	036021			DH2	:EMT	2
284	036402	036252			DT2		
285	036404	035201			EM3		
286	036406	036057			DH3	:EMT	3
287	036410	036270			DT3		
288	036412	035227			EM4		
289	036414	036100			DH4	:EMT	4
290	036416	036302			DT4		
291	036420	035227			EM4		
292	036422	036142			DH5	:EMT	5
293	036424	036302			DT4		
294	036426	035271			EM5		
295	036430	036057			DH3	:EMT	6
296	036432	036336			DT6		
297	036434	035313			EM6		
298	036436	036203			DH6	:EMT	7
299	036440	036320			DT5		
300	036442	035335			EM7		
301	036444	000000			0	:EMT	10
302	036446	000000			0		
303	036450	035371			EM10		
304	036452	000000			0	:EMT	11
305	036454	000000			0		
306	036456	035435			EM11		
307	036460	036057			DH3	:EMT	12
308	036462	036270			DT3		
309	036464	035447			EM12		
310	036466	036203			DH6	:EMT	13
311	036470	036320			DT5		
312	036472	035471			EM13		
313	036474	036203			DH6	:EMT	14
314	036476	036320			DT5		
315	036500	035503			EM14		
316	036502	036057			DH3	:EMT	15
317	036504	036336			DT6		
318	036506	035515			EM15		
319	036510	036100			DH4	:EMT	16
320	036512	036302			DT4		
321	036514	035541			EM16		
322	036516	000000			0	:EMT	17
323	036520	000000			0		
324	036522	035571			EM17		
325	036524	000000			0	:EMT	20

326	036526	000000	0		
327	036530	035435	EM11		
328	036532	036203	DH6	;EMT	21
329	036534	036320	DT5		
330	036536	035617	EM20		
331	036540	036057	DH3	;EMT	22
332	036542	036350	DT7		
333	036544	035652	EM21		
334	036546	036057	DH3	;EMT	23
335	036550	036270	DT3		
336	036552	035617	EM20		
337	036554	000000	0	;EMT	24
338	036556	000000	0		
339	036560	035721	EM22		
340	036562	036057	DH3	;EMT	25
341	036564	036270	DT3		
342					
343					
344	036566				
349		000001			

.EVEN
CORMAX:
.END

ADRCNT = 004373	DH5 036142	DMS213 001634	ERCT06 001734	MPFAIL 005675
AUDONE 003024	DH6 036203	DMS214 001644	ERCT07 001740	MQM 005666
AUSTRT 002446	DISPLA 001200	DMS215 001654	ERCT10 001744	MR 005755
AUTO.S 010512	DISPRE 000174	DMS216 001664	ERCT11 001750	MRESET = 004000
BINWRD 004714	DMACTV 001306	DMS217 001674	ERCT12 001754	MSTCLR = 104412
BIT0 = 000001	DMCM 007320	DMS300 001506	ERCT13 001760	MTITLE 001000
BIT1 = 000002	DMCR00 001500	DMS301 001516	ERCT14 001764	MTSTN 006130
BIT10 = 002000	DMCR01 001510	DMS302 001526	ERCT15 001770	MTSTPC 006031
BIT11 = 004000	DMCR02 001520	DMS303 001536	ERCT16 001774	MVECX 006100
BIT12 = 010000	DMCR03 001530	DMS304 001546	ERCT17 002000	NEXT 001216
BIT13 = 020000	DMCR04 001540	DMS305 001556	ERR 002700	NOACT 007154
BIT14 = 040000	DMCR05 001550	DMS306 001566	ERRCNT 001232	NODEV 002674
BIT15 = 100000	DMCR06 001560	DMS307 001576	ERRFLG 001325	NPRSET 034664
BIT2 = 000004	DMCR07 001570	DMS310 001606	ERRMSG 005172	NUM 006450
BIT3 = 000010	DMCR10 001600	DMS311 001616	ERRPC 002770	OK 002646
BIT4 = 000020	DMCR11 001610	DMS312 001626	ERTAB0 005322	ONE 001302
BIT5 = 000040	DMCR12 001620	DMS313 001636	EXIT = 000205	PACT00 001702
BIT6 = 000100	DMCR13 001630	DMS314 001646	EXITER 005252	PACT01 001706
BIT7 = 000200	DMCR14 001640	DMS315 001656	FLOAT 002536	PACT02 001712
BIT8 = 000400	DMCR15 001650	DMS316 001666	FY 002566	PACT03 001716
BIT9 = 001000	DMCR16 001660	DMS317 001676	HALTS 005222	PACT04 001722
BM 007054	DMCR17 001670	DMTLVL 001402	HILIM 004366	PACT05 001726
BRLVL 012252	DMCSR 001404	DMTVEC 001400	ICOUNT 001222	PACT06 001732
BRW 003730	DMCSRH 001406	DM.END 001700	INBUF 007502	PACT07 001736
BRX 003732	DMCTL 001410	DM.MAP 001500	INCHAR 010020	PACT10 001742
CHRCNT 004712	DMNUM 001310	DONE 003734	INIFLG 001324	PACT11 001746
CKSWR 007606	DMP04 001412	DT1 036240	INSTER = 104404	PACT12 001752
CKSWR1 007666	DMP06 001414	DT2 036252	INSTR = 104403	PACT13 001756
CKSWR2 007700	DMRLVL 001376	DT3 036270	INSTR2 004166	PACT14 001762
CKSWR3 007704	DMRVEC 001374	DT4 036302	INTTY 012266	PACT15 001766
CKSWR4 007710	DMS100 001502	DT5 036320	KMCM 007330	PACT16 001772
CKSWR5 010014	DMS101 001512	DT6 036336	LIMITS 004314	PACT17 001776
CLKX 001242	DMS102 001522	DT7 036350	LINE 007016	PARAM = 104405
CLRC 035026	DMS103 001532	EM1 035072	LOBITS 004372	PARAM1 004234
CNERR 007277	DMS104 001542	EM10 035371	LOCK 001220	PARBIT = 040000
CNT.MA 001702	DMS105 001552	EM11 035435	LOKFLG 001326	PARERR 004310
CNVRT = 104411	DMS106 001562	EM12 035447	LOLIM 004364	PASCNT 001230
CONERR 007223	DMS107 001572	EM13 035471	LPCNT 001224	PERFOR = 004537
CONN 007114	DMS110 001602	EM14 035503	LSTERR 001234	PFTAB 005430
CONTAB 002776	DMS111 001612	EM15 035515	MASKX 001244	POPRO = 012600
CONVRT = 104410	DMS112 001622	EM16 035541	MASTEK 006142	POP1SP = 005726
CORMAX 036566	DMS113 001632	EM17 035571	MCRLF 005672	POP2SP = 022626
CRAM 006606	DMS114 001642	EM2 035137	MCSRX 006072	PRI0 006547
CREAM 001320	DMS115 001652	EM20 035617	MDATA 007544	PS = 177776
CSR 006510	DMS116 001662	EM21 035652	MEMDAT 035052	PUSHRO = 010046
CSRMAP 010514	DMS117 001672	EM22 035721	MEMLD 034726	PUSH1S = 005746
CYCLE 010060	DMS200 001504	EM3 035201	MEMLIM 001304	PUSH2S = 024646
DATABP 005216	DMS201 001514	EM4 035227	MEPASS 005733	QV.FLG 001327
DATACL = 104415	DMS202 001524	EM5 035271	MERRPC 006217	RESREG 005220
DATAHD 005204	DMS203 001534	EM6 035313	MERRX 006117	RESTAR 005350
DELAY = 104413	DMS204 001544	EM7 035335	MERR2 005760	RESTRT 003534
DEVADR 004370	DMS205 001554	ERCT00 001704	MERR3 006005	RES05 = 104407
DEVTAB 003010	DMS206 001564	ERCT01 001710	MILK 001322	RETURN 001214
DH1 035760	DMS207 001574	ERCT02 001714	MLOCK 006043	ROMCLK = 104414
DH2 036021	DMS210 001604	ERCT03 001720	MNEW 006144	RUN 001316
DH3 036057	DMS211 001614	ERCT04 001724	MODU 006704	SAVACT 001312
DH4 036100	DMS212 001624	ERCT05 001730	MPASSX 006106	SAVNUM 001314

SAVPC	001276	SW11	= 004000	TST122	031106	TST43	017610	WRKO.F	005174
SAVR0	001260	SW12	= 010000	TST123	031262	TST44	017764	XBX	005000
SAVR1	001262	SW13	= 020000	TST124	031436	TST45	020140	XCSR	003546
SAVR2	001264	SW14	= 040000	TST125	031612	TST46	020314	XERR	003570
SAVR3	001266	SW15	= 100000	TST126	031766	TST47	020470	XHEAD	006224
SAVR4	001270	TDATA	024542	TST127	032142	TST5	012704	XLOC	003022
SAVR5	001272	TEMP	001416	TST13	013470	TST50	020644	XPASS	003562
SAVSP	001274	TEMP1	001246	TST130	032316	TST51	021072	XSTATQ	007454
SAV05	= 104406	TEMP2	001250	TST131	032472	TST52	021242	XTSTN	005330
SCOPE	= 104400	TEMP3	001252	TST132	032646	TST53	021510	XVEC	003554
SCOP1	= 104401	TEMP4	001254	TST133	033022	TST54	021732	ZERO	001300
SETC	035040	TEMP5	001256	TST134	033176	TST55	022026	\$CRAP	= 177777
SETVEC	034642	TIMER	= 104416	TST135	033352	TST56	022120	\$ENDAD	003522
SKIP	002632	TKCSR	001204	TST136	033544	TST57	022240	\$N	= 000137
SOFTSW	010052	TKDBR	001206	TST137	033766	TST6	013002	\$S	= 000141
SPACNT	= 004713	TLAST	= 033766	TST14	013566	TST60	022404	\$Y	= 000017
SPDAT	035062	TPCSR	001210	TST15	013664	TST61	022510	.BEGIN	003152
SPEED	007340	TPDBR	001212	TST16	013762	TST62	022624	.CNVRT	004472
SPLD	034762	TRPOK	004730	TST17	014060	TST63	022726	.CONVR	004466
STACK	= 001200	TSTNO	001226	TST2	012426	TST64	023064	.DATAC	005552
STAT	001240	TST1	012320	TST20	014156	TST65	023210	.DELAY	005436
STAT1	001366	TST10	013176	TST21	014254	TST66	023320	.EOP	003364
STAT2	001370	TST100	025256	TST22	014352	TST67	023426	.ERRTA	036362
STAT3	001372	TST101	025432	TST23	014450	TST7	013100	.HLT	004750
STRTSW	001236	TST102	025606	TST24	014546	TST70	023624	.INSTE	004154
SV05	004402	TST103	025762	TST25	014672	TST71	023752	.INSTR	004050
SWFLG	010016	TST104	026136	TST26	015016	TST72	024104	.INST1	004070
SWMES	007205	TST105	026312	TST27	015146	TST73	024304	.MSG	004072
SWMES1	007215	TST106	026466	TST3	012456	TST74	024440	.MSTCL	005466
SWR	001202	TST107	026642	TST30	015306	TST75	024552	.PARAM	004174
SWREG	000176	TST11	013274	TST31	015450	TST76	024726	.PFAIL	005336
SW00	= 000001	TST110	027016	TST32	015534	TST77	025102	.RES05	004434
SW01	= 000002	TST111	027172	TST33	015734	TTST	003612	.ROMCL	005504
SW02	= 000004	TST112	027346	TST34	016134	TWOSYN	= 010000	.SAV05	004374
SW03	= 000010	TST113	027522	TST35	016310	TYPDAT	005206	.SCOPE	003576
SW04	= 000020	TST114	027676	TST36	016464	TYPE	= 104402	.SCOP1	003736
SW05	= 000040	TST115	030052	TST37	016664	TYPMSG	005106	.START	002002
SW06	= 000100	TST116	030226	TST4	012606	VEC	006526	.TIMER	005616
SW07	= 000200	TST117	030402	TST40	017104	VECMAP	012010	.TRPSR	004716
SW08	= 000400	TST12	013372	TST41	017260	WHICH	012002	.TRPTA	001330
SW09	= 001000	TST120	030556	TST42	017434	WRDCNT	004710	.TYPE	003766
SW10	= 002000	TST121	030732						

. ABS. 036566 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 53760 WORDS (210 PAGES)
DYNAMIC MEMORY AVAILABLE FOR 71 PAGES
CZDMCD.BIN,CZDMCD.SEQ=CZDMCD.P11