

DMC-11

BASIC W/R MICRO PRO
CZDMCC0

AH-8545C-MC

COPYRIGHT '76-78

FICHE 1 OF 1

JAN 1979

digital

MADE IN USA



.REM 8

IDENTIFICATION

PRODUCT CODE: AC-8544C-MC
PRODUCT NAME: CZDMCCO BSC W/R MICRO-PROC TST
DATE: SEPTEMBER 1978
MAINTAINER: DIAGNOSTICS

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1976, 1978 BY DIGITAL EQUIPMENT CORPORATION

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)

92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134

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.

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.

220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262

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 ME JE
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 ENABLED; AND THERE IS A HARD ERROR (CONSTANT); SW08 IS BEST. (SW14=1,0, SW10=0, SW09=0, SW08=1). FOR INTERMITTENT ERRORS; SW14=1 WILL LOOP ON TEST REGARDLESS OF ERROR OR NOT ERROR. (SW14=1, SW10=0, SW09 0, SW08=1,0)
3. SW11 INHIBIT INTERACTIONS.
4. SW14 LOOP ON CURRENT TEST.

4.2 STARTING ADDRESS

STARTING ADDRESS IS AT 000200 THERE ARE NO OTHER STARTING ADDRESSES FOR THE 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 CUMULATIVE 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:.

408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463

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.

DMC000-DMC17
DMST00-DMST17
(1500)-(1640)

THESE LOCATIONS CONTAIN THE INFORMATION NEEDED TO TEST UP TO 16 (DECIMAL) DMC11S SEQUENTIALLY. 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

464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492

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

CSR: CONTAINS DMC11 CSR ADDRESS

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

STAT2: LOW BYTE IS SWITCH PAC#1 (DDCMP LINE NUMBER)
HIGH BYTE IS SWITCH PAC#2 (BM873 BOOI 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 OF 'MC11 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 OCCURS 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 ('77777) THIS SOFTWARE SWITCH REGISTER IS USED.

CONTROL:

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

549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587

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.

```

588
589
590
591
592
593      ;*AC-8544C-MC  CZDMCCO BASIC DMC11 CONTROLLER TEST
594      ;*COPYRIGHT 1976,1978, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
595      ;-----
596
597      ;STARTING PROCEDURE
598      ;LOAD PROGRAM
599      ;LOAD ADDRESS 000200
600      ;SWR=0  AUTOSIZE DMC11
601      ;SW07=1  USE CURRENT DMC11 PARAMETERS
602      ;SW00=1  INPUT NEW DMC11 PARAMETERS
603      ;PRESS START
604      ;PROGRAM WILL TYPE 'AC-8544C-MC  CZDMCCO BASIC DMC11 CONTROLLER TEST'
605      ;PROGRAM WILL TYPE STATUS MAP
606      ;PROGRAM WILL TYPE 'R' TO INDICATE THAT TESTING HAS STARTED
607      ;AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
608      ;AND THEN RESUME TESTING
609      ;SUBSEQUENT RESTARTS WILL NOT TYPE PROGRAM TITLE
610
611
612
613
614      ;SWITCH REGISTER OPTIONS
615      ;-----
616
617      100000      SW15-100000      ; 1,HALT ON ERROR
618      040000      SW14-40000       ; 1,LOOP ON CURRENT TEST
619      020000      SW13-20000       ; 1,INHIBIT ERROR TIMEOUT
620      010000      SW12=10000       ; 1,DELETE TIMEOUT/BELL ON ERROR.
621      004000      SW11=4000        ; 1,INHIBIT ITERATIONS
622      002000      SW10=2000        ; -1,ESCAPE TO NEXT TEST ON ERROR
623      001000      SW09=1000        ; 1,LOOP WITH CURRENT DATA
624      000400      SW08=400         ; 1,LOOP ON ERROR
625      000200      SW07=200         ; 1,USE CURRENT DMC11 PARAMETERS, =0,AUTOSIZE DMC11
626      000100      SW06=100        ; 1, HALT BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION
627      000040      SW05=40
628      000020      SW04=20
629      000010      SW03=10         ;RESELECT DMC11'S TO BE TESTED (ACTIVE)
630      000004      SW02=4          ;LOCK ON TEST SELECT
631      000002      SW01=2          ;RESTART PROGRAM AT SELECTED TEST
632      000001      SW00=1          ;INPUT DMC11 PARAMETERS
  
```

```

633
634
635          ;REGISTER DEFINITIONS
636          ;-----
637
638          000000      R0=%0          ;GENERAL REGISTER
639          000001      R1=%1          ;GENERAL REGISTER
640          000002      R2=%2          ;GENERAL REGISTER
641          000003      R3=%3          ;GENERAL REGISTER
642          000004      R4=%4          ;GENERAL REGISTER
643          000005      R5=%5          ;GENERAL REGISTER
644          000006      SP=%6         ;PROCESSOR STACK POINTER
645          000007      PC=%7         ;PROGRAM COUNTER
646
647          ;LOCATION EQUIVALENCIES
648          ;-----
649
650          177776      PS=177776      ;PROCESSOR STATUS WORD
651          001200      STACK=1200     ;START OF PROCESSOR STACK
652
653          ;INSTRUCTION DEFINITIONS
654          ;-----
655
656          005746      PUSH1SP=5746    ;DECREMENT PROCESSOR STACK 1 WORD
657          005726      POP1SP=5726     ;INCREMENT PROCESSOR STACK 1 WORD
658          010046      PUSHRO 10046    ;SAVE R0 ON STACK
659          012600      POPRO=12600     ;RESTORE R0 FROM STACK
660          024646      PUSH2SP=24646   ;DECREMENT STACK TWICE
661          022626      POP2SP=22626    ;INCREMENT STACK TWICE
662          .EQUIV EMT.HLT ;BASIC DEFINITION OF ERROR CALL
663
664          ;BIT DEFINITIONS
665          ;-----
666
667          100000      BIT15=100000
668          040000      BIT14=40000
669          020000      BIT13=20000
670          010000      BIT12=10000
671          004000      BIT11=4000
672          002000      BIT10=2000
673          001000      BIT9=1000
674          000400      BIT8=400
675          000200      BIT7=200
676          000100      BIT6=100
677          000040      BIT5=40
678          000020      BIT4=20
679          000010      BIT3=10
680          000004      BIT2=4
681          000002      BIT1=2
682          000001      BIT0=1
683
684

```

```

685
686
687
688
689
690
691
692
693
694
695      000000
696
697
698
699
700      000024 005336
701      000026 000340
702      000030 004750
703      000032 000340
704      000034 004716
705      000036 000340
706
707      000040 000000
708      000042 000000
709      000044 000000
710      000046 003522
711
712      000052 000000
713
714
715      000174 000000
716      000176 000000
717
718
719      000200 000137 002002
720
721
722
723      001000 005377 041501 034055
724      (2) 001016 055103 046504 041503
725      (2)
726
727
728
729      001200 177570
730      001202 177570
  
```

```

;*****
;-----
;TRAPCATCHER FOR ILLEGAL INTERRUPTS
;THE STANDARD 'TRAP CATCHER' IS PLACED
;BETWEEN ADDRESS 0 TO ADDRESS 776.
;IT LOOKS LIKE 'PC+2 HALT'.
;-----
;*****
.=0
;STANDARD INTERRUPT VECTORS
;-----
. 24
      .PFAIL      ;POWER FAIL HANDLER
      340          ;SERVICE AT LEVEL 7
      .HLT        ;ERROR HANDLER
      340          ;SERVICE AT LEVEL 7
      .TRPSRV     ;GENERAL HANDLER DISPATCH SERVICE
      340          ;SERVICE AT LEVEL 7
. 40
      0            ;SAVE FOR ACT-11 OR XXDP
      0            ;RETURN ADDRESS IF UNDER ACT-11 OR XXDP
      0            ;SAVE FOR ACT-11 OR XXDP
      $ENDAD      ;FOR USE WITH ACT-11 OR XXDP
. 52
      0            ;ACT-11 PROGRAM CHARACTERISTICS
. 174
DISPREG:0          ;SOFTWARE DISPLAY REGISTER
SWREG: 0           ;SOFTWARE SWITCH REGISTER
.=200
      JMP      .START      ;GO TO START OF PROGRAM
.=1000
MTITLE: .ASCII <377><12>/AC-8544C-MC/<377>
        .ASCII /CZDMCC0 BASIC DMC11 CONTROLLER TEST/<377>
.=1200
;INDIRECT POINTERS TO SWITCH REGISTER AND LIGHT DISPLAY
;-----
DISPLAY:177570
SWR: 177570
  
```



```

731
732      ;INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS
733      ;-----
734
735      001204 177560      TKCSR: 177560      ;TELETYPE KEYBOARD CONTROL REGISTER
736      001206 177562      TKDBR: 177562      ;TELETYPE KEYBOARD DATA BUFFER
737      001210 177564      TPCSR: 177564      ;TELEPRINTER CONTROL REGISTER
738      001212 177566      TPDBR: 177566      ;TELEPRINTER DATA BUFFER
739
740      ;PROGRAM CONTROL PARAMETERS
741      ;-----
742
743      001214 000000      RETURN: 0      ;SCOPE ADDRESS FOR LOOP ON TEST
744      001216 000000      NEXT: 0      ;ADDRESS OF NEXT TEST TO BE EXECUTED
745      001220 000000      LOCK: 0      ;ADDRESS FOR LOCK ON CURRENT DATA
746      001222 000003      !COUNT: 3      ;NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE EXECUTED
747      001224 000000      LPCNT: 0      ;NUMBER OF ITERATIONS COMPLETED
748      001226 000000      TSTNO: 0      ;NUMBER OF TEST IN PROGRESS
749      001230 000000      PASCNT: 0      ;NUMBER OF PASSES COMPLETED
750      001232 000000      ERRCNT: 0      ;TOTAL NUMBER OF ERRORS
751      001234 000000      LSTERR: 0      ;PC OF LAST ERROR CALL
752
753      ;PROGRAM VARIABLES
754      ;-----
755
756      001236 000000      STRTSW: 0      ;SWITCHES AT START OF PROGRAM
757      001240 000000      STAT: 0      ;DM STATUS WORD STORAGE
758      001242 000000      CLKX: 0
759      001244 000000      MASKX: 0
760      001246 000000      TEMP1: 0      ;TEMPORARY STORAGE
761      001250 000000      TEMP2: 0      ;TEMPORARY STORAGE
762      001252 000000      TEMP3: 0      ;TEMPORARY STORAGE
763      001254 000000      TEMP4: 0      ;TEMPORARY STORAGE
764      001256 000000      TEMP5: 0      ;TEMPORARY STORAGE
765      001260 000000      SAVR0: 0      ;R0 STORAGE
766      001262 000000      SAVR1: 0      ;R1 STORAGE
767      001264 000000      SAVR2: 0      ;R2 STORAGE
768      001266 000000      SAVR3: 0      ;R3 STORAGE
769      001270 000000      SAVR4: 0      ;R4 STORAGE
770      001272 000000      SAVR5: 0      ;R5 STORAGE
771      001274 000000      SAVSP: 0      ;STACK POINTER STORAGE
772      001276 000000      SAVPC: 0      ;PROGRAM COUNTER STORAGE
773      001300 000000      ZERO: 0
774      001302 000001      ONE: 1
775      001304 000000      MEMLIM: 0
776      001306 000001      DMACTV: .BLKW 1      ;HIGHEST LOCATION FOR NPR'S
777      001310 000001      DMNUM: .BLKW 1      ;DMC11'S SELECTED ACTIVE.
778      001312 000001      SAVACT: .BLKW 1      ;OCTAL NUMBER OF DMC11'S.
779      001314 000001      SAVNUM: .BLKW 1      ;ORIGINAL ACTV DEVICES
780      001316 000000      RUN: 0      ;WORKABLE NUMBER
781      ;EVEN      ;POINTER TO RUNNING DEVICE.
782      001320 001472      CREAM: DM.MAP-6      ;TABLE POINTER.
783      001322 001676      MILK: CNT.MAP-4      ;TABLE POINTER
  
```

```
784
785 ;PROGRAM CONTROL FLAGS
786 ;-----
787
788 001324 000 INIFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG
789 001325 000 ERRFLG: .BYTE 0 ;ERROR OCCURED FLAG
790 001326 000 LOKFLG: .BYTE 0 ;LOCK ON CURRENT TEST FLAG
791 001327 000 QV.FLG: .BYTE 0 ;QUICK VERIFY FLAG.
792 ;ON FIRST PASS OF EACH DMC11 ITERATIONS WILL BE SUPPRESS
793 .EVEN
794
795 ;DEFINITIONS FOR TRAP SUBROUTINE CALLS
796 ;POINTERS TO SUBROUTINES CAN BE FOUND
797 ;IN THE TABLE IMMEDIATLY FOLLOWING THE DEFINITIONS
798
799 ;*****
800 ;-----
801 001330 ;TRPTAB:
802 104400 SCOPE=TRAP+0 ;CALL TO SCOPE LOOP AND ITERATION HANDLER
803 001330 003576 .SCOPE
804 104401 SLOP1=TRAP+1 ;CALL TO LOOP ON CURRENT DATA HANDLER
805 001332 003736 .SCOPI
806 104402 TYPE=TRAP+2 ;CALL TO TELETYPE OUTPUT ROUTINE
807 001334 003766 .TYPE
808 104403 INSTR=TRAP+3 ;CALL TO ASCII STRING INPUT ROUTINE
809 001336 004050 .INSTR
810 104404 INSTER=TRAP+4 ;CALL TO INPUT ERROR HANDLER
811 001340 004154 .INSTER
812 104405 PARAM=TRAP+5 ;CALL TO NUMERICAL DATA INPUT ROUTINE
813 001342 004174 .PARAM
814 104406 SAV05=TRAP+6 ;CALL TO REGISTER SAVE ROUTINE
815 001344 004374 .SAV05
816 104407 RES05=TRAP+7 ;CALL TO REGISTER RESTORE ROUTINE
817 001346 004434 .RES05
818 104410 CONVRT=TRAP+10 ;CALL TO DATA OUTPUT ROUTINE
819 001350 004466 .CONVRT
820 104411 CNVRT=TRAP+11 ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
821 001352 004472 .CNVRT
822 104412 MSTCLR=TRAP+12 ;CALL TO ISSUE A MASTER CLEAR
823 001354 005466 .MSTCLR
824 104413 DELAY=TRAP+13 ;CALL TO DELAY
825 001356 005436 .DELAY
826 104414 ROMCLK=TRAP+14 ;CALL TO CLOCK ROM ONCE
827 001360 005504 .ROMCLK
828 104415 DATACLK=TRAP+15 ;CALL TO CLK DATA
829 001362 005552 .DATACLK
830 104416 TIMER=TRAP+16 ;CALL TO DELAY A CLOCK TICK
831 001364 005616 .TIMER
832
833 ;-----
834 ;*****
```

```

835 ;DMC11 CONTROL INDICATORS FOR CURRENT DMC11 UNDER TEST
836 ;-----
837
838 001366 000000 STAT1: 0
839 001370 000000 STAT2: 0
840 001372 000000 STAT3: 0
841
842 ;DMC11 VECTOR AND REGISTER INDIRECT POINTERS
843 ;-----
844
845 001374 000000 DMRVEC: 0 ;POINTER TO DMC11 RECEIVER INTERRUPT VECTOR
846 001376 000000 DMRLVL: 0 ;POINTER TO DMC11 RECEIVER INTERRUPT SERVICE PS
847 001400 000000 DMTVEC: 0 ;POINTER TO DMC11 TRANSMITTER INTERRUPT VECTOR
848 001402 000000 DMTLVL: 0 ;POINTER TO DMC11 TRANSMITTER INTERRUPT SERVICE PS
849 001404 000000 DMCSR: 0 ;POINTER TO DMC11 CONTROL STATUS REGISTER
850 001406 000000 DMCSRH: 0 ;POINTER TO DMC11 CONTROL STATUS REGISTER HIGH BYTE.
851 001410 000000 DMCTL: 0 ;POINTER TO DMC11 CONTROL OUT REGISTER
852 001412 000000 DMP04: 0 ;POINTER TO DMC11 PORT REGISTER(SEL 4)
853 001414 000000 DMP06: 0 ;POINTER TO DMC11 PORT REGISTER(SEL 6)
854
855 ; EMP STORAGE
856 ;-----
857
858 001416 000000 TEMP: 0
859 .+40
860
861 ;DMC11 STATUS TABLE AND ADDRESS ASSIGNMENTS
862 ;-----
863
864 .-1500
865 001500 001500 DM.MAP:
866 001500 000001 DMCRO0: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 00
867 001502 000001 DMS100: .BLKW 1 ;VECTOR FOR DMC11 NUMBER 00
868 001504 000001 DMS200: .BLKW 1 ;DDCMP LINE# FOR DMC11 NUMBER 00
869 001506 000001 DMS300: .BLKW 1 ;3RD STATUS WORD
870
871 001510 000001 DMCRO1: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 01
872 001512 000001 DMS101: .BLKW 1 ;VECTOR FOR DMC11 NUMBER 01
873 001514 000001 DMS201: .BLKW 1 ;DDCMP LINE# FOR DMC11 NUMBER 01
874 001516 000001 DMS301: .BLKW 1 ;3RD STATUS WORD
875
876 001520 000001 DMCRO2: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 02
877 001522 000001 DMS102: .BLKW 1 ;VECTOR FOR DMC11 NUMBER 02
878 001524 000001 DMS202: .BLKW 1 ;DDCMP LINE# FOR DMC11 NUMBER 02
879 001526 000001 DMS302: .BLKW 1 ;3RD STATUS WORD
880
881 001530 000001 DMCRO3: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 03
882 001532 000001 DMS103: .BLKW 1 ;VECTOR FOR DMC11 NUMBER 03
883 001534 000001 DMS203: .BLKW 1 ;DDCMP LINE# FOR DMC11 NUMBER 03
884 001536 000001 DMS303: .BLKW 1 ;3RD STATUS WORD
885
886 001540 000001 DMCRO4: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 04
887 001542 000001 DMS104: .BLKW 1 ;VECTOR FOR DMC11 NUMBER 04
888 001544 000001 DMS204: .BLKW 1 ;DDCMP LINE# FOR DMC11 NUMBER 04
889 001546 000001 DMS304: .BLKW 1 ;3RD STATUS WORD
890

```

891	001550	000001	DMCR05: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 05
892	001552	000001	DMS105: .BLKW	1	:VECTOR FOR DMC11 NUMBER 05
893	001554	000001	DMS205: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 05
894	001556	000001	DMS305: .BLKW	1	:3RD STATUS WORD
895					
896	001560	000001	DMCR06: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 06
897	001562	000001	DMS106: .BLKW	1	:VECTOR FOR DMC11 NUMBER 06
898	001564	000001	DMS206: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 06
899	001566	000001	DMS306: .BLKW	1	:3RD STATUS WORD
900					
901	001570	000001	DMCR07: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 07
902	001572	000001	DMS107: .BLKW	1	:VECTOR FOR DMC11 NUMBER 07
903	001574	000001	DMS207: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 07
904	001576	000001	DMS307: .BLKW	1	:3RD STATUS WORD
905					
906	001600	000001	DMCR10: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 10
907	001602	000001	DMS110: .BLKW	1	:VECTOR FOR DMC11 NUMBER 10
908	001604	000001	DMS210: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 10
909	001606	000001	DMS310: .BLKW	1	:3RD STATUS WORD
910					
911	001610	000001	DMCR11: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 11
912	001612	000001	DMS111: .BLKW	1	:VECTOR FOR DMC11 NUMBER 11
913	001614	000001	DMS211: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 11
914	001616	000001	DMS311: .BLKW	1	:3RD STATUS WORD
915					
916	001620	000001	DMCR12: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 12
917	001622	000001	DMS112: .BLKW	1	:VECTOR FOR DMC11 NUMBER 12
918	001624	000001	DMS212: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 12
919	001626	000001	DMS312: .BLKW	1	:3RD STATUS WORD
920					
921	001630	000001	DMCR13: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 13
922	001632	000001	DMS113: .BLKW	1	:VECTOR FOR DMC11 NUMBER 13
923	001634	000001	DMS213: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 13
924	001636	000001	DMS313: .BLKW	1	:3RD STATUS WORD
925					
926	001640	000001	DMCR14: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 14
927	001642	000001	DMS114: .BLKW	1	:VECTOR FOR DMC11 NUMBER 14
928	001644	000001	DMS214: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 14
929	001646	000001	DMS314: .BLKW	1	:3RD STATUS WORD
930					
931	001650	000001	DMCR15: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 15
932	001652	000001	DMS115: .BLKW	1	:VECTOR FOR DMC11 NUMBER 15
933	001654	000001	DMS215: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 15
934	001656	000001	DMS315: .BLKW	1	:3RD STATUS WORD
935					
936	001660	000001	DMCR16: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 16
937	001662	000001	DMS116: .BLKW	1	:VECTOR FOR DMC11 NUMBER 16
938	001664	000001	DMS216: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 16
939	001666	000001	DMS316: .BLKW	1	:3RD STATUS WORD
940					
941	001670	000001	DMCR17: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 17
942	001672	000001	DMS117: .BLKW	1	:VECTOR FOR DMC11 NUMBER 17
943	001674	000001	DMS217: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 17
944	001676	000001	DMS317: .BLKW	1	:3RD STATUS WORD
945					
946	001700	000000	DM.END: 000000		

```

947
948
949      ;DMC11 PASS COUNT AND ERROR COUNT TABLE
950      ;-----
951      CNT.MAP:
952      PACT00: 0      ;PASS COUNT FOR DMC11 NUMBER 00
953      ERCT00: 0      ;ERROR COUNT FOR DMC11 NUMBER 00
954
955      PACT01: 0      ;PASS COUNT FOR DMC11 NUMBER 01
956      ERCT01: 0      ;ERROR COUNT FOR DMC11 NUMBER 01
957
958      PACT02: 0      ;PASS COUNT FOR DMC11 NUMBER 02
959      ERCT02: 0      ;ERROR COUNT FOR DMC11 NUMBER 02
960
961      PACT03: 0      ;PASS COUNT FOR DMC11 NUMBER 03
962      ERCT03: 0      ;ERROR COUNT FOR DMC11 NUMBER 03
963
964      PACT04: 0      ;PASS COUNT FOR DMC11 NUMBER 04
965      ERCT04: 0      ;ERROR COUNT FOR DMC11 NUMBER 04
966
967      PACT05: 0      ;PASS COUNT FOR DMC11 NUMBER 05
968      ERCT05: 0      ;ERROR COUNT FOR DMC11 NUMBER 05
969
970      PACT06: 0      ;PASS COUNT FOR DMC11 NUMBER 06
971      ERCT06: 0      ;ERROR COUNT FOR DMC11 NUMBER 06
972
973      PACT07: 0      ;PASS COUNT FOR DMC11 NUMBER 07
974      ERCT07: 0      ;ERROR COUNT FOR DMC11 NUMBER 07
975
976      PACT10: 0      ;PASS COUNT FOR DMC11 NUMBER 10
977      ERCT10: 0      ;ERROR COUNT FOR DMC11 NUMBER 10
978
979      PACT11: 0      ;PASS COUNT FOR DMC11 NUMBER 11
980      ERCT11: 0      ;ERROR COUNT FOR DMC11 NUMBER 11
981
982      PACT12: 0      ;PASS COUNT FOR DMC11 NUMBER 12
983      ERCT12: 0      ;ERROR COUNT FOR DMC11 NUMBER 12
984
985      PACT13: 0      ;PASS COUNT FOR DMC11 NUMBER 13
986      ERCT13: 0      ;ERROR COUNT FOR DMC11 NUMBER 13
987
988      PACT14: 0      ;PASS COUNT FOR DMC11 NUMBER 14
989      ERCT14: 0      ;ERROR COUNT FOR DMC11 NUMBER 14
990
991      PACT15: 0      ;PASS COUNT FOR DMC11 NUMBER 15
992      ERCT15: 0      ;ERROR COUNT FOR DMC11 NUMBER 15
993
994      PACT16: 0      ;PASS COUNT FOR DMC11 NUMBER 16
995      ERCT16: 0      ;ERROR COUNT FOR DMC11 NUMBER 16
996
997      PACT17: 0      ;PASS COUNT FOR DMC11 NUMBER 17
998      ERCT17: 0      ;ERROR COUNT FOR DMC11 NUMBER 17
999
  
```

1000

FORMAT OF STATUS TABLE

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

DEFINITION OF FORMAT

- SR: 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 DO FREE RUNNING TESTS ON KMC
 (MUST BE SET TO A ONE MANUALLY [PROGRAM DZDM 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

CZDMC MACY11 30A(1052) 24-AUG-78 12:19 PAGE 23
 CZDMCC.P11 11-AUG-78 08:21 PROGRAM INITIALIZATION AND START UP.

SEQ 0023

```

1055
1056      ;PROGRAM INITIALIZATION
1057      ;LOCK OUT INTERRUPTS
1058      ;SET UP PROCESSOR STACK
1059      ;SET UP POWER FAIL VECTOR
1060      ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
1061      ;TYPE TITLE MESSAGE
1062
1063 002002 012737 000340 177776 .START: MOV #340,PS ;LOCK OUT INTERRUPTS
1064 002010 012706 001200 MOV #STACK,SP ;SET UP STACK
1065 002014 012737 005336 000024 MOV #.PFAIL,@#24 ;SET UP POWER FAIL VECTOR
1066 002022 013737 001310 001314 MOV DMNUM,SAVNUM ;SAVE NUMBER OF DEVICES IN SYSTEM.
1067 002030 005037 010016 CLR SWFLG ;CLEAR SOFT TYPEOUT FLAG
1068 002034 105037 001325 CLRB ERRFLG ;CLEAR ERROR FLAG
1069 002040 105037 001327 CLRB QV.FLG ;ZERO QUICK VERIFY FLAG
1070 002044 012737 001470 001320 MOV #DM.MAP-10,CREAM ;GET MAP POINTER.
1071 002052 012737 001676 001322 MOV #CNT.MAP-4,MILK ;GET PASS COUNT MAP POINTER
1072 002060 012737 000000 001316 MOV #BIT15,RUN ;POINT POINTER TO FIRST DEVICE.
1073 002066 012700 001702 MOV #CNT.MAP,R0 ;PASS COUNT POINTER TO R0
1074 002072 005020 23$: CLR (R0)+ ;CLEAR TABLE
1075 002074 022700 002002 CMP #CNT.MAP+100,R0 ;DONE YET?
1076 002100 001374 BNE 23$ ;KEEP GOING
1077 002102 005037 001234 CLR LSTERR ;CLEAR LAST ERROR POINTER
1078 002106 012737 000001 001226 MOV #1,TSTNO ;SET UP FOR TEST 1
1079 002114 012737 002002 001214 MOV #.START,RETURN ;SET UP FOR POWER FAIL BEFORE
1080 ;TESTING STARTS
1081 002122 013746 000006 MOV @#6,-(SP) ;SAVE CURRENT VECTORS
1082 002126 013746 000004 MOV @#4,-(SP) ;
1083 002132 012737 002166 000004 MOV #6$,@#4 ;SET UP FOR TIMEOUT
1084 002140 012737 177570 001202 MOV #177570,SWR ;SET SWR TO HARD SWR ADDRESS
1085 002146 012737 177570 001200 MOV #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
1086 002154 022777 177777 177020 CMP #-1,@SWR ;REFERENCE HARDWARE SWITCH REGISTER
1087 002162 001402 BEQ 6$+2 ;IF = -1 USE SOFT SWR ANYWAY
1088 002164 000407 BR 7$ ;IF IT EXISTS AND NOT -1 USE HARD SWR
1089 002166 022626 6$: CMP (SP)+,(SP)+ ;ADJUST STACK
1090 002170 012737 000176 001202 MOV #SWREG,SWR ;POINTER TO SOFT SWR
1091 002176 012737 000174 001200 MOV #DISPREG,DISPLAY ;POINTER TO SOFT DISPLAY REG
1092 002204 012637 000004 7$: MOV (SP)+,@#4 ;RESTORE VECTORS
1093 002210 012637 000006 MOV (SP)+,@#6 ;
1094 002214 105737 001324 TSTB INIFLG ;HAS INITIALIZATION BEEN PERFORMED
1095 002220 001006 BNE 20$ ;BR IF YES
1096 002222 022737 003522 000042 CMP #SENDAD,@#42 ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
1097 002230 001402 BEQ 20$ ;
1098 002232 104402 001000 TYPE ,MTITLE ;TYPE TITLE MESSAGE
1099 002236 004737 007606 20$: JSR PC,CKSWR ;CHECK FOR SOFT SWR
1100 002242 017737 176734 001236 MOV @SWR,STRTSW ;STORE STARTING SWITCHES
1101 002250 005737 000042 TST @#42 ;IS IT RUNNING IN AUTO MODE?
1102 002254 001402 BEQ .+6 ;BR IF NO
1103 002256 005037 001236 CLRB STRTSW ;IF YES, CLEAR SWITCHES
1104 002262 032737 000001 001236 BIT #SW00,STRTSW ;IF SW00=1, QUESTIONS ARE ASKED.
1105 002270 001012 BNE 17$ ;BR IF SW00=1
1106 002272 105737 001236 TSTB STRTSW ;BIT7=1??
1107 002276 100007 BPL 17$ ;BR IF SW07=0
1108 002300 005737 001306 TST DMACTV ;ARE ANY DEVICES SELECTED?
1109 002304 001006 BNE 17$ ;BR IF YES
1110 002306 104402 007154 TYPE, NOACT ;NO DEVICES SELECTED.

```

```
1111 002312 000000          HALT          ;STOP THE SHOW
1112 002314 000776          BR           .-2      ;DISQUALIFY CONTINUE SWITCH
1113 002316 004737 010512    17$: JSR      PC,AUTO.SIZE ;GO DO THE AUTO SIZE
1114 002322 105737 001324    14$: TSTB     INIFLG    ;FIRST TIME?
1115 002326 001410          BEQ      21$      ;BR IF YES
1116 002330 105737 001236    TSTB     STRTSW    ;IF USING SAME PARAMETERS DONT TYPE MAP
1117 002334 100431          BMI      1$
1118 002336 032737 000006 001236 BIT      #BIT1.BIT2,STRTSW;IS TEST NO. OR LOCK SELECTED
1119 002344 001403          BEQ      24$      ;IF NO THEN TYPE STATUS
1120 002346 000424          BR           1$      ;IF YES DO NOT TYPE STATUS
1121 002350 005137 001324    21$: COM      INIFLG    ;SET FLAG
1122 002354 104402 006224    24$: TYPE     ,XHEAD    ;TYPE HEADER
1123 002360 012704 001500          MOV      #DM.MAP,R4 ;SET POINTER
1124 002364 010437 001246    5$: MOV      R4,TEMP1  ;SET ADDRESS
1125 002370 012437 001250          MOV      (R4)+,TEMP2 ;SET CSR
1126 002374 001411          BEQ      1$      ;ALL DONE IF ZERO
1127 002376 012437 001252          MOV      (R4)+,TEMP3 ;SET STAT1
1128 002402 012437 001254          MOV      (R4)+,TEMP4 ;SET STAT2
1129 002406 012437 001256          MOV      (R4)+,TEMP5 ;SET STAT3
1130 002412 104410          CONVPT          ;TYPE OUT STATUS MAP
1131 002414 007454          XSTATQ          ;
1132 002416 000762          BR           5$
1133 002420 012700 001500    1$: MOV      #DM.MAP,R0 ;R0 POINTS TO STATUS TABLE
1134
1135 *****
1136 ;*AUTO SIZE TEST
1137 ;*THIS TEST VERIFYS THAT THE DMC11S AND/OR KMC11S ARE AT THE CORRECT FLOATING
1138 ;*ADDRESSES FOR YOUR SYSTEM. IF THIS TEST FAILS, IT IS NOT A HARDWARE ERROR.
1139 ;*CHECK THE ADDRESSES OF ALL FLOATING DEVICES (DJ,DH,DQ,DU,DUP,LK,DMC,DZ,KMC).
1140 ;*IF THERE ARE NO OTHER FLOATING DEVICES BEFORE THE DMC11, THE FIRST
1141 ;*DMC11 ADDRESS IS 760070, KMC11 IS 760110. NO DEVICE SHOULD EVER BE AT
1142 ;*ADDRESS 760000. THIS TEST MAY REQUIRE 2 OR MORE ATTEMPTS TO GET THE
1143 ;*RIGHT ADDRESSES. AFTER YOU HAVE CHANGED THE ADDRESS TO WHAT IT TOLD
1144 ;*YOU THE FIRST TIME, IT MAY COME BACK AND TELL YOU A DIFFERENT ADDRESS
1145 ;*THE NEXT TIME YOU RUN IT. PLEASE HAVE PATIENCE, THE FINAL ADDRESS
1146 ;*WILL BE CORRECT (AS LONG AS ALL DEVICES IN FRONT OF THE DMC'S ARE
1147 ;*CORRECT).
1148 *****
1149
1150 002424 013746 000004          MOV      @#4,-(SP) ;SAVE LOC 4
1151 002430 013746 000006          MOV      @#6,-(SP) ;SAVE LOC 6
1152 002434 005037 000006          CLR      @#6      ;CLEAR VEC+2
1153 002440 005037 001252          CLR      TEMP3    ;CLEAR FLAG
1154 002444 005005          CLR      R5            ;R5=0 DMC, R5 -1 KMC
1155 002446 011037 001404    AUSTRT: MOV      (R0),DMCSR ;GET NEXT DMC SR
1156 002452 001564          BEQ      AUDONE    ;BR IF DONE
1157 002454 005705          TST      R5        ;DMC OR KMC?
1158 002456 001005          BNE      1$      ;BR IF KMC
1159 002460 032760 100000 000002 BIT      #BIT15,2(R0) ;CHECK FOR DMC CSR
1160 002466 001061          BNE      SKIP      ;SKIP IF NOT DMC
1161 002470 000404          BR           2$      ;ITS A DMC SO CONTINUE
1162 002472 032760 100000 000002 1$: BIT      #BIT15,2(R0) ;CHECK FOR KMC CSR
1163 002500 001454          BEQ      SKIP      ;SKIP IF NOT KMC
1164 002502 012737 002674 000004 2$: MOV      #NODEV,@#4 ;SET UP FOR TIMEOUT
1165 002510 005705          TST      R0        ;DMC OR KMC?
1166 002512 001003          BNF      3$      ;BR IF KMC
```

CZDMC MACY11 30A(1052) 24-AUG-78 12:19 PAGE 25
 CZDMCC.P11 11-AUG-78 08:21 PROGRAM INITIALIZATION AND START UP.

SEQ 0025

```

1167 002514 012703 000006      MOV      #6,R3      ;R3 IS COUNT OF DEVICES BEFORE DMC
1168 002520 000402      BR      4$      ;GO ON
1169 002522 012703 000010      3$: MOV      #10,R3     ;R3 IS COUNT OF DEVICES BEFORE KMC
1170 002526 012702 003010      4$: MOV      #DEVTAB,R2   ;R2 IS DEVICE TABLE POINTER
1171 002532 012701 160010      MOV      #160010,R1  ;START WITH ADDRESS 160010
1172 002536 005711      FLOAT: TST      (R1)      ;CHECK ADDRESS IN R1
1173 002540 111204      MOV      (R2),R4     ;IF NO TIMEOUT, GET NEXT ADDRESS
1174 002542 060401      ADD      R4,R1      ;IN R1
1175 002544 005201      INC      R1
1176 002546 040401      BIC      R4,R1
1177 002550 005703      TST      R3
1178 002552 001371      BNE      FLOAT      ;ANY MORE DEVICES TO CHECK FOR?
1179 002554 012737 002700 000004  MOV      #ERR,24$     ;BR IF YES
1180 002562 010137 003022      MOV      R1,XLOC     ;OK ONLY DMC'S ARE LEFT, SET UP FOR TIMEOUT
1181 002566 005705      FY:  TST      R5      ;SAVE FIRST DMC/KMC ADDRESS
1182 002570 001005      BNE      1$      ;DMC OR KMC?
1183 002572 032760 100000 000002  BIT      #BIT15,2(R0) ;BR IF KMC
1184 002600 001014      BNE      SKIP      ;CHECK FOR DMC CSR
1185 002602 000404      BR      2$      ;SKIP IF NOT DMC
1186 002604 032760 100000 000002  1$: BIT      #BIT15,2(R0) ;ITS A DMC SO CONTINUE
1187 002612 001407      BEQ      SKIP      ;CHECK FOR KMC CSR
1188 002614 005711      2$:  TST      (R1)      ;SKIP IF NOT KMC
1189 002616 020137 001404      CMP      R1,DMCSR    ;CHECK DMC ADDRESS
1190 002622 001411      BEQ      OK      ;DOES IT MATCH
1191 002624 062701 000010      ADD      #10,R1     ;BR IF YES
1192 002630 000756      BR      FY      ;GET NEXT DMC ADDRESS
1193 002632 062700 000010      SKIP: ADD      #10,R0   ;DO IT AGAIN
1194 002636 011037 001404      MOV      (R0),DMCSR  ;SKIP TO NEXT CSR IN TABLE
1195 002642 001470      BEQ      AUDONE    ;GET NEXT CSR
1196 002644 000750      BR      FY      ;BR IF DONE
1197 002646 062700 000010      OK:  ADD      #10,R0   ;ELSE CONTINUE
1198 002652 062737 000010 003022  ADD      #10,XLOC     ;SKIP TO NEXT DMC CSR
1199 002660 011037 001404      MOV      (R0),DMCSR  ;UPDATE EXPECTED DMC/KMC ADDRESS
1200 002664 001457      BEQ      AUDONE    ;GET NEXT DMC/KMC CSR
1201 002666 013701 003022      MOV      XLOC,R1   ;BR IF DONE
1202 002672 000735      BR      FY      ;GET EXPECTED DMC/KMC ADDRESS
1203 002674 122243      NODEV: CMP      (R2)+,-(R3) ;CONTINUE
1204 002676 000002      RTI      ;ON TIMEOUT, INC R2, DEC R3
1205 002700 005737 001252      ERR:  TST      TEMP3     ;RETURN
1206 002704 001014      BNE      1$      ;CHECK FLAG IF - 0 TYPE HEADER
1207 002706 104402      TYPE      ;SKIP HEADER
1208 002710 007223      CONERR  ;TYPEOUT HEADER MESSAGE
1209 002712 012737 002700 001276  MOV      #ERR,SAVPC  ;CONFIGURATION ERROR!...
1210 002720 104411      CNVRT      ;SAVE PC FOR TYPEOUT
1211 002722 002770      ERRPC      ;TYPE OUT ERROR PC
1212 002724 104402      TYPE      ;
1213 002726 007277      CNERR      ;TYPE REST OF HEADER
1214 002730 012737 177777 001252  1$: MOV      #-1,TEMP3   ;
1215 002736 010137 001262      MOV      R1,SAVR1     ;SET FLAG SO IT ONLY GETS TYPED ONCE
1216 002742 104410      CNVRT      ;SAVE R1 FOR TYPEOUT
1217 002744 002776      CONTAB      ;
1218 002746 005705      TST      R5      ;TYPE CSR VALUES
1219 002750 001003      BNE      3$      ;DMC OR KMC ?
1220 002752 104402      TYPE      ;BR IF KMC
1221 002754 007320      DMCM      ;
1222 002756 000402      BR      4$      ;CONTINUE

```

CZDMC MACY11 30A(1052) 24-AUG-78 12:19 PAGE 26
CZDMCC.P11 11-AUG-78 08:21

PROGRAM INITIALIZATION AND START UP.

SEQ 0026

1223	002760	104402		3\$:	TYPE			
1224	002762	007330			KMCM			
1225	002764	022626		4\$:	(MP	(SP)+,(SP)+		;ADJUST STACK
1226	002766	000727			BR	OK		;BR TO GET OUT
1227	002770	000001		ERRPC:	1			
1228	002772	006	002		.BYTE	6,2		
1229	002774	001276			SAVPC			
1230	002776	000002		CONTAB:	2			
1231	003000	006	004		.BYTE	6,4		
1232	003002	003022			XLOC			
1233	003004	006	002		.BYTE	6,2		
1234	003006	001404			DMCSR			
1235	003010	007		DEVTAB:	.BYTE	7		;DJ
1236	003011	017			.BYTE	17		;DH
1237	003012	007			.BYTE	7		;DQ
1238	003013	007			.BYTE	7		;DU
1239	003014	007			.BYTE	7		;DUP
1240	003015	007			.BYTE	7		;LK
1241	003016	007			.BYTE	7		;DMC
1242	003017	007			.BYTE	7		;DZ
1243	003020	007			.BYTE	7		;KMC
1244		003022		.EVEN				
1245	003022	000000		XLOC:	0			
1246	003024	005705		AUDONE:	TST	R5		;DMC?
1247	003026	001005			BNE	1\$		;BR IF KMC AND ALL DONE
1248	003030	012705	177777		MOV	#-1,R5		;SET R5 TO -1 (KMC)
1249	003034	012700	001500		MOV	#DM.MAP,R0		;RESET R0 TO START OF TABLE
1250	003040	000602			BR	AUSTR		;GO DO KMC'S
1251	003042	012637	000006	1\$:	MOV	(SP)+, @#6		;RESTORE LOC 6
1252	003046	012637	000004		MOV	(SP)+, @#4		;RESTORE LOC 4
1253	003052	032737	000010	001236	BIT	#SW03,STRTSW		;SELECT SPECIFIC DEVICES??
1254	003060	001422			BEQ	3\$		;BR IF NO.
1255	003062	104402	006144		TYPE	,MNEW		;TYPE THE MESSAGE.
1256	003066	005000			CLR	R0		;ZERO DATA LIGHTS
1257	003070	000000			HALT			;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
1258	003072	027737	176104	001312	CMP	@SWR,SAVACT		;IS THE NUMBER VALID?
1259	003100	101404			BLOS	2\$		;BR IF NUMBER IS OK.
1260	003102	101402	006005		TYPE	,MERR3		;TELL USER OF INVALID NUMBER.
1261	003106	000000			HALT			;STOP EVERY THING.
1262	003110	000776			BR	.-2		;RESTART THE PROGRAM AGAIN.
1263	003112	017737	176064	001306	2\$:	MOV	@SWR,DMACTV	;GET NEW DEVICE PATTERN
1264	003120	013700	001306		MOV	DMACTV,R0		;SHOW THE USER WHAT HE SELECTED.
1265	003124	000000			HALT			;CONTINUE DYNAMIC SWITCHES.
1266	003126	012700	000300		3\$:	MOV	#300,R0	;PREPARE TO CLEAR THE FLOATING
1267	003132	012701	000302		MOV	#302,R1		;VECTOR AREA. 300-776
1268	003136	010120		4\$:	MOV	R1,(R0)+		;START PUTTING 'PC+2 - HALT'
1269	003140	005021			CLR	(R1)+		;IN VECTOR AREA.
1270	003142	022021			CMP	(R0)+,(R1)+		;POP POINTERS
1271	003144	022700	001000		CMP	#1000,R0		;ALL DONE??
1272	003150	001372			BNE	4\$		;BR IF NO.
1273								
1274								
1275								
1276								
1277	003152	012706	001200		.BEGIN:	MOV	#STACK,SP	;SET UP STACK
1278	003156	013746	000006		MOV	@#6,-(SP)		;SAVE LOC 6

;TEST START AND RESTART
:-----

1279	003162	013746	000004		MOV	@#4,-(SP)	:SAVE LOC 4	
1280	003166	005000			CLR	R0	:START AT 0	
1281	003170	012737	003234	000004	MOV	#2\$,@#4	:SET UP FOR TIME OUT	
1282	003176	005037	000006		CLR	@#6	:TO AUTOSIZE MEMORY	
1283	003202	005720		6\$:	TST	(R0)+	:CHECK ADDRESS IN R0	
1284	003204	022700	157776		CMP	#157776,R0	:IS IT AT LEAST 28K	
1285	003210	001374			BNE	6\$	:BR IF NO	
1286	003212	162700	007776		SUB	#7776,R0	:SAVE 2K FOR MONITORS	
1287	003216	010037	001304	7\$:	MOV	R0,MEMLIM	:STORE MEMORY LIMIT	
1288	003222	012637	000004		MOV	(SP)+,@#4	:RESTORE LOC 4	
1289	003226	012637	000006		MOV	(SP)+,@#6	:RESTORE LOC 6	
1290	003232	000413			BR	10\$	:CONTINUE	
1291	003234	022626		2\$:	CMP	(SP)+,(SP)+	:ADJUST STACK	
1292	003236	162700	000004		SUB	#4,R0	:GET LAST GOOD ADDRESS	
1293	003242	162700	007776		SUB	#7776,R0	:SAVE 2K FOR MONITORS	
1294	003246	022700	030000		CMP	#30000,R0	:IS IT 8K?	
1295	003252	001361			BNE	7\$	:BR IF NO	
1296	003254	012700	057400		MOV	#37400,R0	:IF 8K DON'T SAVE 2K	
1297	003260	000756			BR	7\$	:	
1298	003262	012737	000340	177776	10\$:	MOV	#340,PS	:LOCK OUT INTERRUPTS
1299	003270	032737	000004	001236		BIT	#BIT2,STR*SW	:CHECK FOR LOCK ON TEST
1300	003276	001411			BEQ	1\$	:BR IF NO LOCK DESIRED.	
1301	003300	104402	006043		TYPE	,MLOCK	:TYPE LOCK SELECTED.	
1302	003304	012737	000240	003612	MOV	#NOP,TTST	:ADJUST SCOPE ROUTINE.	
1303	003312	012737	000240	003614	MOV	#NOP,TTST+2	:SET UP TO LOCK	
1304	003320	000406			BR	3\$	:CONTINUE ALONG.	
1305	003322	013737	003730	003612	1\$:	MOV	BRW,TTST	:PREPARE NORMAL SCOPE ROUTINE
1306	003330	013737	003732	003614		MOV	BRX,TTST+2	:LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
1307	003336	012737	010060	001214	3\$:	MOV	#CYCLE,RETURN	:START AT 'CYCLE' FIND WHICH DEVICE TO TEST
1308	003344	032737	000002	001236	4\$:	BIT	#SW01,STR*SW	:IS TEST NO. SELECTED?
1309	003352	001002			BNE	5\$	:BR IF YES	
1310	003354	104402	005755		TYPE	,MR	:TYPE R	
1311	003360	000177	175630	5\$:	JMP	@RETURN	:START TESTING	

```

1312                                     ;END OF PASS
1313                                     ;TYPE NAME OF TEST
1314                                     ;UPDATE PASS COUNT
1315                                     ;CHECK FOR EXIT TO ACT-11
1316                                     ;RESTART TEST
1317
1318 003364 000005                      .EOP: RESET                      ;MAKE THE WORLD CLEAN AGAIN.
1319 003366 005037 001234              CLR      LSTERR                  ;CLEAR LAST ERROR PC
1320 003372 105037 001325              CLR      ERRFLG                 ;CLEAR ERROR FLAG
1321 003376 005237 001230              INC      PASCNT                  ;UPDATE PASS COUNT
1322 003402 013777 001230 175570      MOV      PASCNT,@DISPLAY        ;DISPLAY PASS COUNT
1323 003410 104402 005733              TYPE     ,MEPASS                ;TYPE END PASS
1324 003414 104402 006072              TYPE     ,MCSRX                 ;TYPE CSR
1325 003420 104411 003546              CNVRT    ,XCSR                  ;SHOW IT
1326 003424 104402 006100              TYPE     ,MVECX                 ;TYPE VECTOR
1327 003430 104411 003554              CNVRT    ,XVEC                  ;SHOW IT
1328 003434 104402 006106              TYPE     ,MPASSX                ;TYPE PASSES
1329 003440 104411 003562              CNVRT    ,XPASS                 ;SHOW IT
1330 003444 104402 006117              TYPE     ,MERRX                 ;TYPE ERRORS
1331 003450 104411 003570              CNVRT    ,XERR                  ;SHOW IT
1332 003454 013700 001322              MOV      MILK,R0                ;GET POINTER TO PASS COUNT
1333 003460 013720 001230              MOV      PASCNT,(R0)+          ;STORE PASS COUNT FOR THIS DMC11
1334 003464 013720 001232              MOV      ERRCNT,(R0)+          ;STORE ERROR COUNT FOR THIS DMC11
1335 003470 005337 001314              DEC      SAVNUM                 ;ARE ALL DEVICES TESTED?
1336 003474 001017                      BNE      RESTRT                 ;BR IF NO.
1337 003476 112737 000377 001327      MOV      #377,QV.FLG           ;SET THE QUICK VERIFY FLAG.
1338 003504 013737 001310 001314      MOV      DMNUM,SAVNUM           ;RESTORE THE COUNT
1339 003512 013701 000042              MOV      @#42,R1              ;CHECK FOR ACT-11 OR DDP
1340 003516 001406                      BEQ      RESTRT                 ;IF NOT, CONTINUE TESTING
1341 003520 000005                      RESFT                          ;STOP THE SHOW--CLEAR THE WORLD
1342 003522
1343 003522 004711                      $ENDAD: JSR      PC,(R1)
1344 003524 000240                      NOP
1345 003526 000240                      NOP
1346 003530 000240                      NOP
1347 003532 000240                      NOP
1348 003534 012737 010060 001214      RESTRT: MOV      #CYCLE,RETURN
1349 003542 000137 010060              JMP      CYCLE
1350 003546 000001                      XCSR:  1
1351 003550 006 002                      .BYTE  6,2
1352 003552 001404                      DMCSTR
1353 003554 000001                      XVEC:  1
1354 003556 004 002                      .BYTE  4,2
1355 003560 001374                      DMRVEC
1356 003562 000001                      XPASS:  1
1357 003564 006 002                      .BYTE  6,2
1358 003566 001230                      PASCNT
1359 003570 000001                      XERR:  1
1360 003572 006 002                      .BYTE  6,2
1361 003574 001232                      ERRCNT
1362
1363                                     ;SCOPE LOOP AND INTERATION HANDLER
1364                                     ;-----
1365
1366 003576 004737 007606              .SCOPE: JSR      PC,CKSWR          ;CHECK FOR SOFT SWR
1367 003602 010016                      MOV      R0,(SP)              ;SAVE R0 ON THE STACK
  
```



```

1368 003604 032777 040000 175370      BIT      #BIT14,@SWR      ;'LOOP ON THIS TEST'?
1369 003612 001407      TTST:  BEQ      1$              ;BR IF NO. (IF LOCK SW01=1; THIS LOC -240)
1370 003614 000437      BR      3$              ;GOTO 3$ (IF LOCK SW01=1; THIS LOC -240)
1371 003616 005737 003734      TST      DONE          ;WAS TKCSR DONE SET?
1372 003622 001434      BEQ      3$              ;BR IF NO (LOCKED ON TEST)
1373 003624 005037 003734      CLR      DONE          ;YES, CLEAR FLAG
1374 003630 000415      BR      2$              ;GO TO NEXT TEST
1375 003632 032777 004000 175342 1$:  BIT      #SW11,@SWR      ;DELETE ITERATION? (QUICK PASS)
1376 003640 001011      BNE      2$              ;BR IF YES
1377 003642 105737 001327      TSTB     QV.FLG        ;HAVE PASSES BEECOMPLETED?
1378 003646 001406      BEQ      2$              ;BR IF QUICK PASS.
1379 003650 005237 001224      INC      LPCNT         ;UPDATE ITERATION COUNTER
1380 003654 023737 001224 001222      CMP      LPCNT,I COUNT    ;ARE ALL ITERATIONS DONE??
1381 003662 101414      BLOS     3$              ;BR IF NOT YET
1382 003664 105037 001325      CLR      ERRFLG        ;PREPARE FOR NEW TEST
1383 003670 005037 001224      CLR      LPCNT         ;START ICOUNTER AT 0
1384 003674 005037 001220      CLR      LOCK
1385 003700 012737 000020 001222      MOV      #20,I COUNT    ;RESET ITERATIONS
1386 003706 013737 001216 001214      MOV      NEXT,RETURN    ;GET NEXT TEST
1387 003714 011600      3$:  MOV      (SP),R0        ;POP R0 OFF OF THE STACK
1388 003716 022626      POP2SP     ;FAKE AN 'RTI'
1389 003720 013701 001404      MOV      DMCSR,R1        ;R1 CONTAINS BASE DMC ADDRESS
1390 003724 000177 175264      JMP      @RETURN      ;GO DO THE TEST
1391 003730 001407      BRW:    1407
1392 003732 000437      BRX:    437
1393 003734 000000      DONE:   0
1394
1395      ;CHECK FOR FREEZE ON CURRENT DATA
1396      ;-----
1397
1398 003736 004737 007606      .SCOPE1: JSR      PC,CKSWR      ;CHECK FOR SOFT SWR
1399 003742 032777 001000 175232      BIT      #SW09,@SWR      ;IS SW09=1(SET)?
1400 003750 001405      BEQ      1$              ;BR IF NOT SET.
1401 003752 005737 001220      TST      LOCK
1402 003756 001402      BEQ      1$
1403 003760 013716 001220      MOV      LOCK,(SP)      ;GOTO THE ADDRESS IN LOCK.
1404 003764 000002      1$:  RTI              ;GO BACK.
1405
1406      ;TELETYPE OUTPUT ROUTINE
1407      ;-----
1408
1409 003766 010546      .TYPE:  MOV      R5,-(SP)        ;SAVE R5 ON THE STACK.
1410 003770 017605      MOV      @2(SP),R5        ;GET ADDRESS OF MESSAGE.
1411 003774 062766 000002 000002      ADD      #2,2(SP)        ;POP OVER ADDRESS.
1412 004002 005737 010016      4$:  TST      SWFLG        ;SOFT SWR MESSAGE?
1413 004006 001004      BNE      1$              ;IF YES TYPE IT OUT REGARDLESS OF SW12
1414 004010 032777 010000 175154      BIT      #SW12,@SWR      ;INHIBIT ALL PRINT OUT??
1415 004016 001012      BNE      3$              ;BR IF NO PRINT OUT WANTED (SW12-1)
1416 004020 105715      1$:  TSTB     (R5)          ;IS NUMBER MINUS? (MSB 1(BIT7))
1417 004022 100002      BPL      2$              ;BR IF NUMBER IS PLUS
1418 004024 104402 005672      TYPE     ,MCRLF        ;TYPE A CR/LF.
1419 004030 105777 175154      2$:  TSTB     @TPCSR        ;TTY READY?
1420 004034 100375      BPL      2$              ;BR IF NO.
1421 004036 112577 175150      MOV      (R5)+,@TPDBR    ;PRINT CURRENT CHAR.
1422 004042 001357      BNE      4$              ;IF NOT ZERO KEEP PRINTING
1423 004044 012605      3$:  MOV      (SP)+,R5        ;END OF OUTPUT. RESTORE R5

```



```

1424 004046 000002          RTI          ;GO HOME
1425                      ;-----
1426
1427 004050 010346          .INSTR: MOV     R3,-(SP)          ;SAVE R3 ON STACK
1428 004052 010446          MOV     R4,-(SP)          ;SAVE R4 ON STACK
1429 004054 017637 000004 004072  MOV     @4(SP),.MSG
1430 004062 062766 000002 000004  ADD     #2,4(SP)
1431 004070 104402          .INST1: TYPE
1432 004072 000000          .MSG: 0
1433 004074 012704 007502  MOV     #INBUF,R4
1434 004100 012703 000007  MOV     #7,R3
1435 004104 105777 175074  1$:  TSTB    @TKCSR
1436 004110 100375          BPL      1$
1437 004112 117714 175070  MOVB    @TKDBR,(R4)
1438 004116 142714 000200  BICB    #200,(R4)
1439 004122 122427 000015  (MPB    (R4)+,#15
1440 004126 001417          BEQ     INSTR2
1441 004130 105777 175054  2$:  TSTB    @TPCSR
1442 004134 100375          BPL      2$
1443 004136 017777 175044 175046  MOV     @TKDBR,@TPDBR
1444 004144 005303          DEC     R3
1445 004146 001356          BNE     1$
1446 004150 012604          MOV     (SP)+,R4
1447 004152 012603          MOV     (SP)+,R3
1448 004154 104402 005666  .INSTE: TYPE    ,MQM
1449 004160 010346          MOV     R3,-(SP)
1450 004162 010446          MOV     R4,-(SP)
1451 004164 000741          BR      .INST1
1452 004166 012604  INSTR2: MOV     (SP)+,R4          ;RESTORE R4
1453 004170 012603          MOV     (SP)+,R3          ;RESTORE R3
1454 004172 000002          RTI
1455
1456                      ;CONVERT ASCII STRING TO OCTAL
1457                      ;-----
1458
1459 004174 010546          .PARAM: MOV     R5,-(SP)
1460 004176 010446          MOV     R4,-(SP)
1461 004200 016605 000004  MOV     4(SP),R5
1462 004204 012537 004364  MOV     (R5)+,LOLIM
1463 004210 012537 004366  MOV     (R5)+,HILIM
1464 004214 012537 004370  MOV     (R5)+,DEVADR
1465 004220 112537 004372  MOVB    (R5)+,LOBITS
1466 004224 112537 004373  MOVB    (R5)+,ADRCNT
1467 004230 010566 000004  MOV     R5,4(SP)
1468 004234 005005  PARAM1: CLR     R5
1469 004236 012704 007502  MOV     #INBUF,R4
1470 004242 122714 000015  (MPB    #15,(R4)
1471 004246 001420          BEQ     PARERR
1472 004250 121427 000060  1$:  (MPB    (R4),#60
1473 004254 002415          BLT     PARERR
1474 004256 121427 000067  (MPB    (R4),#67
1475 004262 003012          BGT     PARERR
1476 004264 142714 000060  BICB    #60,(R4)
1477 004270 152405          BISB    (R4)+,R5
1478 004272 122714 000015  (MPB    #15,(R4)
1479 004276 001406          BEG     LIMITS

```

```

1480 004300 006305      ASL      R5
1481 004302 006305      ASL      R5
1482 004304 006305      ASL      R5
1483 004306 000760      BR        1$
1484 004310 004404      PARERR: INSTER
1485 004312 000750      BR        PARAM1
1486
1487                      ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
1488                      ;-----
1489
1490 004314 020537 004366  LIMITS: CMP      R5,HILIM
1491 004320 101373          BHI      PARERR
1492 004322 020537 004364  CMP      R5,LOLIM
1493 004326 103770          BLO      PARERR
1494 004330 133705 004372  B1'B    LOBITS,R5
1495 004334 001365          BNE      PARERR
1496
1497                      ;STORE NUMBER AT SPECIFIED ADDRESS
1498
1499 004336 013704 004370 1$:      MOV      DEVADR,R4
1500 004342 010524          MOV      R5,(R4)+
1501 004344 062705 000002      ADD      #2,R5
1502 004350 105337 004373      DECB    ADRCNT
1503 004354 001372          BNE      1$
1504 004356 012604          MOV      (SP)+,R4
1505 004360 012605          MOV      (SP)+,R5
1506 004362 000002          RTI
1507 004364 000000  LOLIM: 0
1508 004366 000000  HILIM: 0
1509 004370 000000  DEVADR: 0
1510 004372 000000  LOBITS: 0
1511                      ADRCNT=LOBITS+1
1512
1513                      ;SAVE PC OF TEST THAT FAILED AND R0-R5
1514                      ;-----
1515
1516 004374 016637 000004 001376 .SAV05: MOV      4(SP),SAVPC      ;SAVE R7 (PC)
1517
1518                      ;SAVE R0-R5
1519
1520 004402 010537 001272  SAV05: MOV      R5,SAVR5      ;SAVE R5
1521 004406 010437 001270      MOV      R4,SAVR4      ;SAVE R4
1522 004412 010337 001266      MOV      R3,SAVR3      ;SAVE R3
1523 004416 010237 001264      MOV      R2,SAVR2      ;SAVE R2
1524 004422 010137 001262      MOV      R1,SAVR1      ;SAVE R1
1525 004426 010037 001260      MOV      R0,SAVR0      ;SAVE R0
1526 004432 000002          RTI                      ;LEAVE.
1527
1528                      ;RESTORE R0-R5
1529
1530 004434 013700 001260  .RES05: MOV      SAVR0,R0      ;RESTORE R0
1531 004440 013701 001262      MOV      SAVR1,R1      ;RESTORE R1
1532 004444 013702 001264      MOV      SAVR2,R2      ;RESTORE R2
1533 004450 013703 001266      MOV      SAVR3,R3      ;RESTORE R3
1534 004454 013704 001270      MOV      SAVR4,R4      ;RESTORE R4
1535 004460 013705 001272      MOV      SAVR5,R5      ;RESTORE R5

```

```

1536 004464 000002 RTI ;LEAVE
1537
1538 ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
1539 -----
1540
1541 004466 104402 005672 .CONVR: TYPE ,MCRLF
1542 004472 010046 .CNVRT: MOV R0,-(SP)
1543 004474 010146 MOV R1,-(SP)
1544 004476 010346 MOV R3,-(SP)
1545 004500 010446 MOV R4,-(SP)
1546 004502 010546 MOV R5,-(SP)
1547 004504 017601 000012 MOV @12(SP),R1
1548 004510 062766 000002 000012 ADD #2,12(SP)
1549 004516 012137 004710 MOV (R1)+,WRDCNT
1550 004522 112137 004712 1: MOV (R1)+,CHRCNT
1551 004526 112137 004713 MOV (R1)+,SPACNT
1552 004532 013137 004714 MOV @ (R1)+,BINWRD
1553 004536 122737 000003 004712 CMPB #3,CHRCNT
1554 004544 001003 BNE 2$
1555 004546 042737 177400 004714 BIC #177400,BINWRD
1556 004554 013704 004714 2$: MOV BINWRD,R4
1557 004560 113705 004712 MOV (R1)+,CHRCNT
1558 004564 012700 001416 3$: MOV #TEMP,R0
1559 004570 010403 MOV R4,R3
1560 004572 042703 177770 BIC #177770,R3
1561 004576 062703 000060 ADD #060,R3
1562 004602 110320 MOV R3,(R0)+
1563 004604 000241 CLC
1564 004606 006004 ROR R4
1565 004610 000241 CLC
1566 004612 006004 ROR R4
1567 004614 000241 CLC
1568 004616 006004 ROR R4
1569 004620 005305 DEC R5
1570 004622 001362 BNE 3$
1571 004624 012703 007544 MOV #MDATA,R3
1572 004630 114023 4$: MOV (R0),(R3)+
1573 004632 105337 004712 DECB CHRCNT
1574 004636 001374 BNE 4$
1575 004640 105737 004713 TSTB SPACNT
1576 004644 001405 BEQ 6$
1577 004646 112723 000040 5$: MOV #040,(R3)+
1578 004652 105337 004713 DECB SPACNT
1579 004656 001373 BNE 5$
1580 004660 105013 6$: CLRB (R3)
1581 004662 104402 007544 TYPE ,MDATA
1582 004666 005337 004710 DEC WRDCNT
1583 004672 001313 BNE 1$
1584 004674 012605 MOV (SP)+,R5
1585 004676 012604 MOV (SP)+,R4
1586 004700 012603 MOV (SP)+,R3
1587 004702 012601 MOV (SP)+,R1
1588 004704 012600 MOV (SP)+,R0
1589 004706 000002 RTI
1590 004710 000000 WRDCNT: 0
1591 004712 000000 CHRCNT: 0

```

```

1592          004713      SPACNT=CHRCNT+1
1593 004714 000000      BINWRD: 0
1594
1595
1596          ;TRAP DISPATCH SERVICE
1597          ;ARGUMENT OF TRAP IS EXTRACTED
1598          ;AND USED AS OFFSET TO OBTAIN POINTER
1599          ;TO SELECTED SUBROUTINE
1600
1601 004716 011646      .TRPSR: MOV      (SP),-(SP)      ;GET PC OF RETURN
1602 004720 162716 000002      SUB      #2,(SP)      ;=PC OF TRAP
1603 004724 017616 000000      MOV      @ (SP),(SP)      ;GET TRP
1604 004730 006316      TRPOK: ASL      (SP)      ;MULTIPLY TRAP ARG BY 2
1605 004732 042716 177001      BIC      #177001,(SP)      ;CLEAR UNWANTED BITS
1606 004736 062716 001330      ADD      #.TRPTAB,(SP)      ;POINTER TO SUBROUTINE ADDRESS
1607 004742 017616 000000      MOV      @ (SP),(SP)      ;SUBROUTINE ADDRESS
1608 004746 000136      JMP      @ (SP)+      ;GO TO SUBROUTINE
1609
1610          ;ERROR HANDLER
1611          ;-----
1612
1613 004750 004737 007606      .HLT:  JSR      PC,CKSWR      ;CHECK FOR SOFT SWR
1614 004754 032777 010000 174220      BIT      #SW12,@SWR      ;BELL ON ERROR?
1615 004762 001406      BEQ      XBX      ;BR IF NO BELL
1616 004764 105777 174270      TSTB      @TPCSR      ;TTY READY.
1617 004770 100003      BPL      XBX      ;DON'T WAIT IF TTY NOT READY.
1618 004772 112777 000207 174212      MOVB      #20,@TPDBR      ;PUSH A BELL AT THE TTY.
1619 005000 032777 020000 174774 XBX:  BIT      #SW13,@SWR      ;DELETE ERROR PRINT OUT?
1620 005006 001105      BNE      HALTS      ;BR IF NO PRINT OUT WANTED.
1621 005010 021637 001234      LMP      (SP),LSTERR      ;WAS THIS ERROR FOUND LAST TIME?
1622 005014 001404      BEQ      1$      ;BR IF YES
1623 005016 011637 001234      MOV      (SP),LSTERR      ;RECORD BEING HERE
1624 005022 105037 001325      CLRB      ERRFLG      ;PREPARE HEADER
1625 005026 104406      1$:  SAV05      ;SAVE ALL PROC REGISTERS
1626 005030 011605      MOV      (SP),R5      ;GET THE PC OF ERROR
1627 005032 162705 000002      SUB      #2,R5      ;GET ADDRESS OF TRAP CALL
1628 005036 011504      MOV      (R5),R4      ;GET HLT INSTRUCTION
1629 005040 006304      ASL      R4      ;MULT BY TWO
1630 005042 061504      ADD      (R5),R4      ;DOUBLE IT
1631 005044 006304      ASL      R4      ;MULT AGAIN
1632 005046 042704 177001      BIC      #177001,R4      ;CLEAR JUNK
1633 005052 062704 036330      ADD      #.ERRTAB,R4      ;GET POINTER
1634 005056 012437 005172      MOV      (R4)+,ERRMSG      ;GET ERROR MESSAGE
1635 005062 012437 005204      MOV      (R4)+,DATAHD      ;GET DATA HEADRER
1636 005066 011437 005216      MOV      (R4),DATABP      ;GET DATA TABLE
1637 005072 105737 001325      TSTB      ERRFLG      ;TYPE HEADREER
1638 005076 001403      BEQ      TYPMSG      ;BR IF YES
1639 005100 005737 005216      TST      DATABP      ;DOES DATA TABLE EXIST?
1640 005104 001040      BNE      TYFDAT      ;BR IF YES.
1641 005106 104402 005672      TYPMSG: TYPE      ,MCRLF      ;
1642 005112 104402 005672      TYPE      ,MCRLF      ;
1643 005116 005737 001220      TST      LOCK      ;
1644 005122 001402      BEQ      1$      ;
1645 005124 104402 006142      TYPE      ,MASTEX      ;
1646 005130 104402 006130      TYPE      ,*STN      ;
1647 005134 104411 005330      TYPE      ,*STN      ;

```

```

1648 005140 104402 006217      TYPE      ,MERRPC      ;TYPE PC.
1649 005144 104411 005322      CNVRT      ,ERTABO      ;SHOW IT
1650 005150 104402 005672      TYPE      ,MCRLF      ;GIVE A CR/LF
1651 005154 112737 177777 001325  MOV      #-1,ERRFLG ;NO MORE HEADER UNLESS NO DATA TABLE.
1652 005162 005737 005172      TST        ERRMSG      ;IS THERE AN ERROR MESSAGE?
1653 005166 001402      BEQ        WRKO.FM      ;BR IF NO.
1654 005170 104402      TYPE      ;TYPE
1655 005172 000000      ERRMSG: 0      ;ERROR MESSAGE
1656 005174      WRKO.FM:      ;
1657 005174 005737 005204      TST        DATAHD     ;DATA HEADER?
1658 005200 001402      BEQ        TYPDAT     ;BR IF NO
1659 005202 104402      TYPE      ;TYPE
1660 005204 000000      DATAHD: 0      ;DATA HEADER
1661 005206 005737 005216      TYPDAT: TST        DATABP     ;DATA TABLE?
1662 005212 001402      BEQ        RESREG     ;BR IF NO.
1663 005214 104410      CNVRT      ;SHOW
1664 005216 000000      DATABP: 0      ;DATA TABLE
1665 005220 104407      RESREG: RES05      ;RESTORE PROC REGISTERS
1666 005222 022737 003522 000042  HALTS:  CMP        #SFNDAD,@#42 ;IF ACT-11 AUTOMATIC MODE, HALT..
1667 005230 001403      BEQ        1$        ;
1668 005232 005777 173744      TST        @SWR      ;HALT ON ERROR?
1669 005236 100005      BPL        EXITER     ;BR IF NO HALT ON ERROR
1670 005240 010046      1$:  PUSHRO      ;SAVE RO
1671 005242 016600 000002      MOV        2(SP),RO ;SHOW ERROR PC IN DATA LIGHTS
1672 005246 000000      HALT      ;HALT
1673 005250 012600      POPRO      ;GET RO
1674 005252 005237 001232      EXITER: INC        ERRCNT     ;UPDATE ERROR COUNT
1675 005256 032777 000400 173716  BIT        #SW08,@SWR ;GOTO TOP OF TEST?
1676 005264 001007      BNE        1$        ;BR IF YES
1677 005266 032777 002000 173706  BIT        #SW10,@SWR ;GOTO NEXT TEST?
1678 005274 001411      BEQ        2$        ;BR IF NO
1679 005276 013737 001216 001214  MOV        NEXT,RETURN ;SET FOR NEXT TEST
1680 005304 012706 001200      1$:  MOV        #STACK,SP ;RESET SP
1681 005310 013701 001404      MOV        DMCSR,R1 ;SET UP R1
1682 005314 000177 173674      JMP        @RETURN ;GOTO SPECIFIED TEST
1683 005320 000002      2$:  RTI        ;RETURN
1684 005322 000001      ERTABO: 1      ;
1685 005324      006      002      .BYTE      6,2
1686 005326 001276      SAVPC      ;
1687 005330 000001      XTSTN: 1      ;
1688 005332      003      002      .BYTE      3,2
1689 005334 001226      TSTNC      ;ENTER HERE ON POWER FAILURE
1690      ;-----
1691
1692
1693
1694 005336      .PFAIL:
1695 005336 012737 005350 000024  MOV        #RESTART,24 ;SET UP FOR POWER UP TRAP
1696 005344 000000      HALT      ;HALT ON POWER DOWN NORMAL
1697 005346 000777      BR        .
1698
1699      ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
1700
1701 005350      RESTART:
1702 005350 012737 005336 000024  MOV        #PFAIL,24 ;SET UP FOR POWER FAILURE
1703 005356 012706 001200      MOV        #STACK,SP ;RESET THE STACK POINTER

```

```

1704 005362 013701 001404      MOV    DMCSR,R1      ;RESTORE R1
1705 005366 005037 001416      CLR    TEMP        ;READY FOR TIMMER
1706 005372 005237 001416      INC    TEMP        ;PLUS ONE TO THE TIMER.
1707 005376 001375              BNE     .-4            ;BR IF MORE TO GO
1708 005400 104402 005675      TYPE   ,MPFAIL      ;TYPE THE MESSAGE
1709 005404 104411 005430      CNVRT  ,PFTAB      ;TELL WHAT TEST TO RETURN TO.
1710 005410 105037 001325      CLR    ERRFLG      ;START CLEAN
1711 005414 005037 001234      CLR    LSTERR      ;.....
1712 005420 005011              CLR    (R1)          ;CLEAR MAINT BITS
1713 005422 104412              MSTCLR ;START CLEAN UP OF DEVICE
1714 005424 000177 173564      JMP     @RETURN    ;START DOING THAT TEST AGAIN.
1715 005430 000001              PFTAB: 1
1716 005432 003      002      .BYTE  3,2
1717 005434 001226              .ISTNO
1718
1719 005436              .DELAY:
1720 005436 012777 000020 173746      MOV    #20,@DMP04
1721 005444 104414              ROMCLK 121111      ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
1722 005446 121111              1$:              ;POKE CLOCK DELAY BIT
1723 005450
1724 005450 104414              ROMCLK 121224      ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
1725 005452 121224              1$:              ;PORT4 IBUS*11
1726 005454 032777 000020 173730      BIT    #BIT4,@DMP04 ;IS CLOCK BIT SET?
1727 005462 001772              BEQ     1$          ;BR IF NO
1728 005464 000002              RTI
1729
1730 005466              .MSTCLR:
1731 005466 152777 000100 173712      BISB   #BIT6,@DMCSRH ;SET MASTER CLEAR
1732 005474 142777 000300 173704      BICB   #BIT6!BIT7,@DMCSRH ;CLEAR MASTER CLEAR AND RUN
1733 005502 000002              RTI              ;RETURN
1734
1735 005504              .ROMCLK:
1736 005504 152777 000002 173674      BISB   #BIT1,@DMCSRH ;SET ROMI
1737 005512 013677 173676      MOV    @ (SP)+,@DMP06 ;LOAD INSTRUCTION IN SEL6
1738 005516 062746 000002              ADD    #2,-(SP)      ;ADJUST STACK
1739 005522 032777 000100 173452      BIT    #SW06,@SWR    ;HALT IF SW06 =1
1740 005530 001401              BEQ     1$          ;BR IF SW06 =0
1741 005532 000000              HALT              ;HALT BEFORE CLOCKING INSTRUCTION
1742 005534 152777 000003 173644      1$:    BISB   #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
1743 005542 142777 000007 173636      BICB   #BIT2!BIT1.BIT0,@DMCSRH ;CLEAR ROM0, ROM1, STEP
1744 005550 000002              RTI
1745
1746 005552              .DATACLK:
1747 005552 013637 001416      MOV    @ (SP)+,TEMP ;PUT TICK COUNT IN TEMP
1748 005556 062746 000002              ADD    #2,-(SP)      ;ADJUST STACK
1749 005562 152777 000020 173616      1$:    BISB   #BIT4,@DMCSRH ;SET STEP LU
1750 005570 027777 173610 173606      CMP    @DMCSR,@DMCSR ;WASTE TIME
1751 005576 142777 000020 173602      BICB   #BIT4,@DMCSRH ;CLEAR STEP LU
1752 005604 005337 001416      DEC    TEMP        ;DEC TICK COUNT
1753 005610 001364              BNE     1$          ;BR IF NOT DONE
1754 005612 000002              RTI              ;RETURN
1755 005614 000001      3$:    .BLKW 1
1756
1757 005616              .TIMER:
1758 005616 013637 001416      MOV    @ (SP)+,TEMP ;MOVE COUNT TO TEMP
1759 005622 062746 000002              ADD    #2,-(SP)      ;ADJUST STACK

```



```
1760 005626 1$:  
1761 005626 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
1762 005630 021364 021364 ;PORT4 IBUS* REG11  
1763 005632 032777 000002 173552 BIT #2, @DMP04 ;IS PGM CLOCK BIT CLEAR?  
1764 005640 001772 BEQ 1$ ;BR IF YES  
1765 005642 2$:  
1766 005642 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304  
1767 005644 021364 021364 ;PORT4 IBUS* REG11  
1768 005646 032777 000002 173536 BIT #2, @DMP04 ;IS PGM CLOCK BIT SET?  
1769 005654 001372 BNE 2$ ;BR IF YES  
1770 005656 005337 001416 DEC TEMP ;DEC COUNT  
1771 005662 001361 BNE 1$ ;BR IF NOT DONE  
1772 005664 000002 RTI ;RETURN  
1773  
1774 005666 020040 000077 MQM: .ASCIIZ / ?/  
(2) 005672 005015 000 MCRLF: .ASCIIZ <15><12>  
(2) 005675 377 053520 020122 MPFAIL: .ASCIIZ <377>/PWR FAILED. RESTART AT TEST /  
(2) 005733 377 047105 020104 MEPASS: .ASCIIZ <377>/END PASS CZDML /  
(2) 005755 377 000122 MR: .ASCIIZ <377>/R/  
(2) 005760 047377 020117 042504 MERR2: .ASCIIZ <377>/NO DEVICES PRESENT./  
(2) 006005 377 047111 052523 MERR3: .ASCIIZ <377>/INSUFFICIENT DATA./  
(2) 006031 377 042524 052123 MTSTPC: .ASCIIZ <377>/TEST PC-/  
(2) 006043 377 047514 045503 MLOCK: .ASCIIZ <377>/LOCK ON SELECTED TEST/  
(2) 006072 051503 035122 000040 MCSRX: .ASCIIZ /CSR: /  
(2) 006100 042526 035103 000040 MVECX: .ASCIIZ /VEC: /  
(2) 006106 040520 051523 051505 MPASSX: .ASCIIZ /PASSES: /  
(2) 006117 105 051122 051117 MERRX: .ASCIIZ /ERRORS: /  
(2) 006130 042524 052123 047040 MTSTN: .ASCIIZ /TEST NO: /  
(2) 006142 000052 MASTER: .ASCIIZ /*/  
(2) 006144 051777 052105 051440 MNEW: .ASCIIZ <377>/SET SWITCH REG TO DMC11'S DESIRED ACTIVE./  
(2) 006217 120 035103 000040 MERRPC: .ASCIIZ /PC: /  
(2) 006224 020212 020040 020040 XHEAD: .ASCIIZ <212>/  
(2) 006263 377 020040 020040 .ASCIIZ <377>/  
(2) 006322 020212 050040 020103 .ASCIIZ <212>/ PC CSR STAT1 STAT2 STAT3/  
(2) 006374 026777 026455 026455 .ASCIIZ <377>/-----  
(2) 006450 044377 053517 046440 NUM: .ASCIIZ <377>/HOW MANY DMC11'S TO BE TESTED?/  
(2) 006510 041777 051123 040440 CSR: .ASCIIZ <377>/CSR ADDRESS?/  
(2) 006526 053377 041505 047524 VEC: .ASCIIZ <377>/VECTOR ADDRESS?/  
(2) 006547 377 051102 050040 PRIO: .ASCIIZ <377>/BR PRIORITY LEVEL? (4,5,6,7)?/  
(2) 006606 044777 020106 046504 CRAM: .ASCIIZ <377>/IF DMC HAS CRAM (M8204) TYPE 'Y', IF CROM (M8200) TYPE 'N'?/  
(2) 006704 053777 044510 044103 MODU: .ASCIIZ <377>/WHICH LINE UNIT? IF NONE TYPE 'N', IF M8201 TYPE '1', IF M8202 TYPE  
(2) 007016 051777 044527 041524 LINE: .ASCIIZ <377>/SWITCH PAC#1 (DDCMP LINE #)?/  
(2) 007054 051777 044527 041524 BM: .ASCIIZ <377>/SWITCH PAC#2 (BM873 BOOT ADD)?/  
(2) 007114 044777 020123 044124 CONN: .ASCIIZ <377>/IS THE LOOP BACK CONNECTOR ON?/  
(2) 007154 047377 020117 042504 NOACT: .ASCIIZ <377>/NO DEVICES ARE SELECTED/  
(2) 007205 377 051412 051127 SWMES: .ASCIIZ <377><12>/SWR= /  
(2) 007215 116 053505 020077 SWMES1: .ASCIIZ /NEW? /  
(2) 007223 377 042377 041515 CONERR: .ASCIIZ <377><377>/DMC11 FOUND AT NON-STANDARD ADDRESS PC: /  
(2) 007277 377 054105 042520 CNERR: .ASCIIZ <377>/EXPECTED FOUND/  
(2) 007320 024040 046504 024503 DMCM: .ASCIIZ / (DMC) /  
(2) 007330 024040 046513 024503 KMCM: .ASCIIZ / (KMC) /  
(2) 007340 042377 041515 030461 SPEED: .ASCIIZ <377>/DMC11-AR(REMOTE,LOW SPEED) OR DMC11-AL(LOCAL,HIGH SPEED) TYPE 'R'  
(2) .EVEN  
(2) 007454 000005 XSTATQ: 5  
1775 007456 006 003 .BYTE 6,  
1776 007460 001246 TEMP1
```



```

1777 007462 006 003 .BYTE 6,3
1778 007464 001250 TEMP2
1779 007466 006 003 .BYTE 6,3
1780 007470 001252 TEMP3
1781 007472 006 003 .BYTE 6,3
1782 007474 001254 TEMP4
1783 007476 006 002 .BYTE 6,2
1784 007500 001256 TEMP5
1785 .EVEN
1786
1787 ;BUFFERS FOR INPUT-OUTPUT
1788
1789 007502 000000 INBUF: 0
1790 007544 007544 .+40
1791 007544 000000 MDATA: 0
1792 007606 .+40
1793
1794
1795 ;ROUTINE USED TO CHANGE SOFTWARE SWITCH
1796 ;REGISTER USING THE CONSOLE TERMINAL
1797 ;-----
1798
1799 007606 022737 000176 001202 CKSWR: CMP #SWREG,SWR ;IS THE SOFT SWR BEING USED?
1800 007614 001077 BNE CKSWR5 ;BR IF NO
1801 007616 105777 171362 TSTB @TKCSR ;IS DONE SET?
1802 007622 100003 BPL 2$ ;GO ON IF NOT SET
1803 007624 012737 177777 003734 MOV #1,DONE ;IF DONE SET, SET FLAG
1804 007632 022777 000007 171346 2$. CMP #7,@TKDDBR ;WAS CTRL G TYPED? (7 BIT ASCII)
1805 007640 001404 BEQ 1$ ;BR IF YES
1806 007642 022777 000207 171336 CMP #207,@TKDDBR ;WAS CTRL G TYPED? (8 BIT ASCII)
1807 007650 001061 BNE CKSWR5 ;BR IF NO
1808 007652 010246 1$: MOV R2,-(SP) ;STORE R2
1809 007654 010346 MOV R3,-(SP) ;STORE R3
1810 007656 010446 MOV R4,-(SP) ;STORE R4
1811 007660 012737 177777 010016 MOV #1,SWFLG ;SET SOFT TYPE OUT FLAG
1812 007666 005002 CKSWR1: CLR R2 ;CLEAR NEW SWR CONTENTS
1813 007670 012704 177777 MOV #1,R4 ;SET FLAG TO ALL ONES
1814 007674 104402 007205 TYPE ,SWMES ;TYPE 'SWR='
1815 007700 104411 CKSWR2: CNVRT ;TYPE OUT PRESENT CONTENTS
1816 007702 010052 SOFTSW ;OF SOFT SWITCH REGISTER
1817 007704 104402 007215 CKSWR3: TYPE ,SWMES1 ;TYPE 'NEW?'
1818 007710 004737 010020 CKSWR4: JSR PC,INCHAR ;GET RESPONSE
1819 007714 022703 000015 CMP #15,R3 ;WAS IT A CR?
1820 007720 001424 BEQ 5$ ;BR IF YES
1821 007722 022703 000012 CMP #12,R3 ;WAS IT A LF?
1822 007726 001416 BEQ 4$ ;BR IF YES
1823 007730 022703 000025 CMP #25,R3 ;WAS IT CTRL U?
1824 007734 001754 BEQ CKSWR1 ;BR IF YES(START OVER)
1825 007736 022703 000007 CMP #7,R3 ;IF CNTL G GET NEXT CHAR
1826 007742 001762 BEQ CKSWR4
1827 007744 001004 CLR R4 ;IT MUST BE A DIGIT SO CLR FLAG
1828 007746 041703 177770 BIC #177770,R3 ;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
1829 007752 006302 ASL R2 ;SHIFT R2 3 TIMES
1830 007754 006302 ASL R2
1831 007756 006302 ASL R2
1832 007760 050302 BIS R3,R2 ;ADD LAST DIGIT

```

CZDMC MACY11 30A(1052) 24-AUG-78 12:19 PAGE 38

CZDMCC.P11 11-AUG-78 08:21

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

SEQ 0038

```

1833 007762 000752          BR      CKSWR4      ;GET NEXT CHARACTER
1834 007764 012766 002002 000006 4$:  MOV      #.START,6(SP) ;LF WAS TYPED SO GO TO START
1835 007772 005704          5$:  TST      R4      ;IS FLAG CLEAR?
1836 007774 001002          BNE      6$      ;IF NOT DON'T CHANGE SOFT SWR
1837 007776 010277 171200      MOV      R2,@SWR      ;IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
1838 010002 005037 010016      6$:  CLR      SWFLG      ;CLEAR TYPEOUT FLAG
1839 010006 012604          MOV      (SP)+,R4      ;RESTORE R4
1840 010010 012603          MOV      (SP)+,R3      ;RESTORE R3
1841 010012 012602          MOV      (SP)+,R2      ;RESTORE R2
1842 010014 000207      CKSWR5: RTS      PC      ;RETURN
1843
1844 010016 000000      SWFLG: 0
1845
1846 010020 105777 171160      INCHAR: TSTB      @TKCSR
1847 010024 100375          BPL      .-4
1848 010026 017703 171154      MOV      @TKDBR,R3
1849 010032 105777 171152      TSTB      @TPCSR
1850 010036 100375          BPL      .-4
1851 010040 010377 171146      MOV      R3,@TPDBR
1852 010044 042703 000200      BIC      #BIT7,R3
1853 010050 000207          RTS      PC
1854
1855 010052 000001          SOFTSW: 1
1856 010054 006          .BYTE      6,2
1857 010056 000176          SWREG

```

```

1858
1859
1860 ;ROUTINE USED TO "CYCLE" THROUGH UP TO 16 DMC11'S
1861 ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
1862 ;AND RUNS THE SPECIFIED DMC11'S. THIS ROUTINE *MUST*
1863 ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
1864 ;SETUP NECESSARY.
1865
1866
1867 010060 005737 001306 CYCLE: TST DMACTV ;ARE ANY DMC11'S TO BE TESTED?
1868 010064 001004 BNE 1$ ;BR IF OK.
1869 010066 104402 007154 TYPE ,NOACT ;NO DMC11'S SELECTED!!
1870 010072 000000 HALT ;STOP THE SHOW.
1871 010074 000776 BR -2 ;DISQUALIFY CONT. SW.
1872 010076 000241 1$: CLC ;CLEAR PROC. CARRY BIT.
1873 010100 006137 001316 ROL RUN ;UPDATE POINTER
1874 010104 005537 001316 ADC RUN ;CATCH CARRY FROM RUN
1875 010110 062737 000004 001322 ADD #4,MILK ;UPDATE POINTER
1876 010116 062737 000010 001320 ADD #10,CREAM ;UPDATE ADDRESS POINTER.
1877 010124 022737 001700 001320 CMP #DM.MAP+200,CREAM
1878 010132 001006 BNE 2$ ;KEEP GOING; NOT ALL TESTED FOR.
1879 010134 012737 001500 001320 MOV #DM.MAP,CREAM ;RESET ADDRESS POINTER.
1880 010142 012737 001702 001322 MOV #CNT.MAP,MILK ;RESET PASS COUNT POINTER
1881 010150 033737 001316 001306 2$: BIT RUN,DMACTV ;IS THIS ONE ACTIVE?
1882 010156 001747 BEQ 1$ ;BR IF NO
1883 010160 013700 001320 MOV CREAM,R0 ;GET ADDRESS POINTER
1884 010164 013702 001322 MOV MILK,R2 ;GET PASS COUNT POINTER
1885 010170 012037 001404 MOV (R0)+,DMCSR ;LOAD SYSTEM CTRL. REG
1886 010174 011037 001374 MOV (R0),DMRVEC ;LOAD VECTOR
1887 010200 042737 177000 001374 BIC #177000,DMRVEC ;CLEAR UNWANTED BITS
1888 010206 012037 001366 MOV (R0)+,STAT1 ;LOAD STAT1
1889 010212 012037 001370 MOV (R0)+,STAT2 ;LOAD STAT2
1890 010216 012037 001372 MOV (R0)+,STAT3 ;LOAD STAT3
1891 010222 012237 001230 MOV (R2)+,PASCNT ;LOAD PASS COUNT
1892 010226 012237 001232 MOV (R2)+,ERRCNT ;LOAD ERROR COUNT
1893 010232 012700 000002 MOV #2,R0 ;SAVE CORE THIS WAY.
1894 010236 013737 001404 001406 MOV DMCSR,DMCSRH
1895 010244 005237 001406 INC DMCSRH
1896 010250 013737 001406 001410 MOV DMCSRH,DMCTL
1897 010256 005237 001410 INC DMCTL
1898 010262 013737 001410 001412 MOV DMCTL,DMP04
1899 010270 060037 001412 ADD R0,DMP04
1900 010274 013737 001412 001414 MOV DMP04,DMP06
1901 010302 060037 001414 ADD R0,DMP06
1902
1903 010306 013737 001374 001376 MOV DMRVEC,DMRLVL ;PTY LVL
1904 010314 060037 001376 ADD R0,DMRLVL ;
1905 010320 013737 001376 001400 MOV DMRLVL,DMTVEC ;TX VEC
1906 010326 060037 001400 ADD R0,DMTVEC ;
1907 010332 013737 001400 001402 MOV DMTVEC,DMTLVL ;TX LVL
1908 010340 060037 001402 ADD R0,DMTLVL ;
1909
1910 010344 032737 000002 001236 BIT #SW01,STRTSW ;IS TEST NO. SELECTED
1911 010352 001450 BEQ 7$ ;BR IF NO
1912 010354
1913 010354 005737 000042 4$: TST #42 ;RUNNING IN AUTO MODE?

```

CZDMC MACY11 30A(1052) 24-AUG-78 12:19 PAGE 40
 CZDMCC.P11 11-AUG-78 08:21

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

SEQ 0040

```

1914 010360 001045      BNE      7$      ;BR IF YES
1915 010362 104402 005672 TYPE      ,MCRLF
1916 010366 104403      INSTR      ;GET TEST NO.
1917 010370 306130      MTSTN
1918 010372 104405      PARAM
1919 010374 000001      1
1920 010376 001000      1000
1921 010400 001226      TSTNO
1922 010402      000      .BYTE 0
1923 010403      001      .BYTE 1
1924 010404 012700 012320 MOV      #TST1,R0
1925 010410 022710 5$:      CMP      (PC)+,(R0)      ;CMP FIRST WORD TO 12737
1926 010412 012737      MOV      (PC)+,@(PC)+
1927 010414 001020      BNE      6$      ;BR IF NOT SAME
1928 010416 023760 001226 000002 CMP      TSTNO,2(R0)      ;DOES TSTNO MATCH?
1929 010424 001014      BNE      6$      ;BR IF NO
1930 010426 022760 001226 000004 CMP      #TSTNO,4(R0)      ;IS LAST WORD OK?
1931 010434 001010      BNE      6$      ;BR IF NO
1932 010436 010037 001214 MOV      R0,RETURN      ;IT IS A LEGAL TEST SO DO IT
1933 010442 104402 005755 TYPE      ,MR
1934 010446 042737 000002 001236 BIC      #SW01,STR*SW
1935 010454 000412      BR      8$
1936 010456 005720 6$:      TST      (R0)+      ;POP R0
1937 010460 020027 033746 CMP      R0,#TLAST+10      ;AT END YET?
1938 010464 001351      BNE      5$      ;BR IF NO
1939 010466 104402 005666 TYPE      ,MQM      ;YES ILLEGAL TEST NO.
1940 010472 000730      BR      4$      ;TRY AGAIN
1941
1942 010474 012737 012320 001214 7$:      MOV      #TST1,RETURN      ;PREPARE RETURN ADDRESS
1943 010502 013701 001404 8$:      MOV      DMCSR,R1      ;R1 = BASE DMC11 ADDRESS
1944 010506 000177 170502 JMP      @RETURN      ;GO START TESTING.
1945
1946
1947      ;ROUTINE USED TO 'AUTO SIZE' THE DMC11
1948      ;CSR AND VECTOR.
1949      ;NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
1950      ;      ADDRESS RANGE (160000:164000)
1951      ;      AND THE VECTOR MAY BE ANY WHERE IN THE
1952      ;      FLOATING VECTOR RANGE (300:770)
1953      ;
1954      ;
1955      AUTO.SIZE:
1956 010512 000005      RESET
1957 010514 012702 001500 CSRMAP: MOV      #DM.MAP,R2      ;INSURE A BUS INIT.
1958 010520 005022 1$:      CLR      (R2)+      ;LOAD MAP POINTER.
1959 010522 022702 001700      CMP      #DM.END,R2      ;ZERO ENTIRE MAP
1960 010526 001374      BNE      1$      ;ALL DONE?
1961 010530 005037 001310      CLR      DMNUM      ;BR IF NO
1962 010534 012702 001500      MOV      #DI*.MAP,R2      ;SET OCTAL NUMBER OF DMC11'S TO 0
1963 010540 005037 001306      CLR      DMACTV      ;R2 POINTS TO DMC MAP
1964 010544 032737 000001 001236 BIT      #SW00,STR*SW      ;CLEAR ACTIVE
1965 010552 001002      BNE      +6      ;QUESTIONS?
1966 010554 000137 011252 JMP      7$      ;BR IF YES
1967 010560 012737 000001 001256 MOV      #1,TEMP5      ;IF NO SKIP QUESTIONS
1968 010566 104403      INSTR      ;START WITH 1
1969 010570 006450      NUM

```

1970	010572	104405			PARAM		
1971	010574	000001			1		
1972	010576	000020			16.		
1973	010600	001252			TEMP3		
1974	010602	000			.BYTE	0	
1975	010603	001			.BYTE	1	
1976	010604	013737	001252	001310	MOV	TEMP3,DMNUM	;DMNUM - HOW MANY
1977	010612	104402	005672	12\$:	TYPE	,MCRLF	
1978	010616	104410			CONVRT		;TYPE WHICH DMC IS BEING DONE
1979	010620	012002			WHICH		;TEMP5 IS WHICH DMC
1980	010622	005237	001256		INC	TEMP5	
1981	010626	104403			INSTR		
1982	010630	006510			CSR		
1983	010632	104405			PARAM		
1984	010634	160000			160000		
1985	010636	164000			164000		
1986	010640	001254			TEMP4		
1987	010642	000			.BYTE	0	
1988	010643	001			.BYTE	1	
1989	010644	013722	001254		MOV	TEMP4,(R2)+	;STORE CSR IN MAP
1990	010650	104403			INSTR		
1991	010652	006526			VEC		
1992	010654	104405			PARAM		
1993	010656	000000			0		
1994	010660	000776			776		
1995	010662	001254			TEMP4		
1996	010664	000			.BYTE	0	
1997	010665	001			.BYTE	1	
1998	010666	013712	001254		MOV	TEMP4,(R2)	;STORE VECTOR IN MAP
1999	010672	104402		10\$:	TYPE		
2000	010674	006547			PRI0		;ASK WHAT BR LEVEL
2001	010676	004737	012266		JSR	PC,INTTY	;GET RESPONSE
2002	010702	022703	000024		CMP	#24,R3	
2003	010706	101014			BHI	50\$	;BR IF LESS THAN 4
2004	010710	022703	000027		CMP	#27,R3	
2005	010714	103411			BLO	50\$	;BR IF GREATER THAN 7
2006	010716	012704	000011		MOV	#11,R4	;R4 = NUMBER OF SHIFTS
2007	010722	006303			ASL	R3	;SHIFT R3 LEFT
2008	010724	005304			DEC	R4	;DEC SHIFT COUNT
2009	010726	001375			BNE	-4	;BR IF NOT DONE
2010	010730	042703	170777		BIC	#170777,R3	;BIC UNWANTED BITS
2011	010734	050312			BIS	R3,(R2)	;PUT BR LEVEL IN STATUS MAP
2012	010736	000403			BR	8\$	;CONTINUE
2013	010740	104402		50\$:	TYPE		
2014	010742	005666			MQM		;RESPONSE IS OUT OF LIMITS
2015	010744	000752			BR	10\$	;TRY AGAIN
2016	010746	104402		8\$:	TYPE		
2017	010750	006606			CRAM		;DOES DMC HAVE CRAM?
2018	010752	004737	012266		JSR	PC,INTTY	;GET REPLY
2019	010756	022703	000131		CMP	#131,R3	
2020	010762	001427			BEQ	9\$	;YES
2021	010764	022703	000116		CMP	#116,R3	;NO
2022	010770	001403			BEQ	40\$	;NOT A Y OR N
2023	010772	104402			TYPE		
2024	010774	005666			MQM		;TYPE '?'
2025	010776	000763			BR	8\$	;ASK AGAIN

CZDMC MACY11 30A(1052) 24-AUG-78 12:19 PAGE 42

CZDMCC.P11 11-AUG-78 08:21

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

SEQ 0042

```

2026 011000 104402          40$: TYPE
2027 011002 007340          SPEED
2028 011004 004737 012266   JSR PC,INTTY ;DMC11-AR OR DMC11-AL?
2029 011010 022703 000122   CMP #122,R3 ;GET RESPONSE
2030 011014 001414          BEQ 16$ ;IS IT R
2031 011016 022703 000114   CMP #114,R3 ;BR IF REMOTE
2032 011022 001403          BEQ 41$ ;IS IT L
2033 011024 104402          TYPE ;BR IF LOCAL
2034 011026 005666          MQM
2035 011030 000763          BR 40$ ;TRY AGAIN
2036 011032 052762 000002 000004 41$: BIS #BIT1,4(R2) ;SET BIT1 IN STAT3
2037 011040 000402          BR 16$ ;CONTINUE
2038 011042 052712 100000   9$: BIS #BIT15,(R2) ;SET BIT 15 IF CRAM
2039 011046 104402          16$: TYPE
2040 011050 006704          MODU ;ASK WHICH LINE UNIT
2041 011052 004737 012266   JSR PC,INTTY ;GET REPLY
2042 011056 022703 000021   CMP #21,R3 ;'1'
2043 011062 001417          BEQ 30$ ;'2'
2044 011064 022703 000022   CMP #22,R3 ;'2'
2045 011070 001412          BEQ 31$ ;'N'
2046 011072 022703 000116   CMP #116,R3 ;'N'
2047 011076 001403          BEQ 32$
2048 011100 104402          TYPE
2049 011102 005666          MQM ;IF NOT A 1,2 OR N TYPE '?'
2050 011104 000760          BR 16$ ;TRY AGAIN
2051 011106 052722 010000   32$: BIS #BIT12,(R2)+ ;SET BIT 12 IN STAT2 IF NO LU
2052 011112 022222          CMP (R2)+,(R2)+ ;POP OVER STAT2 AND STAT3
2053 011114 000447          BR 33$
2054 011116 052712 020000   31$: BIS #BIT13,(R2) ;SET BIT 13 IN STAT2 IF M8202
2055 011122 104402          30$: TYPE
2056 011124 007114          CONN ;ASK IF LOOP-BACK IS ON
2057 011126 004737 012266   JSR PC,INTTY ;GET REPLY
2058 011132 022703 000131   CMP #131,R3 ;Y
2059 011136 001406          BEQ 17$ ;N
2060 011140 022703 000116   CMP #116,R3
2061 011144 001406          BEQ 18$
2062 011146 104402          TYPE
2063 011150 005666          MQM ;IF NOT Y OR N TYPE '?'
2064 011152 000763          BR 30$ ;TRY AGAIN
2065 011154 052722 040000   17$: BIS #BIT14,(R2)+ ;TURNAROUND IS CONNECTED
2066 011160 000402          BR 19$
2067 011162 042722 040000   18$: BIC #BIT14,(R2)+ ;NO TURNAROUND
2068 011166          19$:
2069 011166 104403          INSTR
2070 011170 007016          LINE
2071 011172 104405          PARAM
2072 011174 000000          0
2073 011176 000377          377
2074 011200 001254          TEMP4
2075 011202 000          .BYTE 0
2076 011203 001          .BYTE 1
2077 011204 113722 001254   MOVB TEMP4,(R2)+ ;STORE SWITCH PAC IN MAP
2078 011210 104403          INSTR
2079 011212 007054          BM
2080 011214 104405          PARAM
2081 011216 000000          C

```



```

2082 011220 000377      377
2083 011222 001254      TEMP4
2084 011224 000        .BYTE 0
2085 011225 001        .BYTE 1
2086 011226 113722 001254  MOVB TEMP4,(R2)+ ;STORE SWITCH PAC IN MAP
2087 011232 005722      TST (R2)+ ;POP OVER STAT3
2088 011234 005337 001252 33$: DEC TEMP3 ;DEC DMC COUNT
2089 011240 001402      BEQ 34$ ;BR IF DONE
2090 011242 000137 010612      JMP 12$ ;JUMP IF NOT
2091 011246 000137 011702 34$: JMP 13$ ;CONTINUE
2092 011252 012701 160000 7$: MOV #160000,R1 ;SET FOR FIRST ADDRESS TO BE TESTED
2093 011256 012737 011774 000004 MOV #6$,R1 ;SET FOR NON-EXISTANT DEVICE TIME OUT
2094 011264 005011 2$: CLR (R1) ;CLEAR SEL0
2095 011266 005711      TST (R1) ;IF DMC11 DMC SR S/B 0
2096 011270 001172      BNE 3$ ;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO DMC11
2097 011272 005061 000006      CLR 6(R1) ;CLEAR SEL6
2098 011276 005761 000006      TST 6(R1) ;IF DMC11 THEN DMRIC S/B 0.
2099 011302 001165      BNE 3$ ;BR IF NOT DMC11
2100 011304 012711 002000      MOV #BIT10,(R1) ;SET ROM0
2101 011310 005061 000004      CLR 4(R1) ;CLEAR SEL4
2102 011314 012761 125252 000006      MOV #125252,6(R1) ;WRITE THIS TO SEL6
2103 011322 052711 020000      BIS #BIT13,(R1) ;WRITE IT!
2104 011326 022761 125252 000004      CMP #125252,4(R1) ;WAS IT WRITTEN?
2105 011334 001004      BNE 21$ ;IF NO IT IS NOT CRAM
2106 011336 052762 100000 000002      BIS #BIT15,2(R2) ;SET BIT15 IF CRAM
2107 011344 000431      BR 22$
2108 011346 012711 001000 21$: MOV #BIT9,(R1) ;SET ROM1
2109 011352 012761 100430 000006      MOV #100430,6(R1) ;PUT INSTRUCTION IN SEL6
2110 011360 012711 001400      MOV #BIT9:BIT8,(R1) ;CLOCK INSTRUCTION (MICRO PROC PC TO 0)
2111 011364 012711 002000      MOV #BIT10,(R1) ;SET ROM0
2112 011370 022761 016472 000006      CMP #016472,6(R1) ;IS IT LOCAL CROM?
2113 011376 001411      BEQ 23$ ;BR IF YES
2114 011400 022761 016461 000006      CMP #016461,6(R1) ;IS IT REMOTE CROM?
2115 011406 001410      BEQ 22$ ;BR IF YES
2116 011410 022761 177777 000006      CMP #-1,6(R1) ;NO CROM?
2117 011416 001404      BEQ 22$ ;BR IF YES
2118 011420 000516      BR 3$ ;NOT A DMC
2119 011422 052762 000002 000006 23$: BIS #BIT1,6(R2) ;SET BIT 1 IN STAT3
2120      ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DMC11 CSR ADDRESS.
2121 011430 010122 22$: MOV R1,(R2)+ ;STORE CSR IN CORE TABLE.
2122 011432 012711 001000 15$: MOV #BIT9,(R1) ;CLEAR LINE UNIT LOOP
2123 011436 005061 000004      CLR 4(R1) ;CLEAR PORT4
2124 011442 012761 122113 000006      MOV #122113,6(R1) ;LOAD INSTRUCTION (CLR DTR)
2125 011450 052711 000400      BIS #BIT8,(R1) ;CLOCK INSTRUCTION
2126 011454 012761 021264 000006      MOV #021264,6(R1) ;LOAD INSTRUCTION
2127 011462 052711 000400      BIS #BIT8,(R1) ;CLOCK INSTRUCTION
2128 011466 122761 000377 000004      CMPB #377,4(R1) ;IS IT ALL ONES?
2129 011474 001003      BNE .+10 ;BR IF NO
2130 011476 052712 010000      BIS #BIT12,(R2) ;IF YES, NO LINE UNIT, SET STATUS BIT
2131 011502 000436      BR 20$
2132 011504 032761 000002 000004      BIT #BIT1,4(R1) ;IS SWITCH A ONE?
2133 011512 001403      BEQ .+10 ;BR IF M8201
2134 011514 052712 060000      BIS #BIT13:BIT14,(R2) ;M8202 ASSUME CONNECTOR
2135 011520 000427      BR 20$ ;CONNECTOR ON)
2136 011522 032761 000010 000004      BIT #BIT13,4(R1) ;IS MRDY SET
2137 011530 001023      BNE 20$ ;BR IF M8201 NO CONNECTOR (ON LINE)

```



```

2138 011532 012761 000100 000004      MOV      #BIT6,4(R1)      ;LOAD PORT4
2139 011540 012761 122113 000006      MOV      #122113,6(R1)   ;LOAD INSTRUCTION
2140 011546 052711 000400              BIS      #BIT8,(R1)      ;CLOCK INSTRUCTION(SET DTR)
2141 011552 012761 021264 000006      MOV      #021264,6(R1)  ;LOAD INSTRUCTION
2142 011560 052711 000400              BIS      #BIT8,(R1)      ;CLOCK INSTRUCTION(READ MODEM REG)
2143 011564 032761 000010 000004      BIT      #BIT3,4(R1)    ;IS MRDY SET NOW?
2144 011572 001402              BEQ      20$              ;BR IF NO CONNECTOR
2145 011574 052712 040000              BIS      #BIT14,(R2)    ;SET STATUS BIT FOR CONNECTOR
2146 011600 005722              20$: TST      (R2)+          ;POP POINTER
2147 011602 012761 021324 000006      MOV      #021324,6(R1)  ;PUT INSTRUCTION IN PORT6
2148 011610 012711 001400              MOV      #BIT9!BIT8,(R1) ;PORT4_LU 15
2149 011614 156122 000004              BISB     4(R1),(R2)+    ;STORE-DDCMP LINE # IN TABLE
2150 011620 012761 021344 000006      MOV      #021344,6(R1)  ;PORT6 INSTRUCTION
2151 011626 012711 001400              MOV      #BIT8.BIT9,(R1) ;CLOCK INSTR.
2152 011632 156122 000004              BISB     4(R1),(R2)+    ;STORE BM873 ADD IN TABLE
2153 011636 005722              TST      (R2)+          ;POP OVER STAT3
2154 011640 005011              CLR      (R1)          ;CLEAR ROMI
2155 011642 005237 001310              INC      DMNUM          ;UPDATE DEVICE COUNTER
2156 011646 022737 000020 001310      CMP      #20,DMNUM      ;ARE MAX. NO. OF DEV FOUND?
2157 011654 001412              BEQ      13$              ;YES DON'T LOOK FOR ANY MORE.
2158 011656 005011              3$: CLR      (R1)          ;CLEAR BIT 10
2159 011660 005061 000006              CLR      6(R1)          ;CLEAR SEL 6
2160 011664 062701 000010              14$: ADD     #10,R1        ;UPDATE CSR POINTER ADDRESS
2161 011670 022701 164000              CMP      #164000,R1
2162 011674 001402              BEQ      13$              ;BR IF DONE
2163 011676 000137 011264              JMP      2$              ;JUMP IF NOT
2164 011702 005037 001306              13$: CLR      DMACTV      ;WERE ANY DMC11'S FOUND AT ALL?
2165 011706 005737 001310              TST      DMNUM          ;ERROR AUTO SIZER FOUND NO DMC11'S IN THIS SYS.
2166 011712 001423              BEQ      5$              ;WERE ANY DMC11'S FOUND AT ALL?
2167 011714 013701 001310              MOV      DMNUM,R1        ;ERROR AUTO SIZER FOUND NO DMC11'S IN THIS SYS.
2168 011720 010137 001314              MOV      R1,SAVNUM      ;SAVE NUMBER OF DEVICES
2169 011724 000241              4$: CLC              ;GENERATE ACTIVE REGISTER OF DEVICES.
2170 011726 006137 001306              ROL      DMACTV          ;SET THE BIT
2171 011732 005237 001306              INC      DMACTV
2172 011736 005301              DEC      R1
2173 011740 001371              BNE      4$              ;BR IF MORE TO GENERATE
2174 011742 012737 000006 000004      MOV      #6,20$         ;RESTORE TRAP VECTOR
2175 011750 013737 001306 001322      MOV      DMACTV,SAVACT   ;SAVE ACTIVE REGISTER
2176 011756 000137 012010              JMP      VECMAP          ;GO FIND THE VECTOR NOW.
2177 011762 104402 005760              5$: TYPE     ,MERR2      ;NOTIFY OPR THAT NO DMC11'S FOUND.
2178 011766 005000              CLR      R0              ;MAKE DATA LIGHTS ZERO
2179 011770 000000              HALT
2180 011772 000776              BR      -2              ;STOP THE SHOW
2181 011774 012716 011664              6$: MOV      #14$,(SP)    ;DISABLE CONT. SW.
2182 012000 000002              RTI                    ;ENTERED BY NON-EXISTANT TIME-OUT.
2183                                ;RETURN TO MAINSTREAM
2184 012002 000001              WHICH: 1
2185 012004 002 002              .BYTE 2,2
2186 012006 001256              TEMPS
2187
2188 012010 032737 000001 001236  VECMAP: BIT      #SW00,STRISW
2189 012016 001114              BNE      5$
2190 012020 012737 000340 000022      MOV      #340,20$        ;SET IOT TRAP PRIO TO 7
2191 012026 012737 012202 000020      MOV      #4$ ,20$        ;SET IOT TRAP VECTOR
2192 012034 012702 001500              MOV      #DM.MAP,R2      ;SET SOFTWARE POINTER
2193 012040 012700 000300              MOV      #300,PC         ;FLOATING VECTORS START HERE.

```

2194	012044	012701	000302		MOV	#302,R1	:PC OF IOT INSTR.
2195	012050	010120		1\$:	MOV	R1,(R0)+	:START FILLING VECTOR AREA
2196	012052	012721	000004		MOV	#4,(R1)+	:WITH +2; IOT
2197	012056	022021			CMP	(R0)+,(R1)+	:ADD 2 TO R0 +R1
2198	012060	020127	001000		CMP	R1,#1000	
2199	012064	101771			BLOS	1\$	:BR IF MORE TO FILL
2200	012066	013737	001306	001246	MOV	DMACTV,TEMP1	:STORE TEMPORALLY
2201	012074	006037	001246		ROR	TEMP1	:BRING OUT A BIT
2202	012100	103063		2\$:	BCC	5\$	:BR IF ALL DONE
2203	012102	012704	000012		MOV	#12,R4	:R4 IS INDEX REGISTER
2204	012106	016437	012252	177776	MOV	BRLVL(R4),PS	:SET PS TO 7
2205	012114	011201			MOV	(R2),R1	
2206	012116	012761	000200	000004	MOV	#200,4(R1)	
2207	012124	012711	001000		MOV	#BIT9,(R1)	:SET ROMI
2208	012130	012761	21111	000006	MOV	#12111,6(R1)	:PUT INSTRUCTION IN PORT6
2209	012136	012711	001400		MOV	#BIT9,BIT8,(R1)	:FORCE AN INTERRUPT
2210	012142	105200		3\$:	INCB	R0	:STALL
2211	012144	001376			BNE	-2	:FOR TIME TO INTERRUPT
2212	012146	162704	000002		SUB	#2,R4	:GET NEXT LOWEST PS LEVEL
2213	012152	001404			BEQ	6\$	:BR IF R4 = 0
2214	012154	016437	012252	177776	MOV	BRLVL(R4),PS	:MOVE NEXT LOWER LEVEL IN PS
2215	012162	000767			BR	7\$	:BR TO DELAY
2216	012164	052762	005300	000002	6\$:	BIS	#5300,2(R2)
2217	012172	005011		3\$:	CLR	(R1)	:NO INTERRUPT ASSUME 300 AT LEVEL 5 AND FIX DMC11 LATER
2218	012174	062702	000010		ADD	#10,R2	:CLEAR ROMI
2219	012200	000735			BR	2\$	:POP SOFTWARE POINTER
2220	012202	051662	000002		4\$:	BIS	(SP),2(R2)
2221	012206	042762	000007	000002	BIC	#7,2(R2)	:KEEP GOING
2222	012214	016405	012254		MOV	BRLVL+2(R4),R5	:GET VECTOR ADDRESS
2223	012220	006305			ASL	R5	:CLEAR JUNK
2224	012222	006305			ASL	R5	:GET BR LEVEL OF DMC11
2225	012224	006305			ASL	R5	:SHIFT LEVEL 4 PLACES
2226	012226	006305			ASL	R5	:TO THE LEFT FOR THE
2227	012230	042705	170777		BIC	#170777,R5	:STATUS TABLE
2228	012234	050562	000002		BIS	R5,2(R2)	:CLEAR UNWANTED BITS
2229	012240	022626			CMP	(SP)+,(SP)+	:PUT BR LEVEL IN STATUS TABLE
2230	012242	012716	012172		MOV	#3\$,(SP)	:POP IOT JUNK OFF STACK
2231	012246	000002			RTI		:SET FOR RETURN
2232	012250	000207		5\$:	RTS	PC	:ALL DONE WITH "AUTO SIZING"
2233							
2234	012252	000000		BRLVL:	0	:LEVEL 0	
2235	012254	000000			0	:LEVEL 0	
2236	012256	000200			200	:LEVEL 4	
2237	012260	000240			240	:LEVEL 5	
2238	012262	000300			300	:LEVEL 6	
2239	012264	000340			340	:LEVEL 7	
2240							
2241							
2242	012266	105777	166712	INTTY:	TSTB	@TKSR	:WAIT FOR DONE
2243	012272	100375			BPL	-4	
2244	012274	017703	166706		MOV	@TKDBR,R3	:PUT CHAR IN R3
2245	012300	105777	166704		TSTB	@TPCSR	:WAIT UNTIL PRINTER IS READY
2246	012304	100375			BPL	-4	
2247	012306	010377	166700		MOV	R3,@TPDBR	:ECHO CHAR
2248	012312	042703	000240		BIC	#017,BITS,R3	:MASK OFF LOWER CASE
2249	012316	000207			RTS	PC	:RETURN

```

2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260 012320 012737 000001 001226 TST1: MOV #1,TSTNO
2261 012326 012737 012426 001216 MOV #TST2,NEXT
2262 012334 012737 012366 001220 MOV #1$,LOCK
2263
2264 012342 013701 001404 MOV DMCSR,R1 ;R1 CONTAINS BASE DMC11 ADDRESS
2265 012346 012700 000004 MOV #4,R0 ;R1 CONTAINS BASE DMC11 ADDRESS
2266 012352 012737 012420 000004 MOV #2$,4 ;4 REGISTERS TO BE TESTED
2267 012360 012737 000340 000006 MOV #340,6 ;SET UP TIMEOUT TRAP
2268 012366 005711 1$: TST (R') ;LEVEL 7
2269 012370 000240 NOP ;REFERENCE DEVICE REGISTER
2270 012372 104401 SCOP1 ;SW09=1?
2271 012374 062701 000002 ADD #2,R1 ;NEXT REGISTER
2272 012400 005300 DEC R0 ;DEC REGISTER COUNT
2273 012402 001371 BNE 1$ ;BR IF NOT LAST REGISTER
2274 012404 012737 000006 000004 MOV #6,4 ;RESTORE LOC 4
2275 012412 005037 000006 CLR 6 ;RESTORE LOC 6
2276 012416 104400 SCOPE ;SCOPE THIS TEST
2277 012420 011602 2$: MOV (SP),R2 ;GET PC OF TRAP
2278 012422 104001 HLT 1 ;TIME-OUT ERROR
2279 012424 000002 RTI
2280
2281
2282
2283
2284
2285
2286
2287
2288 012426 012737 000002 001226 TST2: MOV #2,TSTNO
2289 012434 012737 012456 001216 MOV #TST3,NEXT
2290
2291 012442 005011 CLR (R1) ;R1 CONTAINS BASE DMC11 ADDRESS
2292 012444 005005 CLR R5 ;CLEAR DMCSR
2293 012446 011104 MOV (R1),R4 ;CLEAR 'EXPECTED'
2294 012450 001401 BEQ 1$ ;PUT DMCSR IN 'FOUND'
2295 012452 104002 HLT 2 ;BR IF CLEARED
2296 012454 104400 1$: SCOPE ;ERROR DMCSR NOT CLEARED
2297 ;SCOPE THIS TEST
2298
2299
2300
2301
2302
2303
2304
2305

```

```

2306
2307 012456 012737 000003 001226 TST3: MOV #3,TSTNO
2308 012464 012737 012606 001216 MOV #TST4,NEXT
2309 012472 012737 012506 001220 MOV #1$,LOCK
2310
2311 012500 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
2312 012502 012700 000001 ;MASTER CLEAR DMC11
2313 012506 005011 1$: MOV #1,R0 ;START PATTERN AT 1
2314 012510 010005 CLR (R1) ;CLEAR REGISTER
2315 012512 010011 MOV R0,R5 ;PUT DATA IN 'EXPECTED'
2316 012514 011104 MOV R0,(R1) ;WRITE DMC REGISTER WITH PATTERN
2317 012516 020504 MOV (R1),R4 ;READ DMC REGISTER INTO 'FOUND'
2318 012520 001401 CMP R5,R4 ;IS DATA CORRECT
2319 012522 104002 BEQ 2$ ;BR IF YES
2320 012524 104401 HLT 2 ;DATA ERROR
2321 012526 005721 2$: SCOP1 ;SW09=1?
2322 012530 005200 TST (R1)+ ;NEXT REGISTER
2323 012532 022700 000005 INC R0 ;INCREMENT DATA PATTERN
2324 012536 001363 CMP #5,R0 ;LAST REGISTER?
2325 012540 013701 001404 BNE 1$ ;BR IF NO
2326 012544 012700 000001 MOV DMCSR,R1 ;BASE DMC11 ADDRESS TO R1
2327 012550 012737 012556 001220 MOV #1,R0 ;RESTART PATTERN AT 1
2328 012556 010005 3$: MOV #3$,LOCK ;NEW SCOP1
2329 012560 011104 MOV R0,R5 ;PUT DATA IN 'EXPECTED'
2330 012562 020504 MOV (R1),R4 ;READ DMC REGISTER INTO 'FOUND'
2331 012564 001401 CMP R5,R4 ;IS DATA CORRECT
2332 012566 104002 BEQ 4$ ;BR IF YES
2333 012570 104401 HLT 2 ;DUAL ADDRESSING ERROR
2334 012572 005721 4$: SCOP1 ;SW09-1?
2335 012574 005200 TST (R1)+ ;NEXT REGISTER
2336 012576 022700 000005 INC R0 ;INCREMENT PATTERN
2337 012602 001365 CMP #5,R0 ;LAST REGISTER?
2338 012604 104400 BNE 3$ ;BR IF NO
2339 SCOPE ;SCOPE THIS TEST
2340
2341 ;***** TEST 4 *****
2342 ;*CONTROL STATUS REGISTER WRITE/READ TEST
2343 ;*SET BIT0, VERIFY BIT0 WAS SET
2344 ;*CLEAR BIT0, VERIFY BIT0 WAS CLEARED
2345 ;*****
2346
2347 : TEST 4
2348 :-----
2349 012606 012737 000004 001226 TST4: MOV #4,TSTNO
2350 012614 012737 012704 001216 MOV #TST5,NEXT
2351 012622 012737 012632 001220 MOV #1$,LOCK
2352 012630 104412 MSTCLR ;MASTER CLEAR DMC11
2353 012632 013701 001404 1$: MOV DMCSR,R1 ;PUT REGISTER ADDRESS IN R1
2354 012636 012705 000001 MOV #BIT0,R5 ;PUT DATA IN 'EXPECTED'
2355 012642 010511 MOV R5,(R1) ;WRITE BIT 0
2356 012644 011104 MOV (R1),R4 ;READ CONTROL STATUS REGISTER
2357 012646 020504 CMP R5,R4 ;IS DATA CORRECT
2358 012650 001401 BEQ 2$ ;BR IF YES
2359 012652 104002 HLT 2 ;DATA ERROR
2360 012654 104401 2$: SCOP1 ;SW09 UP?
2361 012656 012737 012654 001220 MOV #3$,LOCK ;NEW SCOP1

```

```

2362 012664 042711 000001 3$: BIC #BIT0,(R1) ;CLEAR BIT 0
2363 012670 005005 CLR R5 ;CLEAR 'EXPECTED'
2364 012672 011104 MOV (R1),R4 ;READ CONTROL STATUS REGISTER
2365 012674 001402 BEQ 4$ ;BR IF ZERO
2366 012676 104002 HLT 2 ;DATA ERROR BIT0 NOT CLEARED
2367 012700 104401 SCOP1 ;SW09 UP?
2368 012702 104400 4$: SCOPE ;SCOPE THIS TEST
2369
2370
2371 ;***** TEST 5 *****
2372 ;*CONTROL STATUS REGISTER WRITE/READ TEST
2373 ;*SET BIT1, VERIFY BIT1 WAS SET
2374 ;*CLEAR BIT1, VERIFY BIT1 WAS CLEARED
2375 ;*****
2376
2377 ; TEST 5
2378
2379 012704 012737 000005 001226 TST5: MOV #5,TSTNO
2380 012712 012737 013002 001216 MOV #TST6,NEXT
2381 012720 012737 012730 001220 MOV #1$,LOCK
2382 012726 104412 MSTCLR ;MASTER CLEAR DMC11
2383 012730 013701 001404 1$: MOV DMCSR,R1 ;PUT REGISTER ADDRESS IN R1
2384 012734 012705 000002 MOV #BIT1,R5 ;PUT DATA IN 'EXPECTED'
2385 012740 010511 MOV R5,(R1) ;WRITE BIT 1
2386 012742 011104 MOV (R1),R4 ;READ CONTROL STATUS REGISTER
2387 012744 020504 CMP R5,R4 ;IS DATA CORRECT
2388 012746 001401 BEQ 2$ ;BR IF YES
2389 012750 104002 HLT 2 ;DATA ERROR
2390 012752 104401 2$: SCOP1 ;SW09 UP?
2391 012754 012737 012762 001220 MOV #3$,LOCK ;NEW SCOP1
2392 012762 042711 000002 3$: BIC #BIT1,(R1) ;CLEAR BIT 1
2393 012766 005005 CLR R5 ;CLEAR 'EXPECTED'
2394 012770 011104 MOV (R1),R4 ;READ CONTROL STATUS REGISTER
2395 012772 001402 BEQ 4$ ;BR IF ZERO
2396 012774 104002 HLT 2 ;DATA ERROR BIT1 NOT CLEARED
2397 012776 104401 SCOP1 ;SW09 UP?
2398 013000 104400 4$: SCOPE ;SCOPE THIS TEST
2399
2400
2401 ;***** TEST 6 *****
2402 ;*CONTROL STATUS REGISTER WRITE/READ TEST
2403 ;*SET BIT2, VERIFY BIT2 WAS SET
2404 ;*CLEAR BIT2, VERIFY BIT2 WAS CLEARED
2405 ;*****
2406
2407 ; TEST 6
2408
2409 013002 012737 000006 001226 TST6: MOV #6,TSTNO
2410 013010 012737 013100 001216 MOV #TST7,NEXT
2411 013016 012737 013026 001220 MOV #1$,LOCK
2412 013024 104412 MSTCLR ;MASTER CLEAR DMC11
2413 013026 013701 001404 1$: MOV DMCSR,R1 ;PUT REGISTER ADDRESS IN R1
2414 013032 012705 000004 MOV #BIT2,R5 ;PUT DATA IN 'EXPECTED'
2415 013036 010511 MOV R5,(R1) ;WRITE BIT 2
2416 013040 011104 MOV (R1),R4 ;READ CONTROL STATUS REGISTER
2417 013042 020504 CMP R5,R4 ;IS DATA CORRECT

```

2418	013044	001401				BEQ	2\$		:BR IF YES
2419	013046	104002				HLT	2		:DATA ERROR
2420	013050	104401			2\$:	SCOP1			:SW09 UP?
2421	013052	012737	013060	001220		MOV	#3\$,LOCK		:NEW SCOP1
2422	013060	042711	000004		3\$:	BIC	#BIT2,(R1)		:CLEAR BIT 2
2423	013064	005005				CLR	R5		:CLEAR 'EXPECTED'
2424	013066	011104				MOV	(R1),R4		:READ CONTROL STATUS REGISTER
2425	013070	001402				BEQ	4\$		:BR IF ZERO
2426	013072	104002				HLT	2		:DATA ERROR BIT2 NOT CLEARED
2427	013074	104401				SCOP1			:SW09 UP?
2428	013076	104400			4\$:	SCOPE			:SCOPE THIS TEST

2429
 2430
 2431
 2432
 2433
 2434
 2435
 2436
 2437
 2438
 2439
 2440
 2441
 2442
 2443
 2444
 2445
 2446
 2447
 2448
 2449
 2450
 2451
 2452
 2453
 2454
 2455
 2456
 2457
 2458
 2459
 2460
 2461
 2462
 2463
 2464
 2465
 2466
 2467
 2468

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

: TEST 7

2439	013100	012737	000007	001226	TST7:	MOV	#7,TSTNO		
2440	013106	012737	013176	001216		MOV	#TST10,NEXT		
2441	013114	012737	013124	001220		MOV	#1\$,LOCK		
2442	013122	104412				MSTCLR			:MASTER CLEAR DMC11
2443	013124	013701	001404		1\$:	MOV	DMCSR,R1		:PUT REGISTER ADDRESS IN R1
2444	013130	012705	000040			MOV	#BIT5,R5		:PUT DATA IN 'EXPECTED'
2445	013134	010511				MOV	R5,(R1)		:WRITE BIT 5
2446	013136	011104				MOV	(R1),R4		:READ CONTROL STATUS REGISTER
2447	013140	020504				CMP	R5,R4		:IS DATA CORRECT
2448	013142	001401				BEQ	2\$		:BR IF YES
2449	013144	104002				HLT	2		:DATA ERROR
2450	013146	104401			2\$:	SCOP1			:SW09 UP?
2451	013150	012737	013156	001220		MOV	#3\$,LOCK		:NEW SCOP1
2452	013156	042711	000040		3\$:	BIC	#BIT5,(R1)		:CLEAR BIT 5
2453	013162	005005				CLR	R5		:CLEAR 'EXPECTED'
2454	013164	011104				MOV	(R1),R4		:READ CONTROL STATUS REGISTER
2455	013166	001402				BEQ	4\$		:BR IF ZERO
2456	013170	104002				HLT	2		:DATA ERROR BIT5 NOT CLEARED
2457	013172	104401				SCOP1			:SW09 UP?
2458	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

2469	013176	012737	000010	001226	TST10:	MOV	#10,TSTNO		
2470	013204	012737	013274	001216		MOV	#TST11,NEXT		
2471	013212	012737	013222	001220		MOV	#1\$,LOCK		
2472	013220	104412				MSTCLR			:MASTER CLEAR DMC11
2473	013222	013701	001404		1\$:	MOV	DMCSR,R1		:PUT REGISTER ADDRESS IN R1


```

2474 013226 012705 000100      MOV    #BIT6,R5      ;PUT DATA IN 'EXPECTED'
2475 013232 010511      MOV    R5,(R1)      ;WRITE BIT 6
2476 013234 011104      MOV    (R1),R4      ;READ CONTROL STATUS REGISTER
2477 013236 020504      CMP     R5,R4      ;IS DATA CORRECT
2478 013240 001401      BEQ     2$      ;BR IF YES
2479 013242 104002      HLT     2      ;DATA ERROR
2480 013244 104401      SCOPI      ;SW09 UP?
2481 013246 012737 013254 001220 2$: MOV    #3$,LOCK      ;NEW SCOP1
2482 013254 042711 000100 3$: BIC     #BIT6,(R1)    ;CLEAR BIT 6
2483 013260 005005      CLR     R5      ;CLEAR 'EXPECTED'
2484 013262 011104      MOV    (R1),R4      ;READ CONTROL STATUS REGISTER
2485 013264 001402      BEQ     4$      ;BR IF ZERO
2486 013266 104002      HLT     2      ;DATA ERROR BIT6 NOT CLEARED
2487 013270 104401      SCOPI      ;SW09 UP?
2488 013272 104400      4$: SCOPE      ;SCOPE THIS TEST
2489
2490
2491      ;***** TEST 11 *****
2492      ;*CONTROL STATUS REGISTER WRITE/READ TEST
2493      ;*SET BIT7, VERIFY BIT7 WAS SET
2494      ;*CLEAR BIT7, VERIFY BIT7 WAS CLEARED
2495      ;*****
2496
2497      ; TEST 11
2498      ;-----
2499 013274 012737 000011 001226 TST11: MOV    #11,TSTNO
2500 013302 012737 013372 001216 MOV    #TST12,NEXT
2501 013310 012737 013320 001220 MOV    #1$,LOCK
2502 013316 104412      MSTCLR      ;MASTER CLEAR DMC11
2503 013320 013701 001404 1$: MOV    DMCSR,R1      ;PUT REGISTER ADDRESS IN R1
2504 013324 012705 000200      MOV    #BIT7,R5      ;PUT DATA IN 'EXPECTED'
2505 013330 010511      MOV    R5,(R1)      ;WRITE BIT 7
2506 013332 011104      MOV    (R1),R4      ;READ CONTROL STATUS REGISTER
2507 013334 020504      CMP     R5,R4      ;IS DATA CORRECT
2508 013336 001401      BEQ     2$      ;BR IF YES
2509 013340 104002      HLT     2      ;DATA ERROR
2510 013342 104401      SCOPI      ;SW09 UP?
2511 013344 012737 013352 001226 2$: MOV    #3$,LOCK      ;NEW SCOP1
2512 013352 042711 000200 3$: BIC     #BIT7,(R1)    ;CLEAR BIT 7
2513 013356 005005      CLR     R5      ;CLEAR 'EXPECTED'
2514 013360 011104      MOV    (R1),R4      ;READ CONTROL STATUS REGISTER
2515 013362 001402      BEQ     4$      ;BR IF ZERO
2516 013364 104002      HLT     2      ;DATA ERROR BIT7 NOT CLEARED
2517 013366 104401      SCOPI      ;SW09 UP?
2518 013370 104400      4$: SCOPE      ;SCOPE THIS TEST
2519
2520
2521      ;***** TEST 12 *****
2522      ;*CONTROL STATUS REGISTER WRITE/READ TEST
2523      ;*SET BIT9, VERIFY BIT9 WAS SET
2524      ;*CLEAR BIT9, VERIFY BIT9 WAS CLEARED
2525      ;*****
2526
2527      ; TEST 12
2528      ;-----
2529 013372 012737 000012 001226 TST12: MOV    #12,TSTNO
  
```

```

2530 013400 012737 013470 001216      MOV      #TST13,NEXT
2531 013406 012737 013416 001220      MOV      #1$,LOCK
2532 013414 104412                      MSTCLR
2533 013416 013701 001404          1$:  MOV      DMCSR,R1      ;MASTER CLEAR DMC11
2534 013422 012705 001000          MOV      #BIT9,R5      ;PUT REGISTER ADDRESS IN R1
2535 013426 010511                      MOV      R5,(R1)      ;PUT DATA IN 'EXPECTED'
2536 013430 011104                      MOV      (R1),R4      ;WRITE BIT 9
2537 013432 020504                      CMP      R5,R4      ;READ CONTROL STATUS REGISTER
2538 013434 001401                      BEQ      2$      ;IS DATA CORRECT
2539 013436 104002                      HLT      2      ;BR IF YES
2540 013440 104401          2$:  SCOP1      ;DATA ERROR
2541 013442 012737 013450 001220      MOV      #3$,LOCK      ;SW09 UP?
2542 013450 042711 001000          3$:  BIC      #BIT9,(R1)      ;NEW SCOP1
2543 013454 005005                      CLR      R5      ;CLEAR BIT 9
2544 013456 011104                      MOV      (R1),R4      ;CLEAR 'EXPECTED'
2545 013460 001402                      BEQ      4$      ;READ CONTROL STATUS REGISTER
2546 013462 104002                      HLT      2      ;BR IF ZERO
2547 013464 104401          SCOP1      ;DATA ERROR BIT9 NOT CLEARED
2548 013466 104400          4$:  SCOPE      ;SW09 UP?
2549                                ;SCOPE THIS TEST
2550
2551                                ;***** TEST 13 *****
2552                                ;*CONTROL STATUS REGISTER WRITE/READ TEST
2553                                ;*SET BIT11, VERIFY BIT11 WAS SET
2554                                ;*CLEAR BIT11, VERIFY BIT11 WAS CLEARED
2555                                ;*****
2556
2557                                ; TEST 13
2558                                ;-----
2559 013470 012737 000013 001226  TST13: MOV      #13,TSTNO
2560 013476 012737 013566 001216      MOV      #TST14,NEXT
2561 013504 012737 013514 001220      MOV      #1$,LOCK
2562 013512 104412                      MSTCLR
2563 013514 013701 001404          1$:  MOV      DMCSR,R1      ;MASTER CLEAR DMC11
2564 013520 012705 004000          MOV      #BIT11,R5      ;PUT REGISTER ADDRESS IN R1
2565 013524 010511                      MOV      R5,(R1)      ;PUT DATA IN 'EXPECTED'
2566 013526 011104                      MOV      (R1),R4      ;WRITE BIT 11
2567 013530 020504                      CMP      R5,R4      ;READ CONTROL STATUS REGISTER
2568 013532 001401                      BEQ      2$      ;IS DATA CORRECT
2569 013534 104002                      HLT      2      ;BR IF YES
2570 013536 104401          2$:  SCOP1      ;DATA ERROR
2571 013540 012737 013546 001220      MOV      #3$,LOCK      ;SW09 UP?
2572 013546 042711 004000          3$:  BIC      #BIT11,(R1)      ;NEW SCOP1
2573 013552 005005                      CLR      R5      ;CLEAR BIT 11
2574 013554 011104                      MOV      (R1),R4      ;CLEAR 'EXPECTED'
2575 013556 001402                      BEQ      4$      ;READ CONTROL STATUS REGISTER
2576 013560 104002                      HLT      2      ;BR IF ZERO
2577 013562 104401          SCOP1      ;DATA ERROR BIT11 NOT CLEARED
2578 013564 104400          4$:  SCOPE      ;SW09 UP?
2579                                ;SCOPE THIS TEST
2580
2581                                ;***** TEST 14 *****
2582                                ;*CONTROL STATUS REGISTER WRITE/READ TEST
2583                                ;*SET BIT12, VERIFY BIT12 WAS SET
2584                                ;*CLEAR BIT12, VERIFY BIT12 WAS CLEARED
2585                                ;*****

```



```

2586
2587
2588
2589 013566 012737 000014 001226 TST14: MOV #14,TSTNO
2590 013574 012737 013664 001216 MOV #TST15,NEXT
2591 013602 012737 013612 001220 MOV #1$,LOCK
2592 013610 104412 MSTCLR
2593 013612 013701 001404 1$: MOV DMCSR,R1 ;MASTER CLEAR DMC11
2594 013616 012705 010000 MOV #BIT12,R5 ;PUT REGISTER ADDRESS IN R1
2595 013622 010511 MOV R5,(R1) ;PUT DATA IN 'EXPECTED'
2596 013624 011104 MOV (R1),R4 ;WRITE BIT 12
2597 013626 020504 CMP R5,R4 ;READ CONTROL STATUS REGISTER
2598 013630 001401 BEQ 2$ ;IS DATA CORRECT
2599 013632 104002 HLT 2 ;BR IF YES
2600 013634 104401 2$: SCOP1 ;DATA ERROR
2601 013636 012737 013644 001220 MOV #3$,LOCK ;SW09 UP?
2602 013644 042711 010000 3$: BIC #BIT12,(R1) ;NEW SCOP1
2603 013650 005005 CLR R5 ;CLEAR BIT 12
2604 013652 011104 MOV (R1),R4 ;CLEAR 'EXPECTED'
2605 013654 001402 BEQ 4$ ;READ CONTROL STATUS REGISTER
2606 013656 104002 HLT 2 ;BR IF ZERO
2607 013660 104401 SCOP1 ;DATA ERROR BIT12 NOT CLEARED
2608 013662 104400 4$: SCOPE ;SW09 UP?
2609 ;SCOPE THIS TEST
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619 013664 012737 000015 001226 TST15: MOV #15,TSTNO
2620 013672 012737 013762 001216 MOV #TST16,NEXT
2621 013700 012737 013710 001220 MOV #1$,LOCK
2622 013706 104412 MSTCLR
2623 013710 013701 001410 1$: MOV DMCTL,R1 ;MASTER CLEAR DMC11
2624 013714 012705 000001 MOV #BIT0,R5 ;PUT REGISTER ADDRESS IN R1
2625 013720 010511 MOV R5,(R1) ;PUT DATA IN 'EXPECTED'
2626 013722 011104 MOV (R1),R4 ;WRITE BIT 0
2627 013724 020504 CMP R5,R4 ;READ CONTROL OUT REGISTER
2628 013726 001401 BEQ 2$ ;IS DATA CORRECT
2629 013730 104002 HLT 2 ;BR IF YES
2630 013732 104401 2$: SCOP1 ;DATA ERROR
2631 013734 012737 013742 001220 MOV #3$,LOCK ;SW09 UP?
2632 013742 042711 000001 3$: BIC #BIT0,(R1) ;NEW SCOP1
2633 013746 005005 CLR R5 ;CLEAR BIT 0
2634 013750 011104 MOV (R1),R4 ;CLEAR 'EXPECTED'
2635 013752 001402 BEQ 4$ ;READ CONTROL OUT REGISTER
2636 013754 104002 HLT 2 ;BR IF ZERO
2637 013756 104401 SCOP1 ;DATA ERROR BIT0 NOT CLEARED
2638 013760 104400 4$: SCOPE ;SW09 UP?
2639 ;SCOPE THIS TEST
2640
2641

```

***** TEST 16 *****

2642 ;*CONTROL OUT REGISTER WRITE/READ TEST
 2643 ;*SET BIT1, VERIFY BIT1 WAS SET
 2644 ;*CLEAR BIT1, VERIFY BIT1 WAS CLEARED
 2645 ;*****
 2646

2647 : TEST 16
 2648 :-----
 2649 013762 012737 000016 001226 TST16: MOV #16,TSTNO
 2650 013770 012737 014060 001216 MOV #TST17,NEXT
 2651 013776 012737 014006 001220 MOV #1\$,LOCK
 2652 014004 104412 MSTCLR ;MASTER CLEAR DMC11
 2653 014006 013701 001410 1\$: MOV DMCTL,R1 ;PUT REGISTER ADDRESS IN R1
 2654 014012 012705 000002 MOV #BIT1,R5 ;PUT DATA IN 'EXPECTED'
 2655 014016 010511 MOV R5,(R1) ;WRITE BIT 1
 2656 014020 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
 2657 014022 020504 CMP R5,R4 ;IS DATA CORRECT
 2658 014024 001401 BEQ 2\$;BR IF YES
 2659 014026 104002 HLT 2 ;DATA ERROR
 2660 014030 104401 2\$: SCOP1 ;SW09 UP?
 2661 014032 012737 014040 001220 MOV #3\$,LOCK ;NEW SCOP1
 2662 014040 042711 000002 3\$: BIC #BIT1,(R1) ;CLEAR BIT 1
 2663 014044 005005 CLR R5 ;CLEAR 'EXPECTED'
 2664 014046 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
 2665 014050 001402 BEQ 4\$;BR IF ZERO
 2666 014052 104002 HLT 2 ;DATA ERROR BIT1 NOT CLEARED
 2667 014054 104401 SCOP1 ;SW09 UP?
 2668 014056 104400 4\$: SCOPE ;SCOPE THIS TEST
 2669
 2670

2671 ;***** TEST 17 *****
 2672 ;*CONTROL OUT REGISTER WRITE/READ TEST
 2673 ;*SET BIT2, VERIFY BIT2 WAS SET
 2674 ;*CLEAR BIT2, VERIFY BIT2 WAS CLEARED
 2675 ;*****
 2676

2677 : TEST 17
 2678 :-----
 2679 014060 012737 000017 001226 TST17: MOV #17,TSTNO
 2680 014066 012737 014156 001216 MOV #TST20,NEXT
 2681 014074 012737 014104 001220 MOV #1\$,LOCK
 2682 014102 104412 MSTCLR ;MASTER CLEAR DMC11
 2683 014104 013701 001410 1\$: MOV DMCTL,R1 ;PUT REGISTER ADDRESS IN R1
 2684 014110 012705 000004 MOV #BIT2,R5 ;PUT DATA IN 'EXPECTED'
 2685 014114 010511 MOV R5,(R1) ;WRITE BIT 2
 2686 014116 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
 2687 014120 020504 CMP R5,R4 ;IS DATA CORRECT
 2688 014122 001401 BEQ 2\$;BR IF YES
 2689 014124 104002 HLT 2 ;DATA ERROR
 2690 014126 104401 2\$: SCOP1 ;SW09 UP?
 2691 014130 012737 014136 001220 MOV #3\$,LOCK ;NEW SCOP1
 2692 014136 042711 000004 3\$: BIC #BIT2,(R1) ;CLEAR BIT 2
 2693 014142 005005 CLR R5 ;CLEAR 'EXPECTED'
 2694 014144 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
 2695 014146 001402 BEQ 4\$;BR IF ZERO
 2696 014150 104002 HLT 2 ;DATA ERROR BIT2 NOT CLEARED
 2697 014152 104401 SCOP1 ;SW09 UP?

```

2698 014154 104400 4$: SCOPE ;SCOPE THIS TEST
2699
2700
2701 :***** TEST 20 *****
2702 :*CONTROL OUT REGISTER WRITE/READ TEST
2703 :*SET BIT6, VERIFY BIT6 WAS SET
2704 :*CLEAR BIT6, VERIFY BIT6 WAS CLEARED
2705 :*****
2706
2707 : TEST 20
2708 :-----
2709 014156 012737 000020 001226 TST20: MOV #20,TSTNO
2710 014164 012737 014254 001216 MOV #TST21,NEXT
2711 014172 012737 014202 001220 MOV #1$,LOCK
2712 014200 104412 MSTCLR ;MASTER CLEAR DMC11
2713 014202 013701 001410 1$: MOV DMCTL,R1 ;PUT REGISTER ADDRESS IN R1
2714 014206 012705 000100 MOV #BIT6,R5 ;PUT DATA IN 'EXPECTED'
2715 014212 010511 MOV R5,(R1) ;WRITE BIT 6
2716 014214 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
2717 014216 020504 CMP R5,R4 ;IS DATA CORRECT
2718 014220 001401 BEQ 2$ ;BR IF YES
2719 014222 104002 HLT 2 ;DATA ERROR
2720 014224 104401 2$: SCOP1 ;SW09 UP?
2721 014226 012737 014234 001220 MOV #3$,LOCK ;NEW SCOP1
2722 014234 042711 000100 3$: BIC #BIT6,(R1) ;CLEAR BIT 6
2723 014240 005005 CLR R5 ;CLEAR 'EXPECTED'
2724 014242 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
2725 014244 001402 BEQ 4$ ;BR IF ZERO
2726 014246 104002 HLT 2 ;DATA ERROR BIT6 NOT CLEARED
2727 014250 104401 SCOP1 ;SW09 UP?
2728 014252 104400 4$: SCOPE ;SCOPE THIS TEST
2729
2730
2731 :***** TEST 21 *****
2732 :*CONTROL OUT REGISTER WRITE/READ TEST
2733 :*SET BIT7, VERIFY BIT7 WAS SET
2734 :*CLEAR BIT7, VERIFY BIT7 WAS CLEARED
2735 :*****
2736
2737 : TEST 21
2738 :-----
2739 014254 012737 000021 001226 TST21: MOV #21,TSTNO
2740 014262 012737 014352 001216 MOV #TST22,NEXT
2741 014270 012737 014300 001220 MOV #1$,LOCK
2742 014276 104412 MSTCLR ;MASTER CLEAR DMC11
2743 014300 013701 001410 1$: MOV DMCTL,R1 ;PUT REGISTER ADDRESS IN R1
2744 014304 012705 000200 MOV #BIT7,R5 ;PUT DATA IN 'EXPECTED'
2745 014310 010511 MOV R5,(R1) ;WRITE BIT 7
2746 014312 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
2747 014314 020504 CMP R5,R4 ;IS DATA CORRECT
2748 014316 001401 BEQ 2$ ;BR IF YES
2749 014320 104002 HLT 2 ;DATA ERROR
2750 014322 104401 2$: SCOP1 ;SW09 UP?
2751 014324 012737 014332 001220 MOV #3$,LOCK ;NEW SCOP1
2752 014332 042711 000200 3$: BIC #BIT7,(R1) ;CLEAR BIT 7
2753 014336 005005 CLR R5 ;CLEAR 'EXPECTED'

```

```

2754 014340 011104      MOV      (R1),R4      ;READ CONTROL OUT REGISTER
2755 014342 001402      BEQ      4$      ;BR IF ZERO
2756 014344 104002      HLT      2      ;DATA ERROR BIT7 NOT CLEARED
2757 014346 104401      SCOP1     ;SW09 UP?
2758 014350 104400      4$:      SCOPE     ;SCOPE THIS TEST
2759
2760
2761      ;***** TEST 22 *****
2762      ;*CONTROL OUT REGISTER WRITE/READ TEST
2763      ;*SET BIT12, VERIFY BIT12 WAS SET
2764      ;*CLEAR BIT12, VERIFY BIT12 WAS CLEARED
2765      ;*****
2766
2767      : TEST 22
2768      :-----
2769 014352 012737 000022 001226 TST22: MOV      #22,TSTNO
2770 014360 012737 014450 001216      MOV      #TST23,NEXT
2771 014366 012737 014376 001220      MOV      #1$,LOCK
2772 014374 104412      MSTCLR     ;MASTER CLEAR DMC11
2773 014376 013701 001410      1$:      MOV      DMCTL,R1      ;PUT REGISTER ADDRESS IN R1
2774 014402 012705 010000      MOV      #BIT12,R5      ;PUT DATA IN 'EXPECTED'
2775 014406 010511      MOV      R5,(R1)      ;WRITE BIT 12
2776 014410 011104      MOV      (R1),R4      ;READ CONTROL OUT REGISTER
2777 014412 020504      CMP      R5,R4      ;IS DATA CORRECT
2778 014414 001401      BEQ      2$      ;BR IF YES
2779 014416 104002      HLT      2      ;DATA ERROR
2780 014420 104401      2$:      SCOP1     ;SW09 UP?
2781 014422 012737 014430 001220      MOV      #3$,LOCK      ;NEW SCOP1
2782 014430 042711 010000      3$:      BIC      #BIT12,(R1)      ;CLEAR BIT 12
2783 014434 005005      CLR      R5      ;CLEAR 'EXPECTED'
2784 014436 011104      MOV      (R1),R4      ;READ CONTROL OUT REGISTER
2785 014440 001402      BEQ      4$      ;BR IF ZERO
2786 014442 104002      HLT      2      ;DATA ERROR BIT12 NOT CLEARED
2787 014444 104401      SCOP1     ;SW09 UP?
2788 014446 104400      4$:      SCOPE     ;SCOPE THIS TEST
2789
2790
2791      ;***** TEST 23 *****
2792      ;*CONTROL OUT REGISTER WRITE/READ TEST
2793      ;*SET BIT13, VERIFY BIT13 WAS SET
2794      ;*CLEAR BIT13, VERIFY BIT13 WAS CLEARED
2795      ;*****
2796
2797      : TEST 23
2798      :-----
2799 014450 012737 000023 001226 TST23: MOV      #23,TSTNO
2800 014456 012737 014546 001216      MOV      #TST24,NEXT
2801 014464 012737 014474 001220      MOV      #1$,LOCK
2802 014472 104412      MSTCLR     ;MASTER CLEAR DMC11
2803 014474 013701 001410      1$:      MOV      DMCTL,R1      ;PUT REGISTER ADDRESS IN R1
2804 014500 012705 020000      MOV      #BIT13,R5      ;PUT DATA IN 'EXPECTED'
2805 014504 010511      MOV      R5,(R1)      ;WRITE BIT 13
2806 014506 011104      MOV      (R1),R4      ;READ CONTROL OUT REGISTER
2807 014510 020504      CMP      R5,R4      ;IS DATA CORRECT
2808 014512 001401      BEQ      2$      ;BR IF YES
2809 014514 104002      HLT      2      ;DATA ERROR

```

```

2810 014516 104401 2$: SCOP1 ;SW09 UP?
2811 014520 012737 014526 001220 MOV #3$,LOCK ;NEW SCOP1
2812 014526 042711 020000 3$: BIC #BIT13,(R1) ;CLEAR BIT 13
2813 014532 005005 CLR R5 ;CLEAR 'EXPECTED'
2814 014534 011104 MOV (R1),R4 ;READ CONTROL OUT REGISTER
2815 014536 001402 BEQ 4$ ;BR IF ZERO
2816 014540 104002 HLT 2 ;DATA ERROR BIT13 NOT CLEARED
2817 014542 104401 SCOP1 ;SW09 UP?
2818 014544 104400 4$: SCOPE ;SCOPE THIS TEST
2819
2820
2821 ;***** TEST 24 *****
2822 ;*PORT4 REGISTER WRITE/READ TEST
2823 ;*FLOAT A ONE THROUGH PORT4 REGISTER
2824 ;*FLOAT A ZERO THROUGH PORT4 REGISTER
2825 ;*****
2826
2827 ; TEST 24
2828 -----
2829 014546 012737 000024 001226 TST24: MOV #24,TSTNO
2830 014554 012737 014672 001216 MOV #TST25,NEXT
2831 014562 012737 014602 001220 MOV #64$,LOCK
2832 014570 104412 MSTCLR ;MASTER CLEAR DMC11
2833 014572 013701 001412 MOV DMP04,R1 ;PUT REGISTER ADDRESS IN R1
2834 014576 012700 000001 MOV #1,R0 ;START WITH BIT0
2835 014602 64$: MOV R0,R5 ;PUT 'EXPECTED' IN R5
2836 014602 010005 MOV R5,(R1) ;WRITE PORT4 REGISTER
2837 014604 010511 MOV (R1),R4 ;READ PORT4 REGISTER
2838 014606 011104 CMP R5,R4 ;COMPARE EXPECTED AND FOUND
2839 014610 020504 BEQ 65$ ;BR IF OK
2840 014612 001401 HLT 2 ;WRITE/READ ERROR
2841 014614 104002 65$: SCOP1 ;LOOP TO 64$ IF SW09-1
2842 014616 104401 CLC ;CLEAR CARRY
2843 014620 000241 ROL R0 ;SHIFT TO NEXT BIT
2844 014622 006100 BNE 64$ ;BR IF NOT DONE YET?
2845 014624 001366 MOV #66$,LOCK ;NEW SCOP1
2846 014626 012737 014640 001220 MOV #1,R0 ;START WITH BIT0
2847 014634 012700 000001 66$: COM R0 ;CHANGE TO A FLOATING ZERO
2848 014640 005100 MOV R0,R5 ;PUT 'EXPECTED' IN R5
2849 014642 010005 MOV R5,(R1) ;WRITE PORT4 REGISTER
2850 014644 010511 MOV (R1),R4 ;READ PORT4 REGISTER
2851 014646 011104 CMP R5,R4 ;COMPARE EXPECTED AND FOUND
2852 014648 020504 BEQ 67$ ;BR IF OK
2853 014650 001401 HLT 2 ;WRITE/READ ERROR
2854 014652 001401 67$: SCOP1 ;LOOP TO 66$ IF SW09 1
2855 014654 104002 COM R0 ;CHANGE BACK TO A FLOATING ONE
2856 014656 104401 CLC ;CLEAR CARRY
2857 014660 005100 ROL R0 ;SHIFT TO NEXT BIT
2858 014662 000241 BNE 66$ ;BR IF NOT DONE YET?
2859 014664 006100 SCOPE ;SCOPE THIS TEST
2860 014666 001364
2861 014670 104400
2862
2863
2864 ;***** TEST 25 *****
2865 ;*PORT6 REGISTER WRITE/READ TEST

```

```
2866      ;*FLOAT A ONE THROUGH PORT6 REGISTER
2867      ;*FLOAT A ZERO THROUGH PORT6 REGISTER
2868      ;*****
2869
2870      : TEST 25
2871      :-----
2872 014672 012737 000025 001226 TST25: MOV #25,TSTNO
2873 014700 012737 015016 001216      MOV #TST26,NEXT
2874 014706 012737 014726 001220      MOV #64$,LOCK
2875 014714 104412          MSTCLR      ;MASTER CLEAR DMC11
2876 014716 013701 001414          MOV DMP06,R1      ;PUT REGISTER ADDRESS IN R1
2877 014722 012700 000001          MOV #1,R0      ;START WITH BIT0
2878 014726          64$: MOV R0,R5      ;PUT 'EXPECTED' IN R5
2879 014726 010005          MOV R5,(R1)      ;WRITE PORT6 REGISTER
2880 014730 010511          MOV (R1),R4      ;READ PORT6 REGISTER
2881 014732 011104          CMP R5,R4      ;COMPARE EXPECTED AND FOUND
2882 014734 020504          BEQ 65$      ;BR IF OK
2883 014736 001401          HLT 2      ;WRITE/READ ERROR
2884 014740 104002          65$: SCOP1      ;LOOP TO 64$ IF SW09=1
2885 014742 104401          CLC      ;CLEAR CARRY
2886 014744 000241          ROL R0      ;SHIFT TO NEXT BIT
2887 014746 006100          BNE 64$      ;BR IF NOT DONE YET?
2888 014750 001366          MOV #66$,LOCK      ;NEW SCOP1
2889 014752 012737 014764 001220      MOV #1,R0      ;START WITH BIT0
2890 014760 012700 000001          66$: COM R0      ;CHANGE TO A FLOATING ZERO
2891 014764          MOV R0,R5      ;PUT 'EXPECTED' IN R5
2892 014766 010005          MOV R5,(R1)      ;WRITE PORT6 REGISTER
2893 014770 010511          MOV (R1),R4      ;READ PORT6 REGISTER
2894 014772 011104          CMP R5,R4      ;COMPARE EXPECTED AND FOUND
2895 014774 020504          BEQ 67$      ;BR IF OK
2896 014776 001401          HLT 2      ;WRITE/READ ERROR
2897 015000 104002          67$: COM R0      ;CHANGE BACK TO A FLOATING ONE
2898 015002 104401          CLC      ;CLEAR CARRY
2899 015004 005100          ROL R0      ;SHIFT TO NEXT BIT
2900 015006 000241          BNE 66$      ;BR IF NOT DONE YET?
2901 015010 006100          SCOPE      ;SCOPE THIS TEST
2902 015012 001364
2903 015014 10440C
2904
2905
2906
2907      ;***** TEST 26 *****
2908      ;*UNIBUS REGISTER BYTE DUAL ADDRESSING TEST
2909      ;*LOAD ALL REGISTERS WITH INCREMENTING PATTERN
2910      ;*READ BACK ALL REGISTERS TO VERIFY CORRECT ADDRESSING
2911      ;*****
2912
2913      : TEST 26
2914      :-----
2915 015016 012737 000026 001226 TST26: MOV #26,TSTNO
2916 015024 012737 015146 001216      MOV #TST27,NEXT
2917 015032 012737 015046 001220      MOV #18,LOCK
2918
2919 015040 104412          MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
2920 015042 012700 000001          MOV #1,R0      ;MASTER CLEAR DMC11
2921 015046 105011          18: CLRB R1      ;START PATTERN AT 1
2922          ;CLEAR REGISTER
```


2922	015050	110005		MOVB	R0,R5	:PUT DATA IN 'EXPECTED'
2923	015052	110011		MOVB	R0,(R1)	:WRITE DMC REGISTER WITH PATTERN
2924	015054	111104		MOVB	(R1),R4	:READ DMC REGISTER INTO 'FOUND'
2925	015056	020504		CMP	R5,R4	:IS DATA CORRECT
2926	015060	001401		BEQ	2\$	:BR IF YES
2927	015062	104002		HLT	2	:DATA ERROR
2928	015064	104401	2\$:	SCOPI		:SW09=1?
2929	015066	105721		ISTB	(R1)+	:NEXT REGISTER
2930	015070	005200		INC	R0	:INCREMENT DATA PATTERN
2931	015072	022700	00001*	CMP	#11,R0	:LAST REGISTER?
2932	015076	001363		BNE	1\$	:BR IF NO
2933	015100	013701	001404	MOV	DMCSR,R1	:BASE DMC11 ADDRESS TO R1
2934	015104	012700	000001	MOV	#1,R0	:RESTART PATTERN AT 1
2935	015110	012737	01511*	MOV	#3\$,LOCK	:NEW SCOPI
2936	015116	110005		MOVB	R0,R5	:PUT DATA IN 'EXPECTED'
2937	015120	111104		MOVB	(R1),R4	:READ DMC REGISTER INTO 'FOUND'
2938	015122	020504		CMP	R5,R4	:IS DATA CORRECT
2939	015124	001401		BEQ	4\$	:BR IF YES
2940	015126	104002		HLT	2	:DUAL ADDRESSING ERROR
2941	015130	104401	4\$:	SCOPI		:SW09=1?
2942	015132	105721		ISTB	(R1)+	:NEXT REGISTER
2943	015134	005200		INC	R0	:INCREMENT PATTERN
2944	015136	022700	00001*	CMP	#11,R0	:LAST REGISTER?
2945	015142	001365		BNE	3\$	:BR IF NO
2946	015144	104400		SCOPE		:SCOPE THIS TEST

***** 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

2957	015146	012737	000027	001226	1\$:	MOV	#27,ISTNO	
2958	015154	012737	015306	001216		MOV	#TST30,NEXT	
2959	015162	012737	015200	001220		MOV	#1\$,LOCK	
2960								:R1 CONTAINS BASE DMC11 ADDRESS
2961	015170	104412				MSTCLR		:MASTER CLEAR DMC11
2962	015172	012711	003000			MOV	#BIT9,BIT10,(R1)	:SEL6 IS NOW THE IR
2963	015176	005005				CLR	R5	:PUT 'EXPECTED' IN R5
2964	015200	010561	000006		1\$:	MOV	R5,6(R1)	:CLEAR THE IR
2965	015204	016104	000006			MOV	6(R1),R4	:READ THE IR
2966	015210	020504				CMP	R5,R4	:IS IT CLEARED?
2967	015212	001401				BEQ	2\$	:BR IF YES
2968	015214	104023				HLT	23	:ERROR IR IS NOT CLEAR
2969	015216	104401			2\$:	SCOPI		:LOOP TO 1\$ IF SW09 1
2970	015220	012737	015232	001		MOV	#3\$,LOCK	:NEW SCOPI
2971	015226	012705	177777			MOV	#-1,R5	:PUT 'EXPECTED' IN R5
2972	015232	010561	000006		1\$:	MOV	R5,6(R1)	:WRITE ALL ONES TO THE IR
2973	015236	016104	000006			MOV	6(R1),R4	:READ THE IR
2974	015242	020504				CMP	R5,R4	:IS IT ALL ONES?
2975	015244	001401				BEQ	4\$	:BR IF YES
2976	015246	104023				HLT	2	:ERROR IR IS NOT ALL ONE
2977	015250	104411			4\$:	SCOPI		:LOOP TO 3\$ IF SW09 1

```

2978 015252 012737 015262 001220      MOV    #5$,LOCK      ;NEW SCOPE1
2979 015260 005005                      CLR    R5           ;PUT 'EXPECTED' IN R5
2980 015262 000005      5$:      RESET      ;BUS RESET
2981 015264 012711 003000      MOV    #BIT9:BIT10,(R1) ;SEL6 IS IR
2982 015270 016104 000006      MOV    6(R1),R4      ;READ THE IR
2983 015274 020504      CMP    R5,R4          ;IS IT CLEARED?
2984 015276 001401      BEQ    6$           ;BR IF YES
2985 015300 104023      HLT    23           ;ERROR, IR IS NOT CLEARED
2986 015302 104401      6$:      SCOP1      ;LOOP TO 5$ IF SW09=1
2987 015304 104401      SCOP1      ;SCOPE THIS TEST
2988
2989
2990
2991      ;***** TEST 30 *****
2992      ;*MAINTENANCE INSTRUCTION REGISTER TEST
2993      ;*VERIFY THAT THE MAINT IR CAN BE WRITTEN TO ALL ZEROS'
2994      ;*AND ALL ONES'. VERIFY THAT IT IS CLEARED ON A MASTER RESET.
2995      ;*****
2996
2997      : TEST 30
2998      :-----
2998 015306 012737 000030 001226 1$T30: MOV    #30,TSTNO
2999 015314 012737 015450 001216      MOV    #TST31,NEXT
3000 015322 012737 015340 001220      MOV    #1$,LOCK
3001
3002 015330 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
3003 015332 012711 003000      MOV    #BIT9:BIT10,(R1) ;MASTER CLEAR DMC11
3004 015336 005005      CLR    R5           ;SEL6 IS NOW THE IR
3005 015340 010561 000006      1$:      MOV    R5,6(R1)      ;PUT 'EXPECTED' IN R5
3006 015344 016104 000006      MOV    6(R1),R4      ;CLEAR THE IR
3007 015350 020504      CMP    R5,R4          ;READ THE IR
3008 015352 001401      BEQ    2$           ;IS IT CLEARED?
3009 015354 104023      HLT    23           ;BR IF YES
3010 015356 104401      2$:      SCOP1      ;ERROR IR IS NOT CLEAR
3011 015360 012737 015372 001220      MOV    #3$,LOCK      ;LOOP TO 1$ IF SW09=1
3012 015366 012705 177777      MOV    #-1,R5      ;NEW SCOPE1
3013 015372 010561 000006      3$:      MOV    R5,6(R1)      ;PUT 'EXPECTED' IN R5
3014 015376 016104 000006      MOV    6(R1),R4      ;WRITE ALL ONES TO THE IR
3015 015402 020504      CMP    R5,R4          ;READ THE IR
3016 015404 001401      BEQ    4$           ;IS IT ALL ONES?
3017 015406 104023      HLT    23           ;BR IF YES
3018 015410 104401      4$:      SCOP1      ;ERROR IR IS NOT - ALL ONES
3019 015412 012737 015422 001220      MOV    #5$,LOCK      ;LOOP TO 3$ IF SW09=1
3020 015420 005005      CLR    R5           ;NEW SCOPE1
3021 015422 052711 040000      5$:      BIS    #BIT14,(R1)      ;PUT 'EXPECTED' IN R5
3022 015426 012711 003000      MOV    #BIT9:BIT10,(R1) ;MASTER CLEAR
3023 015432 016104 000006      MOV    6(R1),R4      ;SEL6 IS IR
3024 015436 020504      CMP    R5,R4          ;READ THE IR
3025 015440 001401      BEQ    6$           ;IS IT CLEARED?
3026 015442 104023      HLT    23           ;BR IF YES
3027 015444 104401      6$:      SCOP1      ;ERROR, IR IS NOT CLEARED
3028 015446 104401      SCOP1      ;LOOP TO 5$ IF SW09=1
3029
3030      ;***** TEST 31 *****
3031      ;*MICRO PROCESSOR TEST
3032      ;*LOAD DMP06 WITH A MICRO-PROCESSOR INSTRUCTION, CLOCK IT
3033

```



```

3034      ;*VERIFY INSTRUCTION EXECUTED PROPERLY
3035      ;*INSTRUCTION SHOULD MOVE IBUS*4 TO IBUS*5, IBUS*4 IS ALL 1'S
3036      ;*AND IBUS*5 IS ALL 0'S. RESULT SHOULD BE ALL 1'S IN SEL4
3037      ;*****
3038
3039      : TEST 31
3040      :-----
3041 015450 012737 000031 001226 TST31: MOV #31,TSTNO
3042 015456 012737 015534 001216 MOV #TST32,NEXT
3043
3044 015464 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3045 015466 012761 000377 000004 MOV #377,4(R1) ;MASTER CLEAR DMC11
3046 015474 012711 001000 MOV #BIT9,(R1) ;PORT4 HI-BYTE=0'S LO-BYTE=1'S
3047 015500 012761 121105 000006 MOV #121105,6(R1) ;SET ROMI
3048 015506 052711 001400 BIS #BIT8!BIT9,(R1) ;INSTRUCTION TO PORT6
3049 015512 000240 NOP ;CLK INSTRUCTION, MOVE IBUS*4 TO IBUS*5
3050 015514 012705 177777 MOV #-1,R5 ;PUT 'EXPECTED' IN R5
3051 015520 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' INTO R4
3052 015524 020504 CMP R5,R4 ;IS DATA CORRECT
3053 015526 001401 BEQ 1$ ;BR IF YES
3054 015530 104003 HLT 3 ;ERROR
3055 015532 104400 1$: SCOPE ;SCOPE THIS TEST
3056
3057
3058      ;***** TEST 32 *****
3059      ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
3060      ;*FLOAT A 1 THROUGH IBUS* REGISTER 0
3061      ;*FLOAT A 0 THROUGH IBUS* REGISTER 0
3062      ;*****
3063
3064      : TEST 32
3065      :-----
3066 015534 012737 000032 001226 TST32: MOV #32,TSTNO
3067 015542 012737 015734 001216 MOV #TST33,NEXT
3068 015550 012737 015570 001220 MOV #64$,LOCK
3069
3070 015556 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3071 015560 012702 000000 MOV #0,R2 ;MASTER CLEAR DMC11
3072 015564 012700 000001 MOV #1,R0 ;SAVE REGISTER ADDRESS FOR TYPEOUT
3073 015570 64$: MOV R0,4(R1) ;START WITH BIT 0
3074 015570 010061 000004 MOV #30,4(R1) ;PUT PATTERN INTO PORT4
3075 015574 042761 000030 000004 BIC #30,4(R1) ;CLEAR UNWANTED BITS
3076 015602 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3077 015604 121100 121100.0 ;MOV DATA TO IBUS* REGISTER 0
3078 015606 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3079 015610 121005 121005.<0*20> ;READ FROM IBUS* REGISTER 0
3080 015612 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3081 015614 042705 000030 BIC #30,R5 ;CLEAR UNWANTED BITS
3082 015620 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
3083 015624 120504 CMPB R5,R4 ;DATA CORRECT?
3084 015626 001401 BEQ 65$ ;BR IF YES
3085 015630 104004 HLT 4 ;ERROR
3086 015632 104401 65$: SCOP1 ;SW09-1?
3087 015634 000241 CLC ;CLEAR CARRY
3088 015636 106100 ROLR R0 ;SHIFT BIT IN R0
3089 015640 001353 BNF 64$ ;IF R0=0 THEN DONE

```

```

3090 015642 012737 015656 001220      MOV    #67$,LOCK      ;NEW SCOP1
3091 015650 012700 000001              MOV    #1,R0        ;START WITH BIT 0
3092 015654 005100                    69$:  COM    R0          ;CHANGE TO FLOATING ZERO
3093 015656                    67$:
3094 015656 010061 000004              MOV    R0,4(R1)      ;PUT PATTERN INTO PORT4
3095 015662 042761 00003C 000004      BIC    #30,4(R1)     ;CLEAR UNWANTED BITS
3096 015670 104414                    ROMCLK  ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3097 015672 121100                    121100!0 ;MOV DATA TO IBUS* REGISTER 0
3098 015674 104414                    ROMCLK  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3099 015676 121005                    121005!<0*20> ;READ FROM IBUS* REGISTER 0
3100 015700 010005                    MOV    R0,R5        ;PUT EXPECTED IN R5
3101 015702 042705 000030              BIC    #30,R5        ;CLEAR UNWANTED BITS
3102 015706 116104 000005              MOV    5(R1),R4     ;PUT 'FOUND' INTO R4
3103 015712 120504                    (MPB    R5,R4        ;DATA CORRECT?
3104 015714 001401                    BEQ    68$          ;BR IF YES
3105 015716 104004                    HLT    4            ;ERROR
3106 015720 104401                    68$:  SCOP1          ;SW09=1?
3107 015722 005100                    COM    R0          ;CHANGE TO FLOATING 1
3108 015724 000241                    CLC           ;CLEAR CARRY
3109 015726 106100                    ROLB   R0          ;SHIFT BIT IN R0
3110 015730 001351                    BNE    69$          ;IF R0=0 THEN DONE
3111 015732 104400                    SCOPE          ;SCOPE THIS TEST
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122 015734 012737 000033 001226  TST33:  MOV    #33,TSTNO
3123 015742 012737 016134 001216      MOV    #TST34,NEXT
3124 015750 012737 015770 001220      MOV    #64$,LOCK
3125
3126 015756 104412                    MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
3127 015760 012702 000002              MOV    #2,R2        ;MASTER CLEAR DMC11
3128 015764 012700 000001              MOV    #1,R0        ;SAVE REGISTER ADDRESS FOR TYPEOUT
3129 015770                    64$:  MOV    R0,4(R1)      ;START WITH BIT 0
3130 015770 010061 000004              BIC    #70,4(R1)     ;PUT PATTERN INTO PORT4
3131 015774 042761 000070 000004      ROMCLK  ;CLEAR UNWANTED BITS
3132 016002 104414                    121100!2 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3133 016004 121102                    ROMCLK  ;MOV DATA TO IBUS* REGISTER 2
3134 016006 104414                    121005.<2*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3135 016010 121045                    MOV    R0,R5        ;READ FROM IBUS* REGISTER 2
3136 016012 010005                    MOV    R0,R5        ;PUT EXPECTED IN R5
3137 016014 042705 000070              BIC    #70,R5        ;CLEAR UNWANTED BITS
3138 016020 116104 000005              MOV    5(R1),R4     ;PUT 'FOUND' INTO R4
3139 016024 120504                    (MPB    R5,R4        ;DATA CORRECT?
3140 016026 001401                    BEQ    65$          ;BR IF YES
3141 016030 104004                    HLT    4            ;ERROR
3142 016032 104401                    65$:  SCOP1          ;SW09-1?
3143 016034 000241                    CLC           ;CLEAR CARRY
3144 016036 106100                    ROLB   R0          ;SHIFT BIT IN R0
3145 016040 001353                    BNE    64$          ;IF R0 0 THEN DONE

```

```

3146 016042 012737 016056 001220      MOV    #67$,LOCK      ;NEW SCOP1
3147 016050 012700 000001              MOV    #1,R0        ;START WITH BIT 0
3148 016054 005100                      COM     R0          ;CHANGE TO FLOATING ZERO
3149 016056                                69$:
3150 016056 010061 000004              MOV    R0,4(R1)      ;PUT PATTERN INTO PORT4
3151 016062 042761 000070 000004      BIC     #70,4(R1)    ;CLEAR UNWANTED BITS
3152 016070 104414                      ROMCLK 121100.2      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3153 016072 121102                      ROMCLK 121005!<2*20> ;MOV DATA TO IBUS* REGISTER 2
3154 016074 104414                      ROMCLK 121005!<2*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3155 016076 121045                      MOV     R0,R5        ;READ FROM IBUS* REGISTER 2
3156 016100 010005                      MOV     R0,R5        ;PUT EXPECTED IN R5
3157 016102 042705 000070              BIC     #70,R5      ;CLEAR UNWANTED BITS
3158 016106 116104 000005              MOV     5(R1),R4    ;PUT 'FOUND' INTO R4
3159 016112 120504                      CMP     R5,R4        ;DATA CORRECT?
3160 016114 001401                      BEQ     68$          ;BR IF YES
3161 016116 104004                      HLT     4            ;ERROR
3162 016120 104401                      68$: SCOP1        ;SW09=1?
3163 016122 005100                      COM     R0          ;CHANGE TO FLOATING 1
3164 016124 000241                      CLC                     ;CLEAR CARRY
3165 016126 106100                      ROL     R0          ;SHIFT BIT IN R0
3166 016130 001351                      BNE     69$          ;IF R0=0 THEN DONE
3167 016132 104400                      SCOPE              ;SCOPE THIS TEST
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178 016134 012737 000034 001226  TST34:  MOV    #34,TSTNO
3179 016142 012737 016310 001216      MOV    #TST35,NEXT
3180 016150 012737 016170 001220      MOV    #64$,LOCK
3181
3182 016156 104412                      MSTCLR              ;R1 CONTAINS BASE DMC11 ADDRESS
3183 016160 012702 000004              MOV    #4,R2        ;MASTER CLEAR DMC11
3184 016164 012700 000001              MOV    #1,R0        ;SAVE REGISTER ADDRESS FOR TYPEOUT
3185 016170                                64$:
3186 016170 010061 000004              MOV    R0,4(R1)      ;START WITH BIT 0
3187 016174 104414                      ROMCLK 121100.4      ;PUT PATTERN INTO PORT4
3188 016176 121104                      ROMCLK 121005!<4*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3189 016200 104414                      ROMCLK 121005!<4*20> ;MOV DATA TO IBUS* REGISTER 4
3190 016202 121105                      MOV     R0,R5        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3191 016204 010005                      MOV     R0,R5        ;READ FROM IBUS* REGISTER 4
3192 016206 116104 000005              MOV     5(R1),R4    ;PUT EXPECTED IN R5
3193 016212 120504                      CMP     R5,R4        ;PUT 'FOUND' INTO R4
3194 016214 001401                      BEQ     65$          ;DATA CORRECT?
3195 016216 104004                      HLT     4            ;BR IF YES
3196 016220 104401                      65$: SCOP1        ;ERROR
3197 016222 000241                      CLC                     ;SW09=1?
3198 016224 106100                      ROL     R0          ;CLEAR CARRY
3199 016226 001360                      BNE     64$          ;SHIFT BIT IN R0
3200 016230 012737 016244 001220      MOV    #64$,LOCK    ;IF R0=0 THEN DONE
3201 016236 012700 000001              MOV    #1,R0        ;NEW SCOP1
                          ;START WITH BIT 0

```

```

3202 016242 005100      69$: COM      R0      ;CHANGE TO FLOATING ZERO
3203 016244      67$:      MOV      R0,4(R1) ;PUT PATTERN INTO PORT4
3204 016244 010061 000004      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3205 016250 104414      121100.4 ;MOV DATA TO IBUS* REGISTER 4
3206 016252 121104      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3207 016254 104414      121005!<4*20> ;READ FROM IBUS* REGISTER 4
3208 016256 121105      MOV      R0,R5 ;PUT EXPECTED IN R5
3209 016260 010005      MOV      5(R1),R4 ;PUT 'FOUND' INTO R4
3210 016262 116104 000005      CMPB     R5,R4 ;DATA CORRECT?
3211 016266 120504      BEQ      68$ ;BR IF YES
3212 016270 001401      HLT      4 ;ERROR
3213 016272 104004      68$: SCOP1 ;SW09=1?
3214 016274 104401      COM      R0 ;CHANGE TO FLOATING 1
3215 016276 005100      CLC      ;CLEAR CARRY
3216 016300 000241      ROLB     R0 ;SHIFT BIT IN R0
3217 016302 106100      BNE      69$ ;IF R0=0 THEN DONE
3218 016304 001356      SCOPE ;SCOPE THIS TEST
3219 016306 104400
3220
3221
3222 ;***** TEST 35 *****
3223 ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
3224 ;*FLOAT A 1 THROUGH IBUS* REGISTER 5
3225 ;*FLOAT A 0 THROUGH IBUS* REGISTER 5
3226 ;*****
3227
3228 ; TEST 35
3229
3230 016310 012737 000035 001226 TST35: MOV      #35,TSTNO
3231 016316 012737 016464 001216      MOV      #TST36,NEXT
3232 016324 012737 016344 001220      MOV      #64$,LOCK
3233
3234 016332 104412      MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3235 016334 012702 000005      MOV      #5,R2 ;MASTER CLEAR DMC11
3236 016340 012700 000001      MOV      #1,R0 ;SAVE REGISTER ADDRESS FOR TYPEOUT
3237 016344      64$: ;START WITH BIT 0
3238 016344 010061 000004      MOV      R0,4(R1) ;PUT PATTERN INTO PORT4
3239 016350 104414      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3240 016352 121105      121100.5 ;MOV DATA TO IBUS* REGISTER 5
3241 016354 104414      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3242 016356 121125      121005!<5*20> ;READ FROM IBUS* REGISTER 5
3243 016360 010005      MOV      R0,R5 ;PUT EXPECTED IN R5
3244 016362 116104 000005      MOV      5(R1),R4 ;PUT 'FOUND' INTO R4
3245 016366 120504      CMPB     R5,R4 ;DATA CORRECT?
3246 016370 001401      BEQ      65$ ;BR IF YES
3247 016372 104004      HLT      4 ;ERROR
3248 016374 104401      65$: SCOP1 ;SW09=1?
3249 016376 000241      CLC      ;CLEAR CARRY
3250 016400 106100      ROLB     R0 ;SHIFT BIT IN R0
3251 016402 001360      BNE      64$ ;IF R0=0 THEN DONE
3252 016404 012737 016420 001220      MOV      #67$,LOCK ;NEW SCOP1
3253 016412 012700 000001      MOV      #1,R0 ;START WITH BIT 0
3254 016416 005100      64$: COM      R0 ;CHANGE TO FLOATING ZERO
3255 016420      67$:
3256 016420 010061 000004      MOV      R0,4(R1) ;PUT PATTERN INTO PORT4
3257 016424 104414      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```

```

3258 016426 121105      121100!5      :MOV DATA TO IBUS* REGISTER 5
3259 016430 104414      ROMCLK      :NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
3260 016432 121125      121005.<5*20> :READ FROM IBUS* REGISTER 5
3261 016434 010005      MOV R0,R5    :PUT EXPECTED IN R5
3262 016436 116104 000005 :MOVSB 5(R1),R4 :PUT 'FOUND' INTO R4
3263 016442 120504      CMPB R5,R4   :DATA CORRECT?
3264 016444 001401      BEQ 68$      :BR IF YES
3265 016446 104004      HLT 4        :ERROR
3266 016450 104401      68$: SCOP1    :SW09=1?
3267 016452 005100      COM R0       :CHANGE TO FLOATING 1
3268 016454 000241      CLC         :CLEAR CARRY
3269 016456 106100      ROLB R0      :SHIFT BIT IN R0
3270 016460 001356      BNE 69$      :IF R0=0 THEN DONE
3271 016462 104400      SCOPE        :SCOPE THIS TEST
3272
3273
3274 :***** TEST 36 *****
3275 :*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
3276 :*FLOAT A * THROUGH IBUS* REGISTER 10
3277 :*FLOAT A 0 THROUGH IBUS* REGISTER 10
3278 :*THE NPR RQ BIT (BIT 0) IS MASKED DURING THIS TEST
3279 :*****
3280
3281 : TEST 36
3282 :-----
3283 016464 012737 000036 001226 TS 36: MOV #36,ISTNO
3284 016472 012737 016664 001216 MOV #TST37,NEXT
3285 016500 012737 016520 001220 MOV #64$,LOCK
3286
3287 016506 104412      MSTCLR      :R1 CONTAINS BASE DMC11 ADDRESS
3288 016510 012702 000010      MOV #10,R2 :MASTER CLEAR DMC11
3289 016514 012700 000001      MOV #1,R0  :SAVE REGISTER ADDRESS FOR TYPEOUT
3290 016520      64$:          :START WITH BIT 0
3291 016520 010061 000004      MOV R0,4(R1) :PUT PATTERN INTO PORT4
3292 016524 042761 000141 000004 BIC #141,4(R1) :CLEAR UNWANTED BITS
3293 016532 104414      ROMCLK      :NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
3294 016534 121110      121100!10    :MOV DATA TO IBUS* REGISTER 10
3295 016536 104414      ROMCLK      :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3296 016540 121205      121005!<10*20> :READ FROM IBUS* REGISTER 10
3297 016542 010005      MOV R0,R5    :PUT EXPECTED IN R5
3298 016544 042705 000141      BIC #141,R5 :CLEAR UNWANTED BITS
3299 016550 116104 000005      MOVSB 5(R1),R4 :PUT 'FOUND' INTO R4
3300 016554 120504      CMPB R5,R4   :DATA CORRECT?
3301 016556 001401      BEQ 65$      :BR IF YES
3302 016560 104004      HLT 4        :ERROR
3303 016562 104401      65$: SCOP1    :SW09=1?
3304 016564 000241      CLC         :CLEAR CARRY
3305 016566 106100      ROLB R0      :SHIFT BIT IN R0
3306 016570 001353      BNE 64$      :IF R0=0 THEN DONE
3307 016572 012737 016606 001220      MOV #67$,LOCK :NEW SCOPE
3308 016600 012700 000001      MOV #1,R0  :START WITH BIT 0
3309 016604 005100      69$: COM R0    :CHANGE TO FLOATING ZERO
3310 016606      67$:
3311 016606 010061 000004      MOV R0,4(R1) :PUT PATTERN INTO PORT4
3312 016612 042761 000141 000004 BIC #141,4(R1) :CLEAR UNWANTED BITS
3313 016620 104414      ROMCLK      :NEXT WORD IS INSTRUCTION, ROMCLK PC-5304

```

3314	016622	121110				121100.10	:MOV DATA TO IBUS* REGISTER 10
3315	016624	104414				ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3316	016626	121205				121005!<10*20>	:READ FROM IBUS* REGISTER 10
3317	016630	010005				MOV R0,R5	:PUT EXPECTED IN R5
3318	016632	042705	000141			BIC #141,R5	:CLEAR UNWANTED BITS
3319	016636	116104	000005			MOVB 5(R1),R4	:PUT 'FOUND' INTO R4
3320	016642	120504				CMPS R5,R4	:DATA CORRECT?
3321	016644	001401				BEQ 68\$	:BR IF YES
3322	016646	104004				HLT 4	:ERROR
3323	016650	104401			68\$:	SCOP1	:SW09-1?
3324	016652	005100				COM R0	:CHANGE TO FLOATING 1
3325	016654	000241				CLC	:CLEAR CARRY
3326	016656	106100				ROLB R0	:SHIFT BIT IN R0
3327	016660	001351				BNE 69\$	:IF R0 0 THEN DONE
3328	016662	104400				SCOPE	:SCOPE THIS TEST
3329							
3330							
3331						:***** TEST 37 *****	
3332						:*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST	
3333						:*FLOAT A 1 THROUGH IBUS* REGISTER 11	
3334						:*FLOAT A 0 THROUGH IBUS* REGISTER 11	
3335						:*THE BR RQ BIT, PGM CLOCK BIT, FORCE POWER FAIL BIT	
3336						:*(BITS 7,4,1) ARE ALL MASKED DURING THIS TEST	
3337						:*****	
3338							
3339						: TEST 37	
3340						-----	
3341	016664	012737	000037	001226	TST37:	MOV #37,TSTNO	
3342	016672	012737	017104	001216		MOV #TST40,NEXT	
3343	016700	012737	016720	001220		MOV #64\$,LOCK	
3344							
3345	016706	104412				MSTCLR	:R1 CONTAINS BASE DMC11 ADDRESS
3346	016710	012702	000011			MOV #11,R2	:MASTER CLEAR DMC11
3347	016714	012700	000001			MOV #1,R0	:SAVE REGISTER ADDRESS FOR TYPEOUT
3348	016720				64\$:		:START WITH BIT 0
3349	016720	010061	000004			MOV R0,4(R1)	:PUT PATTERN INTO PORT4
3350	016724	042761	000262	000004		BIC #262,4(R1)	:CLEAR UNWANTED BITS
3351	016732	104414				ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3352	016734	121111				121100 11	:MOV DATA TO IBUS* REGISTER 11
3353	016736	104414				ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3354	016740	121225				121005!<11*20>	:READ FROM IBUS* REGISTER 11
3355	016742	010005				MOV R0,R5	:PUT EXPECTED IN R5
3356	016744	042705	000262			BIC #262,R5	:CLEAR UNWANTED BITS
3357	016750	052705	000020			BIS #20,R5	:ADD THESE BITS
3358	016754	116104	000005			MOVB 5(R1),R4	:PUT 'FOUND' INTO R4
3359	016760	052704	000020			BIS #20,R4	:ADD THIS BIT
3360	016764	120504				CMPS R5,R4	:DATA CORRECT?
3361	016766	001401				BEQ 65\$	:BR IF YES
3362	016770	104004				HLT 4	:ERROR
3363	016772	104401			65\$:	SCOP1	:SW09=1?
3364	016774	000241				CLC	:CLEAR CARRY
3365	016776	106100				ROLB R0	:SHIFT BIT IN R0
3366	017000	001347				BNE 64\$	:IF R0=0 THEN DONE
3367	017002	012737	017016	001220		MOV #64\$,LC	:NEW SCOP1
3368	017010	012700	000001			MOV #,R0	:START WITH BIT 0
3369	017014	005100			69\$:	COM R0	:CHANGE TO FLOATING ZERO


```
3370 017016
3371 017016 010061 000004
3372 017022 042761 000262 000004
3373 017030 104414
3374 017032 121111
3375 017034 104414
3376 017036 121225
3377 017040 010005
3378 017042 042705 000262
3379 017046 052705 000020
3380 017052 116104 000005
3381 017056 052704 000020
3382 017062 120504
3383 017064 001401
3384 017066 104004
3385 017070 104401
3386 017072 005100
3387 017074 000241
3388 017076 106100
3389 017100 001345
3390 017102 104400
3391
3392
3393
3394
3395
3396
3397
3398
3399
3400
3401 017104 012737 000040 001226 1ST40:
3402 017112 012737 017260 001216
3403 017120 012737 017140 001220
3404
3405 017126 104412
3406 017130 012702 000000
3407 017134 012700 000001
3408 017140
3409 017140 010061 000004
3410 017144 104414
3411 017146 122100
3412 017150 104414
3413 017152 021005
3414 017154 010005
3415 017156 116104 000005
3416 017162 120504
3417 017164 001401
3418 017166 104005
3419 017170 104401
3420 017172 000241
3421 017174 106100
3422 017176 001360
3423 017200 012737 017214 001220
3424 017206 012700 000001
3425 017212 005100

67$:
MOV R0,4(R1) ;PUT PATTERN INTO PORT4
BIC #262,4(R1) ;CLEAR UNWANTED BITS
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
121100.11 ;MOV DATA TO IBUS* REGISTER 11
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
121005!<11*20> ;READ FROM IBUS* REGISTER 11
MOV R0,R5 ;PUT EXPECTED IN R5
BIC #262,R5 ;CLEAR UNWANTED BITS
BIS #20,R5 ;ADD THESE BITS
MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
BIS #20,R4 ;ADD THIS BIT
CMPB R5,R4 ;DATA CORRECT?
BEQ 68$ ;BR IF YES
HLT 4 ;ERROR
68$: SCOP1 ;SW09=1?
COM R0 ;CHANGE TO FLOATING 1
CLC ;CLEAR CARRY
ROLB R0 ;SHIFT BIT IN R0
BNE 69$ ;IF R0=0 THEN DONE
SCOPE ;SCOPE THIS TEST

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

; TEST 40
;-----
MOV #40,TSTNO
MOV #TST41,NEXT
MOV #64$,LOCK
;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SAVE REGISTER ADDRESS FOR TYPEOUT
;START WITH BIT 0
64$: MOV R0,4(R1) ;PUT PATTERN INTO PORT4
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122100!0 ;MOV DATA TO IBUS REGISTER 0
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
21005!<0*20> ;READ FROM IBUS REGISTER 0
MOV R0,R5 ;PUT EXPECTED IN R5
MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
CMPB R5,R4 ;DATA CORRECT?
BEQ 65$ ;BR IF YES
HLT 5 ;ERROR
65$: SCOP1 ;SW09=1?
CLC ;CLEAR CARRY
ROLB R0 ;SHIFT BIT IN R0
BNE 64$ ;IF R0=0 THEN DONE
MOV #67$,LOCK ;NEW SCOP1
MOV #,R0 ;START WITH BIT 0
69$: COM R0 ;CHANGE TO FLOATING ZERO
```

```

3426 017214
3427 017214 010061 000004
3428 017220 104414
3429 017222 122100
3430 017224 104414
3431 017226 021005
3432 017230 010005
3433 017232 116104 000005
3434 017236 120504
3435 017240 001401
3436 017242 104005
3437 017244 104401
3438 017246 005100
3439 017250 000241
3440 017252 106100
3441 017254 001356
3442 017256 104400
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453 017260 012737 000041 001226 TST41:
3454 017266 012737 017434 001215
3455 017274 012737 017314 001220
3456
3457 017302 104412
3458 017304 012702 000001
3459 017310 012700 000001
3460 017314
3461 017314 010061 000004
3462 017320 104414
3463 017322 122101
3464 017324 104414
3465 017326 021025
3466 017330 010005
3467 017332 116104 000005
3468 017336 120504
3469 017340 001401
3470 017342 104005
3471 017344 104401
3472 017346 000241
3473 017350 106100
3474 017352 001360
3475 017354 012737 017370 001220
3476 017362 012700 000001
3477 017366 005100
3478 017370
3479 017370 010061 000004
3480 017374 104414
3481 017376 122101

67$:
MOV R0,4(R1) ;PUT PATTERN INTO PORT4
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
122100!0 ;MOV DATA TO IBUS REGISTER 0
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
21005!<0*20> ;READ FROM IBUS REGISTER 0
MOV R0,R5 ;PUT EXPECTED IN R5
MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
CMPB R5,R4 ;DATA CORRECT?
BEQ 68$ ;BR IF YES
HLT 5 ;ERROR
68$: SCOP1 ;SW09=1?
COM R0 ;CHANGE TO FLOATING 1
CLC ;CLEAR CARRY
ROLB R0 ;SHIFT BIT IN R0
BNE 69$ ;IF R0=0 THEN DONE
SCOPE ;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
:-----
MOV #41,TSTNO
MOV #TST42,NEXT
MOV #64$,LOCK

MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
MOV #1,R2 ;MASTER CLEAR DMC11
MOV #1,R0 ;SAVE REGISTER ADDRESS FOR TYPEOUT
;START WITH BIT 0

64$: MOV R0,4(R1) ;PUT PATTERN INTO PORT4
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122100.1 ;MOV DATA TO IBUS REGISTER 1
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
21005.<1*20> ;READ FROM IBUS REGISTER 1
MOV R0,R5 ;PUT EXPECTED IN R5
MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
CMPB R5,R4 ;DATA CORRECT?
BEQ 65$ ;BR IF YES
HLT 5 ;ERROR
65$: SCOP1 ;SW09=1?
CLC ;CLEAR CARRY
ROLB R0 ;SHIFT BIT IN R0
BNE 64$ ;IF R0=0 THEN DONE
MOV #67$,LOCK ;NEW SCOP1
MOV #1,R0 ;START WITH BIT 0
69$: COM R0 ;CHANGE TO FLOATING ZERO
67$: MOV R0,4(R1) ;PUT PATTERN INTO PORT4
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
122100.1 ;MOV DATA TO IBUS REGISTER 1

```



```
3482 017400 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3483 017402 021025 21005.<1*20> ;READ FROM IBUS REGISTER 1
3484 017404 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3485 017406 116104 000005 MOV 5(R1),R4 ;PUT 'FOUND' INTO R4
3486 017412 120504 CMPB R5,R4 ;DATA CORRECT?
3487 017414 001401 BEQ 68$ ;BR IF YES
3488 017416 104005 HLT 5 ;ERROR
3489 017420 104401 68$: SCOP1 ;SW09=1?
3490 017422 005100 COM R0 ;CHANGE TO FLOATING 1
3491 017424 000241 CLC ;CLEAR CARRY
3492 017426 106100 ROLB R0 ;SHIFT BIT IN R0
3493 017430 001356 BNE 69$ ;IF R0=0 THEN DONE
3494 017432 104400 SCOPE ;SCOPE THIS TEST
3495
3496
3497 ;***** TEST 42 *****
3498 ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3499 ;*FLOAT A 1 THROUGH IBUS REGISTER 2
3500 ;*FLOAT A 0 THROUGH IBUS REGISTER 2
3501 ;*****
3502
3503 ; TEST 42
3504 ;-----
3505 017434 012737 000042 001226 TST42: MOV #42,TSTNO
3506 017442 012737 017610 001216 MOV #TST43,NEXT
3507 017450 012737 017470 001220 MOV #64$,LOCK
3508
3509 017456 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3510 017460 012702 000002 MOV #2,R2 ;MASTER CLEAR DMC11
3511 017464 012700 000001 MOV #1,R0 ;SAVE REGISTER ADDRESS FOR TYPEOUT
3512 017470 64$: ;START WITH BIT 0
3513 017470 010061 000004 MOV R0,4(R1) ;PUT PATTERN INTO PORT4
3514 017474 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3515 017476 122102 122100.2 ;MOV DATA TO IBUS REGISTER 2
3516 017500 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3517 017502 021045 21005.<2*20> ;READ FROM IBUS REGISTER 2
3518 017504 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3519 017506 116104 000005 MOV 5(R1),R4 ;PUT 'FOUND' INTO R4
3520 017512 120504 CMPB R5,R4 ;DATA CORRECT?
3521 017514 001401 BEQ 65$ ;BR IF YES
3522 017516 104005 HLT 5 ;ERROR
3523 017520 104401 65$: SCOP1 ;SW09=1?
3524 017522 000241 CLC ;CLEAR CARRY
3525 017524 106100 ROLB R0 ;SHIFT BIT IN R0
3526 017526 001360 BNE 64$ ;IF R0=0 THEN DONE
3527 017530 012737 017544 001220 MOV #67$,LOCK ;NEW SCOP1
3528 017536 012700 000001 MOV #1,R0 ;START WITH BIT 0
3529 017542 005100 69$: COM R0 ;CHANGE TO FLOATING ZERO
3530 017544 67$:
3531 017544 010061 000004 MOV R0,4(R1) ;PUT PATTERN INTO PORT4
3532 017550 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3533 017552 122102 122100.2 ;MOV DATA TO IBUS REGISTER 2
3534 017554 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3535 017556 021045 21005.<2*20> ;READ FROM IBUS REGISTER 2
3536 017560 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3537 017562 116104 000005 MOV 5(R1),R4 ;PUT 'FOUND' INTO R4
```

```

3538 017566 120504      CMPB    R5,R4      ;DATA CORRECT?
3539 017570 001401      BEQ     68$      ;BR IF YES
3540 017572 104005      HLT     5          ;ERROR
3541 017574 104401      68$:    SCOP1    ;SW09=1?
3542 017576 005100      COM     R0      ;CHANGE TO FLOATING 1
3543 017600 000241      CLC          ;CLEAR CARRY
3544 017602 106100      ROLB    R0      ;SHIFT BIT IN R0
3545 017604 001356      BNE     69$      ;IF R0=0 THEN DONE
3546 017606 104400      SCOPE    ;SCOPE THIS TEST
3547
3548
3549
3550      ;***** TEST 43 *****
3551      ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3552      ;*FLOAT A 1 THROUGH IBUS REGISTER 3
3553      ;*FLOAT A 0 THROUGH IBUS REGISTER 3
3554      ;*****
3555      : TEST 43
3556      :-----
3557 017610 012737 000043 001226 TST43: MOV     #43,TSTNO
3558 017616 012737 017764 001216 MOV     #TST44,NEXT
3559 017624 012737 017644 001220 MOV     #64$,LOCK
3560
3561 017632 104412      MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
3562 017634 012702 000003      MOV     #3,R2      ;MASTER CLEAR DMC11
3563 017640 012700 000001      MOV     #1,R0      ;SAVE REGISTER ADDRESS FOR TYPEOUT
3564 017644      64$:    MOV     R0,4(R1)      ;START WITH BIT 0
3565 017644 010061 000004      ROMCLK    ;PUT PATTERN INTO PORT4
3566 017650 104414      122100 3      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3567 017652 122103      ROMCLK    ;MOV DATA TO IBUS REGISTER 3
3568 017654 104414      21005!<3*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3569 017656 021065      MOV     R0,R5      ;READ FROM IBUS REGISTER 3
3570 017660 010005      MOV     5(R1),R4    ;PUT EXPECTED IN R5
3571 017662 116104 000005      CMPB    R5,R4    ;PUT 'FOUND' INTO R4
3572 017666 120504      BEQ     65$      ;DATA CORRECT?
3573 017670 001401      HLT     5          ;BR IF YES
3574 017672 104005      65$:    SCOP1    ;ERROR
3575 017674 104401      CLC          ;SW09=1?
3576 017676 000241      ROLB    R0      ;CLEAR CARRY
3577 017700 106100      BNE     64$      ;SHIFT BIT IN R0
3578 017702 001360      MOV     #67$,LOCK    ;IF R0=0 THEN DONE
3579 017704 012737 017720 001220 MOV     #1,R0      ;NEW SCOPE
3580 017712 012700 000001      COM     R0      ;START WITH BIT 0
3581 017716 005100      60$:    MOV     R0,4(R1) ;CHANGE TO FLOATING ZERO
3582 017720      61$:    ROMCLK    ;PUT PATTERN INTO PORT4
3583 017720 010061 000004      122100 3      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3584 017724 104414      ROMCLK    ;MOV DATA TO IBUS REGISTER 3
3585 017726 122103      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3586 017730 104414      21005!<3*20> ;READ FROM IBUS REGISTER 3
3587 017732 021065      MOV     R0,R5      ;PUT EXPECTED IN R5
3588 017734 010005      MOV     5(R1),R4    ;PUT 'FOUND' INTO R4
3589 017736 116104 000005      CMPB    R5,R4    ;DATA CORRECT?
3590 017742 120504      BEQ     68$      ;BR IF YES
3591 017744 001401      HLT     5          ;ERROR
3592 017746 104005      68$:    SCOP1    ;SW09=1?
3593 017750 104401

```

```

3594 017752 005100 COM R0 ;CHANGE TO FLOATING 1
3595 017754 000241 CLC ;CLEAR CARRY
3596 017756 106100 ROLB R0 ;SHIFT BIT IN R0
3597 017760 001356 BNE 69$ ;IF R0=0 THEN DONE
3598 017762 104400 SCOPE ;SCOPE THIS TEST
3599
3600
3601 ;***** TEST 44 *****
3602 ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3603 ;*FLOAT A 1 THROUGH IBUS REGISTER 4
3604 ;*FLOAT A 0 THROUGH IBUS REGISTER 4
3605 ;*****
3606
3607 ; TEST 44
3608 ;-----
3609 017764 012737 000044 001226 TST44: MOV #44,TSTNO
3610 017772 012737 020140 001216 MOV #TST45,NEXT
3611 020000 012737 020020 001220 MOV #64$,LOCK
3612
3613 020006 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3614 020010 012702 000004 ;MASTER CLEAR DMC11
3615 020014 012700 000001 MOV #4,R2 ;SAVE REGISTER ADDRESS FOR TYPEOUT
3616 020020 64$: MOV #1,R0 ;START WITH BIT 0
3617 020020 010061 000004 MOV R0,4(R1) ;PUT PATTERN INTO PORT4
3618 020024 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
3619 020026 122104 122100!4 ;MOV DATA TO IBUS REGISTER 4
3620 020030 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
3621 020032 021105 21005.<4*20> ;READ FROM IBUS REGISTER 4
3622 020034 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3623 020036 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
3624 020042 120504 CMPB R5,R4 ;DATA CORRECT?
3625 020044 001401 BEQ 65$ ;BR IF YES
3626 020046 104005 HLT 5 ;ERROR
3627 020050 104401 65$: SCOP1 ;SW09-1?
3628 020052 000241 CLC ;CLEAR CARRY
3629 020054 106100 ROLB R0 ;SHIFT BIT IN R0
3630 020056 001360 BNE 64$ ;IF R0=0 THEN DONE
3631 020060 012737 020074 001220 MOV #67$,LOCK ;NEW SCOP1
3632 020066 012700 000001 MOV #1,R0 ;START WITH BIT 0
3633 020072 005100 69$: COM R0 ;CHANGE TO FLOATING ZERO
3634 020074 67$:
3635 020074 010061 000004 MOV R0,4(R1) ;PUT PATTERN INTO PORT4
3636 020100 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
3637 020102 122104 122100.4 ;MOV DATA TO IBUS REGISTER 4
3638 020104 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3639 020106 021105 21005.<4*20> ;READ FROM IBUS REGISTER 4
3640 020110 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3641 020112 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
3642 020116 120504 CMPB R5,R4 ;DATA CORRECT?
3643 020120 001401 BEQ 68$ ;BR IF YES
3644 020122 104005 HLT 5 ;ERROR
3645 020124 104401 68$: SCOP1 ;SW09=1?
3646 020126 005100 COM R0 ;CHANGE TO FLOATING 1
3647 020130 000241 CLC ;CLEAR CARRY
3648 020132 106100 ROLB R0 ;SHIFT BIT IN R0
3649 020134 001356 BNE 64$ ;IF R0=0 THEN DONE

```

```

3650 020136 104400 SCOPE ;SCOPE THIS TEST
3651
3652
3653 ***** TEST 45 *****
3654 *MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3655 *FLOAT A 1 THROUGH IBUS REGISTER 5
3656 *FLOAT A 0 THROUGH IBUS REGISTER 5
3657 *****
3658
3659 ; TEST 45
3660 -----
3661 020140 012737 000045 001226 TST45: MOV #45,TSTNO
3662 020146 012737 020314 001216 MOV #TST46,NEXT
3663 020154 012737 020174 001220 MOV #64$,LOCK
3664
3665 020162 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3666 020164 012702 000005 MCV #5,R2 ;MASTER CLEAR DMC11
3667 020170 012700 000001 MOV #1,R0 ;SAVE REGISTER ADDRESS FOR TYPEOUT
3668 020174 64$: ;START WITH BIT 0
3669 020174 010061 000004 MOV R0,4(R1) ;PUT PATTERN INTO PORT4
3670 020200 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3671 020202 122105 122100.5 ;MOV DATA TO IBUS REGISTER 5
3672 020204 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3673 020206 021125 21005.<5*20> ;READ FROM IBUS REGISTER 5
3674 020210 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3675 020212 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
3676 020216 120504 CMPB R5,R4 ;DATA CORRECT?
3677 020220 001401 BEQ 65$ ;BR IF YES
3678 020222 104005 HLT 5 ;ERROR
3679 020224 104401 65$: SCOP1 ;SW09-1?
3680 020226 000241 CLC ;CLEAR CARRY
3681 020230 106100 ROLB R0 ;SHIFT BIT IN R0
3682 020232 001360 BNE 64$ ;IF R0=0 THEN DONE
3683 020234 012737 020250 001220 MOV #67$,LOCK ;NEW SCOP1
3684 020242 012700 000001 MOV #1,R0 ;START WITH BIT 0
3685 020246 005100 64$: COM R0 ;CHANGE TO FLOATING ZERO
3686 020250 67$:
3687 020250 010061 000004 MOV R0,4(R1) ;PUT PATTERN INTO PORT4
3688 020254 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3689 020256 122105 122100.5 ;MOV DATA TO IBUS REGISTER 5
3690 020260 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3691 020262 021125 21005.<5*20> ;READ FROM IBUS REGISTER 5
3692 020264 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3693 020266 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
3694 020272 120504 CMPB R5,R4 ;DATA CORRECT?
3695 020274 001401 BEQ 68$ ;BR IF YES
3696 020276 104005 HLT 5 ;ERROR
3697 020300 104401 68$: SCOP1 ;SW09-1?
3698 020302 005100 COM R0 ;CHANGE TO FLOATING 1
3699 020304 000241 CLC ;CLEAR CARRY
3700 020306 106100 ROLB R0 ;SHIFT BIT IN R0
3701 020310 001356 BNE 69$ ;IF R0=0 THEN DONE
3702 020312 104400 SCOPE ;SCOPE THIS TEST
3703
3704
3705 ***** TEST 46 *****

```

```

3706      : *MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3707      : *FLOAT A 1 THROUGH IBUS REGISTER 6
3708      : *FLOAT A 0 THROUGH IBUS REGISTER 6
3709      : *****
3710
3711      : TEST 46
3712      : -----
3713 020314 012737 000046 001226 TST46: MOV #46,TSTNO
3714 020322 012737 020470 001216 MOV #TST47,NEXT
3715 020330 012737 320350 001220 MOV #64$,LOCK
3716
3717 020336 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3718 020340 012702 000006 ;MASTER CLEAR DMC11
3719 020344 012700 000001 ;SAVE REGISTER ADDRESS FOR TYPEOUT
3720 020350 ;START WITH BIT 0
3721 020350 010061 000004 64$: MOV R0,4(R1) ;PUT PATTERN INTO PORT4
3722 020354 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3723 020356 122106 122100:6 ;MOV DATA TO IBUS REGISTER 6
3724 020360 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3725 020362 021145 21005!<6*20> ;READ FROM IBUS REGISTER 6
3726 020364 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3727 020366 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
3728 020372 120504 CMPS R5,R4 ;DATA CORRECT?
3729 020374 001401 BEQ 65$ ;BR IF YES
3730 020376 104005 HLT 5 ;ERROR
3731 020400 104401 65$: SCOP1 ;SW09=1?
3732 020402 000241 CLC ;CLEAR CARRY
3733 020404 106100 ROLB R0 ;SHIFT BIT IN R0
3734 020406 001360 BNE 64$ ;IF R0=0 THEN DONE
3735 020410 012737 020424 001220 MOV #67$,LOCK ;NEW SCOP1
3736 020416 012700 000001 MOV #1,R0 ;START WITH BIT 0
3737 020422 005100 66$: COM R0 ;CHANGE TO FLOATING ZERO
3738 020424 67$:
3739 020424 010061 000004 MOV R0,4(R1) ;PUT PATTERN INTO PORT4
3740 020430 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3741 020432 122106 122100:6 ;MOV DATA TO IBUS REGISTER 6
3742 020434 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3743 020436 021145 21005!<6*20> ;READ FROM IBUS REGISTER 6
3744 020440 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3745 020442 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
3746 020446 120504 CMPS R5,R4 ;DATA CORRECT?
3747 020450 001401 BEQ 68$ ;BR IF YES
3748 020452 104005 HLT 5 ;ERROR
3749 020454 104401 68$: SCOP1 ;SW09=1?
3750 020456 005100 COM R0 ;CHANGE TO FLOATING 1
3751 020460 000241 CLC ;CLEAR CARRY
3752 020462 106100 ROLB R0 ;SHIFT BIT IN R0
3753 020464 001356 BNE 69$ ;IF R0=0 THEN DONE
3754 020466 104400 SCOP2 ;SCOPE THIS TEST
3755
3756
3757      : ***** TEST 47 *****
3758      : *MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3759      : *FLOAT A 1 THROUGH IBUS REGISTER 7
3760      : *FLOAT A 0 THROUGH IBUS REGISTER 7
3761      : *****
  
```

```

3762
3763
3764      : TEST 47
3765      :-----
3765 020470 012737 000047 001226 1ST47: MOV #47,TSTNO
3766 020476 012737 020644 001216 MOV #TST50,NEXT
3767 020504 012737 020524 001220 MOV #64$,LOCK
3768
3769 020512 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3770 020514 012702 000007 ;MASTER CLEAR DMC11
3771 020520 012700 000001 MOV #7,R2 ;SAVE REGISTER ADDRESS FOR TYPEOUT
3772 020524 64$: MOV #1,R0 ;START WITH BIT 0
3773 020524 010061 000004 MOV R0,4(R1) ;PUT PATTERN INTO PORT4
3774 020530 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3775 020532 122107 122100.7 ;MOV DATA TO IBUS REGISTER 7
3776 020534 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3777 020536 021165 21005.<7*20> ;READ FROM IBUS REGISTER 7
3778 020540 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3779 020542 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
3780 020546 120504 CMPB R5,R4 ;DATA CORRECT?
3781 020550 001401 BEQ 65$ ;BR IF YES
3782 020552 104005 HLT 5 ;ERROR
3783 020554 104401 65$: SCOP1 ;SW09-1?
3784 020556 000241 CLC ;CLEAR CARRY
3785 020560 106100 ROLB R0 ;SHIFT BIT IN R0
3786 020562 001360 BNE 64$ ;IF R0=0 THEN DONE
3787 020564 012737 020600 001220 MOV #67$,LOCK ;NEW SCOP1
3788 020572 012700 000001 MOV #1,R0 ;START WITH BIT 0
3789 020576 005100 67$: COM R0 ;CHANGE TO FLOATING ZERO
3790 020600 67$:
3791 020600 010061 000004 MOV R0,4(R1) ;PUT PATTERN INTO PORT4
3792 020604 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3793 020606 122107 122100.7 ;MOV DATA TO IBUS REGISTER 7
3794 020610 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3795 020612 021165 21005.<7*20> ;READ FROM IBUS REGISTER 7
3796 020614 010005 MOV R0,R5 ;PUT EXPECTED IN R5
3797 020616 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' INTO R4
3798 020622 120504 CMPB R5,R4 ;DATA CORRECT?
3799 020624 001401 BEQ 68$ ;BR IF YES
3800 020626 104005 HLT 5 ;ERROR
3801 020630 104401 68$: SCOP1 ;SW09=1?
3802 020632 005100 COM R0 ;CHANGE TO FLOATING 1
3803 020634 000241 CLC ;CLEAR CARRY
3804 020636 106100 ROLB R0 ;SHIFT BIT IN R0
3805 020640 001356 BNE 69$ ;IF R0=0 THEN DONE
3806 020642 104400 SCOPE ;SCOPE THIS TEST
3807
3808
3809
3810      :***** TEST 50 *****
3811      :*MICRO PROCESSOR IBUS DUAL ADDRESS TEST
3812      :*WRITE ALL IBUS REGISTERS WITH INCREMENTING PATTERN
3813      :*READ ALL IBUS REGISTERS TO VERIFY CORRECT ADDRESSING
3814      :*****
3815
3816      : TEST 50
3817      :-----
3817 020644 012737 000050 001226 1ST50: MOV #50,TSTNO
  
```



```

3818 020652 012737 021072 001216      MOV      #TST51,NEXT
3819 020660 012737 020676 001220      MOV      #1$,LOCK
3820                                     ;R1 CONTAINS BASE DMC11 ADDRESS
3821 020666 104412                                     ;MASTER CLEAR DMC11
3822 020670 012700 000001      MOV      #1,R0      ;START WITH A ONE
3823 020674 005002      CLR      R2      ;R2 CONTAINS ADDRESS OF REGISTER
3824 020676 010203 1$:      MOV      R2,R3      ;R3=REGISTER ADDRESS
3825 020700 010061 000004      MOV      R0,4(R1)    ;WRITE DATA TO PORT4
3826 020704 042737 000017 020720      BIC      #17,5$    ;CLEAR ADDRESS FIELD OF INSTRUCTION
3827 020712 050337 020720      BIS      R3,5$    ;ADD ADDRESS TO INSTRUCTION
3828 020716 104414      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3829 020720 122100 5$:      122100    ;MOVE DATA TO IBUS REGISTER
3830 020722 006303      ASL      R3      ;SHIFT ADDRESS
3831 020724 006303      ASL      R3      ;4 TIMES TO GET
3832 020726 006303      ASL      R3      ;IT TO BITS 4-7
3833 020730 006303      ASL      R3      ;OF NEXT INSTRUCTION
3834 020732 042737 000360 020746      BIC      #360,6$    ;CLEAR ADDRESS FIELD
3835 020740 050337 020746      BIS      R3,6$    ;ADD ADDRESS TO INSTRUCTION
3836 020744 104414      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3837 020746 021005 6$:      21005    ;READ FROM IBUS REGISTER
3838 020750 010005      MOV      R0,R5      ;PUT 'EXPECTED' IN R5
3839 020752 116104 000005      MOVB     5(R1),R4    ;PUT 'FOUND' IN R4
3840 020756 120504      CMPB     R5,R4      ;IS DATA CORRECT?
3841 020760 001401      BEQ      2$      ;BR IF YES
3842 020762 104005      HLT      5      ;DATA ERROR
3843 020764 104401 2$:      SCOP1    ;SW09=1?
3844 020766 005200      INC      R0      ;INCREMENT PATTERN
3845 020770 005202      INC      R2      ;INCREMENT REGISTER ADDRESS
3846 020772 022702 000010      CMP      #7+1,R2 ;LAST ADDRESS DONE?
3847 020776 001337      BNE      1$      ;BR IF NO
3848 021000 012737 021016 001220      MOV      #3$,LOCK    ;NEW SCOP1
3849 021006 012700 000001      MOV      #1,R0      ;RESTART PATTERN TO 1
3850 021012 005002      CLR      R2      ;RESTART AT ADDRESS 0
3851 021014 005003      CLR      R3      ;RESTART AT ADDRESS 0
3852 021016 042737 000360 021032 3$:      BIC      #360,7$    ;CLEAR ADDRESS FIELD OF INSTRUCTION
3853 021024 050337 021032      BIS      R3,7$    ;ADD ADDRESS TO INSTRUCTION
3854 021030 104414      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
3855 021032 021005 7$:      21005    ;READ FROM IBUS REGISTER
3856 021034 010005      MOV      R0,R5      ;PUT 'EXPECTED' IN R5
3857 021036 116104 000005      MOVB     5(R1),R4    ;PUT 'FOUND' IN R5
3858 021042 120504      CMPB     R5,R4      ;DATA CORRECT?
3859 021044 001401      BEQ      4$      ;BR IF YES
3860 021046 104005      HLT      5      ;DUAL ADDRESSING ERROR
3861 021050 104401 4$:      SCOP1    ;SW09=1?
3862 021052 005200      INC      R0      ;INCREMENT PATTERN
3863 021054 005202      INC      R2      ;NEXT ADDRESS
3864 021056 062703 000020      ADD      #20,R3      ;ADD 1 TO ADDRESS IN R3(SHIFTED 4 TIMES)
3865 021062 022702 000010      CMP      #7+1,R2 ;LAST ADDRESS DONE?
3866 021066 001353      BNE      3$      ;BR IF NO
3867 021070 104400      SCOPE      ;SCOPE THIS TEST
3868
3869
3870 ***** TEST 51 *****
3871 ;*MICRO PROCESSOR BR REGISTER TEST
3872 ;*FLOAT A 1 THROUGH THE BR
3873 ;*FLOAT A 0 THROUGH THE BR

```

```

3874                                     ;:*****
3875                                     ;
3876                                     ; TEST 51
3877                                     ;-----
3878 021072 012737 000051 001226 TST51: MOV #51,TSTNO
3879 021100 012737 021242 001216 MOV #TST52,NEXT
3880 021106 012737 021122 001220 MOV #64$,LOCK
3881                                     ;R1 CONTAINS BASE DMC11 ADDRESS
3882 021114 104412 MSTCLR ;MASTER CLEAR DMC11
3883 021116 012700 000001 MOV #1,R0 ;START PATTERN WITH BIT0
3884                                     64$:
3885 021122 010061 000004 MOV R0,4(R1) ;WRITE PATTERN IN PORT4
3886 021126 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3887 021130 120500 120500 ;MOVE DATA TO THE BR REGISTER
3888 021132 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3889 021134 061225 061225 ;MOVE BR TO PORT 5
3890 021136 010005 MOV R0,R5 ;PUT 'EXPECTED' IN R5
3891 021140 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' IN R4
3892 021144 120504 CMPB R5,R4 ;DATA CORRECT?
3893 021146 001401 BEQ 65$ ;BR IF YES
3894 021150 104006 HLT 6 ;DATA ERROR
3895 021152 104401 65$: SCOP1
3896 021154 000241 CLC ;CLEAR CARRY
3897 021156 106100 ROLB R0 ;SHIFT BIT IN R0
3898 021160 001360 BNE 64$ ;DONE IF R0=0
3899 021162 012737 021176 001220 MOV #67$,LOCK ;NEW SCOP1
3900 021170 012700 000001 MOV #1,R0 ;START PATTERN WITH BIT0
3901 021174 005100 69$: COM R0 ;CHANGE TO FLOATING ZERO
3902 021176 005100 67$:
3903 021176 010061 000004 MOV R0,4(R1) ;WRITE PATTERN IN PORT4
3904 021202 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3905 021204 120500 120500 ;MOVE DATA TO THE BR REGISTER
3906 021206 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3907 021210 061225 061225 ;MOVE BR TO PORT 5
3908 021212 010005 MOV R0,R5 ;PUT 'EXPECTED' IN R5
3909 021214 116104 000005 MOVB 5(R1),R4 ;PUT 'FOUND' IN R4
3910 021220 120504 CMPB R5,R4 ;DATA CORRECT?
3911 021222 001401 BEQ 68$ ;BR IF YES
3912 021224 104006 HLT 6 ;DATA ERROR
3913 021226 104401 68$: SCOP1
3914 021230 005100 COM R0 ;CHANGE BACK TO A ONE
3915 021232 000241 CLC ;CLEAR CARRY
3916 021234 106100 ROLB R0 ;SHIFT BIT IN R0
3917 021236 001356 BNE 69$ ;DONE IF R0=0
3918 021240 104400 SCOPE ;SCOPE THIS TEST
3919
3920
3921                                     ;***** TEST 52 *****
3922                                     ;*SCRATCH PAD TEST
3923                                     ;*FLOAT A 1 THROUGH EACH SCRATCH PAD LOCATION
3924                                     ;*FLOAT A 0 THROUGH EACH SCRATCH PAD LOCATION
3925                                     ;:*****
3926
3927                                     ; TEST 52
3928                                     ;-----
3929 021242 012737 000052 001226 TST52: MOV #52,TSTNO
  
```


3930	021250	012737	021510	001216	MOV	#TST53,NEXT	
3931	021256	012737	021274	001220	MOV	#64\$,LOCK	
3932							:R1 CONTAINS BASE DMC11 ADDRESS
3933	021264	104412			MSTCLR		:MASTER CLEAR DMC11
3934	021266	005002			CLR	R2	:START AT ADDRESS ZERO
3935	021270	012700	000001		MOV	#1,R0	:START WITH BIT0
3936	021274	042737	000017	021314	BIC	#17,65\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
3937	021302	050237	021314		BIS	R2,65\$	:ADD ADDRESS TO INSTRUCTION
3938	021306	010061	000004		MOV	R0,4(R1)	:WRITE PATTERN TO PORT4
3939	021312	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3940	021314	123100			123100		:WRITE SCRATCH PAD(ADDRESS IN R2)
3941	021316	042737	000017	021332	BIC	#17,66\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
3942	021324	050237	021332		BIS	R2,66\$	:ADD ADDRESS TO INSTRUCTION
3943	021330	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3944	021332	040600			040600		:MOV SP TO BR
3945	021334	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
3946	021336	061225			061225		:MOVE BR TO PORT5
3947	021340	010005			MOV	R0,R5	:PUT 'EXPECTED' IN R5
3948	021342	116104	000005		MOVB	5(R1),R4	:PUT 'FOUND' IN R4
3949	021346	120504			CMPB	R5,R4	:DATA CORRECT
3950	021350	001401			BEQ	67\$	:BR IF YES
3951	021352	104007			HLT	7	:DATA ERROR
3952	021354	104401			SCOP1		:SW09 1?
3953	021356	000241			CLC		:CLEAR CARRY
3954	021360	106100			ROLB	R0	:SHIFT BIT IN R0
3955	021362	001344			BNE	64\$	:DONE IF R0=0
3956	021364	012737	021400	001220	MOV	#69\$,LOCK	:NEW SCOP1
3957	021372	012700	000001		MOV	#1,R0	:START WITH BIT0
3958	021376	005100			COM	R0	:CHANGE TO FLOATING ZERO
3959	021400	042737	000017	021420	BIC	#17,70\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
3960	021406	050237	021420		BIS	R2,70\$	:ADD ADDRESS TO INSTRUCTION
3961	021412	010061	000004		MOV	R0,4(R1)	:WRITE PATTERN TO PORT4
3962	021416	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
3963	021420	123100			123100		:WRITE SCRATCH PAD(ADDRESS IN R2)
3964	021422	042737	000017	021436	BIC	#17,71\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
3965	021430	050237	021436		BIS	R2,71\$	:ADD ADDRESS TO INSTRUCTION
3966	021434	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3967	021436	040600			040600		:MOV SP TO BR
3968	021440	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3969	021442	061225			061225		:MOVE BR TO PORT5
3970	021444	010005			MOV	R0,R5	:PUT 'EXPECTED' IN R5
3971	021446	116104	000005		MOVB	5(R1),R4	:PUT 'FOUND' IN R4
3972	021452	120504			CMPB	R5,R4	:DATA CORRECT
3973	021454	001401			BEQ	72\$	:BR IF YES
3974	021456	104007			HLT	7	:DATA ERROR
3975	021460	104401			SCOP1		:SW09=1?
3976	021462	005100			COM	R0	:CHANGE BACK TO A ONE
3977	021464	000241			CLC		:CLEAR CARRY
3978	021466	106100			ROLB	R0	:SHIFT BIT IN R0
3979	021470	001342			BNE	73\$	:DONE IF R0=0
3980	021472	012700	000001		MOV	#1,R0	:RESTART AT BIT 0
3981	021476	005202			INC	R2	:NEXT SP ADDRESS
3982	021500	022702	000020		CMP	#20,R2	:LAST ADDRESS?
3983	021504	001273			BNE	64\$	:BR IF NO
3984	021506	104400			SCOPE		:SCOPE THIS TEST
3985							

```

3986
3987
3988
3989
3990
3991
3992
3993
3994
3995 021510 012737 000053 001226 TST53:
3996 021516 012737 021732 001216
3997 021524 012737 021542 001220
3998
3999 021532 104412
4000 021534 012700 000001
4001 021540 005003
4002 021542 010302
4003 021544 042737 000017 021564
4004 021552 050237 021564
4005 021556 010061 000004
4006 021562 104414
4007 021564 123100
4008 021566 042737 000017 021602
4009 021574 050237 021602
4010 021600 104414
4011 021602 060600
4012 021604 104414
4013 021606 061225
4014 021610 010005
4015 021612 116104 000005
4016 021616 120504
4017 021620 001401
4018 021622 104007
4019 021624 104401
4020 021626 005200
4021 021630 005203
4022 021632 022703 000020
4023 021636 001341
4024 021640 012737 021654 001220
4025 021646 012700 000001
4026 021652 005003
4027 021654 010302
4028 021656 042737 000017 021672
4029 021664 050237 021672
4030 021670 104414
4031 021672 060600
4032 021674 104414
4033 021676 061225
4034 021700 010005
4035 021702 116104 000005
4036 021706 120504
4037 021710 001401
4038 021712 104007
4039 021714 104401
4040 021716 005200
4041 021720 005203

;***** 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
-----
MOV #53,TSTNO
MOV #TST54,NEXT
MOV #1$,LOCK

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;START WITH A 1
;ADDRESS 0
;MOVE ADDRESS TO R2
;CLEAR ADDRESS FIELD
;ADD ADDRESS TO INSTRUCTION
;WRITE PATTERN TO PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;WRITE SP(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
;MOV BR TO PORT5
;PUT 'EXPECTED' IN R5
;PUT 'FOUND' IN R4
;DATA CORRECT?
;BR IF YES
;DATA ERROR
;SW09-0
;INCREMENT PATTERN
;NEXT ADDRESS
;LAST ADDRESS DONE?
;BR IF NO
;NEW SCOP1
;RESTART PATTERN AT 1
;RESTART AT ADDRESS ZERO
;PUT 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
;MOV BR TO PORT5
;PUT 'EXPECTED' IN R5
;PUT 'FOUND' IN R4
;DATA CORRECT?
;BR IF YES
;SP ADDRESSING ERROR
;SW09 1?
;INCREMENT PATTERN
;NEXT ADDRESS

1$: MOV R3,R2
   BIC #17,2$
   BIS R2,2$
   MOV R0,4(R1)
   ROMCLK 123100
   BIC #17,3$
   BIS R2,3$
   ROMCLK 60600
   ROMCLK 61225
   MOV R0,R5
   MOVB 5(R1),R4
   CMPB R5,R4
   BEQ 4$
   HLT 7
   SCOP1
   INC R0
   INC R3
   CMP #20,R3
   BNE 1$
   MOV #5$,LOCK
   MOV #1,R0
   CLR R3
   MOV R3,R2
   BIC #17,6$
   BIS R2,6$
   ROMCLK 60600
   ROMCLK 61225
   MOV R0,R5
   MOVB 5(R1),R4
   CMPB R5,R4
   BEQ 7$
   HLT 7
   SCOP1
   INC R0
   INC R3

5$: MOV R3,R2
   BIC #17,6$
   BIS R2,6$
   ROMCLK 60600
   ROMCLK 61225
   MOV R0,R5
   MOVB 5(R1),R4
   CMPB R5,R4
   BEQ 7$
   HLT 7
   SCOP1
   INC R0
   INC R3

6$: MOV R3,R2
   BIC #17,6$
   BIS R2,6$
   ROMCLK 60600
   ROMCLK 61225
   MOV R0,R5
   MOVB 5(R1),R4
   CMPB R5,R4
   BEQ 7$
   HLT 7
   SCOP1
   INC R0
   INC R3

7$: MOV R3,R2
   BIC #17,6$
   BIS R2,6$
   ROMCLK 60600
   ROMCLK 61225
   MOV R0,R5
   MOVB 5(R1),R4
   CMPB R5,R4
   BEQ 7$
   HLT 7
   SCOP1
   INC R0
   INC R3

```

CZDMC MACY11 30A(1052) 24-AUG-78 12:19 PAGE 78
 CZDMCC.P11 11-AUG-78 08:21 DMC11 SCRATCH PAD TESTS

SEQ 0078

```

4042 021722 022703 000020      CMP    #20,R3 ;LAST ADDRESS DONE?
4043 021726 001352      BNE     5$      ;BR IF NO
4044 021730 104400      SCOPE      ;SCOPE THIS TEST
4045
4046
4047      ;***** TEST 54 *****
4048      ;*INTERRUPT TEST
4049      ;*TEST THAT DEVICE CAN INTERRUPT TO VECTOR A
4050      ;*****
4051
4052      ; TEST 54
4053      ;-----
4054 021732 012737 000054 001226 TST54: MOV    #54,TSTNO
4055 021740 012737 022026 001216      MOV    #TST55,NEXT
4056
4057 021746 000005      RESET      ;R1 CONTAINS BASE DMC11 ADDRESS
4058 021750 005011      CLR     (R1) ;BUS RESET
4059 021752 004537 034612      JSR     R5,SETVEC ;CLEAR RUN
4060 021756 022020      3$      ;SET UP VECTORS
4061 021760 022016      2$      ;XX0
4062 021762      340      340      ;XX4
4063 021764 012737 000340 177776 1$: .BYTE 340,340 ;LEVEL 7
4064 021772 012761 000200 000004      MOV    #340,PS ;PS = LEVEL 7
4065 022000 104414      MOV    #200,4(R1) ;WRITE PORT4
4066 022002 121111      ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4067 022004 005037 177776      121111 ;SET BR RQ IN IBUS* REG 11
4068 022010 000240      CLR     PS ;ALLOW INTERRUPT
4069 022012 104010      NOP
4070 022014 000403      HLT     10 ;NO INTERRUPT
4071 022016 104011      BR      4$
4072 022020 012706 001200      2$: HLT     11 ;WRONG VECTOR
4073 022024 104400      3$: MOV     #STACK,SP ;RESET STACK
4074      4$: SCOPE      ;SCOPE THIS TEST
4075
4076      ;***** TEST 55 *****
4077      ;*INTERRUPT TEST
4078      ;*TEST THAT DEVICE CAN INTERRUPT TO VECTOR B
4079      ;*****
4080
4081      ; TEST 55
4082      ;-----
4083 022026 012737 000055 001226 TST55: MOV    #55,TSTNO
4084 022034 012737 022120 001216      MOV    #TST56,NEXT
4085
4086 022042 104412      MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
4087 022044 004537 034612      JSR     R5,SETVEC ;MASTER CLEAR DMC11
4088 022050 022110      2$      ;SET UP VECTORS
4089 022052 022112      3$      ;XX0
4090 022054      340      340      ;XX4
4091 022056 012737 000340 177776 1$: .BYTE 340,340 ;LEVEL 7
4092 022064 012761 000300 000004      MOV    #340,PS ;PS = LEVEL 7
4093 022072 104414      MOV    #300,4(R1) ;WRITE PORT4
4094 022074 121111      ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4095 022076 005037 177776      121111 ;SET BR RQ IN IBUS* REG 11
4096 022102 000240      CLR     PS ;ALLOW INTERRUPT
4097 022104 104010      HLT     10 ;NO INTERRUPT

```

```

4098 022106 000403      BR      4$
4099 022110 104011      2$:    HLT      11      ;WRONG VECTOR
4100 022112 012706 001200 3$:    MOV      #STACK,SP      ;RESET STACK
4101 022116 104400      4$:    SCOPE      ;SCOPE THIS TEST
4102
4103
4104      ;***** TEST 56 *****
4105      ;*PRIORITY INTERRUPT TESTS
4106      ;*SET PS TO ALL BR LEVELS EQUAL OR GREATER THAN
4107      ;*THE DMC11 LEVEL,VERIFY THAT DMC11 DOES NOT INTERRUPT
4108      ;*****
4109
4110      ; TEST 56
4111      ;-----
4112 022120 012737 000056 001226 TST56: MOV      #56,TSTNO
4113 022126 012737 022240 001216      MOV      #TST57,NEXT
4114
4115 022134 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
4116 022136 012702 000340      MOV      #340,R2      ;MASTER CLEAR DMC11
4117 022142 010237 177776      MOV      R2,PS      ;PUT LEVEL 7 IN R2
4118 022146 013700 001366      MOV      STAT1,R0      ;SET PRIORITY TO 7
4119 022152 006200      ASR      R0      ;GET BR LEVEL OF DMC11
4120 022154 006200      ASR      R0      ;SHIFT R0 4 TIMES
4121 022156 006200      ASR      R0      ;TO GET PROPER LEVEL
4122 022160 006200      ASR      R0
4123 022162 042700 177437      BIC      #177437,R0      ;CLEAR UNWANTED BITS
4124 022166 004537 034612      JSR      R5,SETVEC      ;SET UP VECTORS
4125 022172 022234      2$:      ;A VECTOR
4126 022174 022234      2$:      ;B VECTOR
4127 022176      340      340      .BYTE      340,340      ;PRIORITY 7
4128 022200 012761 000200 000004 4$: MOV      #200,4(R1)      ;LOAD PORT4
4129 022206 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
4130 022210 121111      121111      ;SET BR REQUEST
4131 022212 010237 177776      5$: MOV      R2,PS      ;PUT LEVEL IN R2 IN PS
4132 022216 000240      NOP
4133 022220 020002      CMP      R0,R2      ;IS PRESENT PS LEVEL - TO DMC LEVEL
4134 022222 001403      BEQ      1$      ;BR IF YES
4135 022224 162702 000040      SUB      #40,R2      ;NO GET NEXT LOWER LEVEL IN R2
4136 022230 000770      BR      5$      ;AND CONTINUE WITH TEST
4137 022232 104400      1$:    SCOPE      ;SCOPE THIS TEST
4138 022234 104020      2$:    HLT      20      ;ERROR UNEXPECTED INTERRUPT
4139 022236 000002      RTI
4140
4141      ;***** TEST 57 *****
4142      ;*PRIORITY INTERRUPT TESTS
4143      ;*SET PS TO ALL BR LEVELS LESS THAN THE DMC11 LEVEL
4144      ;*VERIFY THAT THE DMC11 WILL INTERRUPT
4145      ;*****
4146
4147      ; TEST 57
4148      ;-----
4149
4150 022240 012737 000057 001226 TST57: MOV      #57,TSTNO
4151 022246 012737 022404 001216      MOV      #TST60,NEXT
4152
4153 022254 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
      ;MASTER CLEAR DMC11

```

4154	022256	012702	000340		MOV	#340,R2	:PUT LEVEL 7 IN R2
4155	022262	010237	177776		MOV	R2,PS	:SET PRIORITY TO 7
4156	022266	013700	001366		MOV	STAT1,R0	:GET BR LEVEL OF DMC11
4157	022272	006200			ASR	R0	:SHIFT R0 4 TIMES
4158	022274	006200			ASR	R0	:TO GET PROPER LEVEL
4159	022276	006200			ASR	R0	
4160	022300	006200			ASR	R0	
4161	022302	042700	177437		BIC	#177437,R0	:CLEAR UNWANTED BITS
4162	022306	010002			MOV	R0,R2	:PUT DMC LEVEL IN R2
4163	022310	162702	000040		SUB	#40,R2	:GET NEXT LOWER LEVEL IN R2
4164	022314	004537	034612		JSR	R5,SETVEC	:SET UP VECTORS
4165	022320	022366			2\$		:A VECTOR
4166	022322	022374			3\$		:B VECTOR
4167	022324	340	340		.BYTE	340,340	:PRIORITY 7
4168	022326	012761	000200	000004	4\$: MOV	#200,4(R1)	:LOAD PORT4
4169	022334	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4170	022336	121111			121111		:SET BR REQUEST
4171	022340	010237	177776		5\$: MOV	R2,PS	:PUT LEVEL IN R2 IN PS
4172	022344	000240			NOP		
4173	022346	104010			HLT	10	:ERROR, NO INTERRUPT
4174	022350	022702	000140		6\$: CMP	#140,R2	:IS IT DOWN TO LEVEL 3 YET?
4175	022354	001403			BEO	1\$	:YES,DMC DID NOT INTERRUPT, ERROR
4176	022356	162702	000040		SUB	#40,R2	:PUT NEXT LOWER LEVEL IN R2
4177	022362	000761			BR	4\$	:CONTINUE TEST
4178	022364	104400			1\$: SCOPE		:SCOPE THIS TEST
4179	022366	012716	022350		2\$: MOV	#6\$, (SP,	:SET UP FOR RTI
4180	022372	000002			RTI		
4181	022374	104011			3\$: HLT	11	:ERROR, WRONG VECTOR
4182	022376	012716	022350		MOV	#6\$, (SP)	:SET UP FOR RTI
4183	022402	000002			RTI		
4184							
4185							
4186							
4187							
4188							
4189							
4190							
4191							
4192							
4193	022404	012737	000060	001226	TST60: MOV	#60,TSTNO	
4194	022412	012737	022510	001216	MOV	#TST61,NEXT	
4195							
4196	022420	000005			RESET		:R1 CONTAINS BASE DMC11 ADDRESS
4197	022422	005011			CLR	(R1)	:BUS RESET
4198	022424	005061	000004		CLR	4(R1)	:CLEAR RUN
4199	022430	004537	034634		JSR	R5,NPRSET	:CLR PORT4
4200	022434	000000			0		:SET UP IBUS REG 0-7
4201	022436	177777			-1		:IN DATA
4202	022440	022506			3\$		:OUT DATA
4203	022442	022504			2\$		:IN BA
4204	022444	005037	022504		CLR	2\$	:OUT BA
4205	022450	012761	000021	000004	MOV	#21,4(R1)	:CLEAR 2\$
4206	022456	104414			ROMCLK		:WRITE PORT4
4207	022460	121110			121110		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4208	022462	000240			NOP		:SET NPR BITS IN IBUS* REG 11
4209	022464	012705	177777		MOV	#-1,R5	:PUT 'EXPECTED' IN R5

***** TEST 60 *****
 :*NPR TEST
 :*TEST OF DATO, 1 WORD FROM UPROC TO 11 MEMORY
 :*****

: TEST 60

```

4210 022470 013704 022504      MOV      2$,R4      ;PUT 'FOUND' IN R4
4211 022474 020504      CMP      R5,R4      ;DATA CORRECT?
4212 022476 001401      BEQ      4$      ;BR IF YES
4213 022500 104012      HLT      12      ;ERROR NPR FAILED
4214 022502 104400      4$:      SCOPE      ;SCOPE THIS TEST
4215 022504 000000      2$:      0      ;OUT BA
4216 022506 000000      3$:      0      ;IN BA
4217
4218
4219
4220      ;***** TEST 61 *****
4221      ;*NPR TEST
4222      ;*TEST OF DAT1, 1 WORD FROM 11 MEMORY TO UPROC
4223      ;*****
4224
4225      ; TEST 61
4226      ;-----
4226 022510 012737 000061 001226 1ST61: MOV      #61,TSTNO
4227 022516 012737 022624 001216      MOV      #TST62,NEXT
4228
4229 022524 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
4230 022526 005061 000004      CLR      4(R1)      ;MASTER CLEAR DMC11
4231 022532 004537 034634      JSR      R5,NPRSET      ;CLR PORT4
4232 022536 000000      0      ;SET UP IBUS REG 0-7
4233 022540 177777      -1      ;IN DATA
4234 022542 022622      3$      ;OUT DATA
4235 022544 022620      2$      ;IN BA
4236 022546 012737 177777 022622      MOV      #-1,3$      ;OUT BA
4237 022554 012761 000001 000004      MOV      #1,4(R1)      ;PUT DATA IN 3$
4238 022562 104414      ROMCLK      ;WRITE PORT4
4239 022564 121110      121110      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4240 022566 000240      NOP      ;SET NPR BITS IN IBUS* REG 11
4241 022570 012705 177777      MOV      #-1,R5      ;PUT 'EXPECTED' IN R5
4242 022574 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4243 022576 021004      021004      ;MOVE IN DATA LOW BYTE TO PORT4
4244 022600 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4245 022602 021025      021025      ;MOVE IN DATA HIGH BYTE TO PORT5
4246 022604 016104 000004      MOV      4(R1),R4      ;PUT 'FOUND' IN R4
4247 022610 020504      CMP      R5,R4      ;DATA CORRECT?
4248 022612 001401      BEQ      4$      ;BR IF YES
4249 022614 104012      HLT      12      ;ERROR NPR FAILED
4250 022616 104400      4$:      SCOPE      ;SCOPE THIS TEST
4251 022620 000000      2$:      0      ;OUT BA
4252 022622 000000      3$:      0      ;IN BA
4253
4254
4255      ;***** TEST 62 *****
4256      ;*NPR TEST
4257      ;*TEST OF DAT0B, 1 BYTE FROM UPROC TO 11 MEMORY
4258      ;*****
4259
4260      ; TEST 62
4261      ;-----
4262 022624 012737 000062 001226 1ST62: MOV      #62,TSTNO
4263 022632 012737 022726 001216      MOV      #TST63,NEXT
4264
4265 022640 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
      ;MASTER CLEAR DMC11

```


PC	Instruction	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	Op420	Op421	Op422	Op423	Op424	Op425	Op426	Op427	Op428	Op429	Op430	Op431	Op432	Op433	Op434	Op435	Op436	Op437	Op438	Op439	Op440	Op441	Op442	Op443	Op444	Op445	Op446	Op447	Op448	Op449	Op450	Op451	Op452	Op453	Op454	Op455	Op456	Op457	Op458	Op459	Op460	Op461	Op462	Op463	Op464	Op46
----	-------------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	------

```

4322 023060 104012          HLT      12          ;ERROR BA 16 AND 17 FAILED
4323 023062 104400          3$:      SCOPE          ;SCOPE THIS TEST
4324
4325
4326          ;***** TEST 64 *****
4327          ;*TEST OF EA BITS 16 AND 17
4328          ;*DO A DATI USING IN BA BITS 16 AND 17
4329          ;*VERIFY CORRECT RESULTS
4330          ;*****
4331
4332          ; TEST 64
4333          ;-----
4334 023064 012737 000064 001226 TST64: MOV      #64,TSTNO
4335 023072 012737 023210 001216      MOV      #TST65,NEXT
4336
4337 023100 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
4338 023102 013737 001412 023130      MOV      DMP04,1$      ;MASTER CLEAR DMC11
4339 023110 013737 001412 023126      MOV      DMP04,2$      ;USE SEL4 FOR ADDRESS
4340 023116 004537 034634          JSR      R5,NPRSET      ;USE SEL4 FOR ADDRESS
4341 023122 000000          0          ;LOAD BA AND DATA
4342 023124 125252          125252      0          ;IN DATA
4343 023126 000000          2$:      0          ;OUT DATA
4344 023130 000000          1$:      0          ;IN BA
4345 023132 012761 000015 000004      MOV      #15,4(R1)      ;OUT BA
4346 023140 012761 121110 000006      MOV      #121110,6(R1)      ;LOAD SEL4
4347 023146 012711 003000          MOV      #BIT9!BIT10,(R1) ;PUT INSTRUCTION IN SEL6
4348 023152 052711 000400          BIS      #BIT8,(R1)      ;SET CROMI AND CROMO..
4349 023156 000240          NOP          ;CLOCK IT!
4350 023160 012705 121110          MOV      #121110,R5      ;WAIT FOR NPR
4351 023164 104414          ROMCLK      ;PUT 'EXPECTED' IN R5
4352 023166 021004          021004      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4353 023170 104414          ROMCLK      ;MOVE IN DATA LB TO SEL4
4354 023172 021025          021025      ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4355 023174 016104 000004          MOV      4(R1),R4      ;MOVE IN DATA HB TO SEL5
4356 023200 020504          CMP      R5,R4      ;PUT 'FOUND' IN R4
4357 023202 001401          BEQ      3$      ;CORRECT RESULTS ?
4358 023204 104012          HLT      12      ;BR IF YES
4359 023206 104400          3$:      SCOPE      ;ERROR BA 16 AND 17 FAILED
4360          ;SCOPE THIS TEST
4361
4362          ;***** TEST 65 *****
4363          ;*NPR NON-EXISTENT MEMORY TEST
4364          ;*DO A DATO TO A NON-EXISTENT ADDRESS
4365          ;*VERIFY THAT THE NON-EXISTENT BIT SET IN IBUS REG 11
4366          ;*****
4367
4368          ; TEST 65
4369          ;-----
4370 023210 012737 000065 001226 TST65: MOV      #65,TSTNO
4371 023216 012737 023320 001216      MOV      #TST66,NEXT
4372
4373 023224 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
4374 023226 004537 034634          JSR      R5,NPRSET      ;MASTER CLEAR DMC11
4375 023232 000000          0          ;LOAD IBUS REGISTERS 0-7
4376 023234 000000          0          ;IN DATA
4377 023236 177320          177320      0          ;OUT DATA
          ;IN BA

```


4378	023240	177320			177320	:OUT BA
4379	023242	012761	000014	000004	MOV #14,4(R1)	:SET OUT BA BITS 16+17 IN PORT4
4380	023250	104414			ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4381	023252	121111			121111	:SET OUTBA 16 AND 17
4382	023254	012761	000021	000004	MOV #21,4(R1)	:SET NPR REQUEST BITS IN PORT4
4383	023262	104414			ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4384	023264	121110			121110	:MOV IBUS* 4 TO IBUS* 10
4385	023266	000240			NOP	
4386	023270	104414			ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4387	023272	121225			121225	:MOV IBUS*11 TO IBUS*5
4388	023274	012705	000001		MOV #1,R5	:PUT 'EXPECTED' IN R5
4389	023300	116104	000005		MOVB 5(R1),R4	:PUT 'FOUND' IN R4
4390	023304	042704	177776		BIC #177776,R4	:CLEAR UNWANTED BITS
4391	023310	020504			CMP R5,R4	:DATA CORRECT?
4392	023312	001401			BEQ 1\$	:BR IF YES
4393	023314	104012			HLT 12	:ERROR NON-EXISTENT MEM BIT FAILED TO SET
4394	023316	104400			SCOPE	:SCOPE THIS TEST

\$:

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

: TEST 66

4405	023320	012737	000066	001226	TST66:	MOV #66,TSTNO	
4406	023326	012737	023426	001216		MOV #TST67,NEXT	
4407							
4408	023334	104412			MSTCLR	:R1 CONTAINS BASE DMC11 ADDRESS	
4409	023336	004537	034634		JSP R5,NPRSET	:MASTER CLEAR DMC11	
4410	023342	000000			0	:LOAD IBUS REGISTERS 0-7	
4411	023344	000000			0	:IN DATA	
4412	023346	177320			177320	:OUT DATA	
4413	023350	177320			177320	:IN BA	
4414	023352	005061	000004		CLR 4(R1)	:OUT BA	
4415	023356	104414			ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304	
4416	023360	121111			121111	:CLEAR NON-EXISTENT BIT	
4417	023362	012761	000015	000004	MOV #15,4(R1)	:SET NPR REQUEST BITS IN PORT4	
4418	023370	104414			ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
4419	023372	121110			121110	:MOV IBUS* 4 TO IBUS* 10	
4420	023374	000240			NOP		
4421	023376	104414			ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304	
4422	023400	121225			121225	:MOV IBUS*11 TO IBUS*5	
4423	023402	012705	000001		MOV #1,R5	:PUT 'EXPECTED' IN R5	
4424	023406	116104	000005		MOVB 5(R1),R4	:PUT 'FOUND' IN R4	
4425	023412	042704	177776		BIC #177776,R4	:CLEAR UNWANTED BITS	
4426	023416	020504			CMP R5,R4	:DATA CORRECT?	
4427	023420	001401			BEQ 1\$	:BR IF YES	
4428	023422	104012			HLT 12	:ERROR NON-EXISTENT MEM BIT FAILED TO SET	
4429	023424	104400			SCOPE	:SCOPE THIS TEST	

\$:

***** TEST 67 *****
 :*NPR TEST*

4430

4431

4432

4433

```

4434      ;*USING DATC, NPR A BINARY COUNT (0-377 )
4435      ;*FROM MICRO-PROCESSOR TO ALL AVAILABLE MEMORY
4436      ;*****
4437
4438      : TEST 67
4439      :-----
4440 023426 012737 000067 001226 TS'67: MOV #67,TSTNO
4441 023434 012737 000003 001222 MOV #3,ICOUNT
4442 023442 012737 023624 001216 MOV #TST70,NEXT
4443
4444 023450 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
4445 023452 005037 023622 CLR 5$ ;MASTER CLEAR DMC11
4446 023456 005000 CLR R0 ;START FLAG AT 0
4447 023460 012702 036534 MOV #CORMAX,R2 ;DATA
4448 023464 ;$: ;ADDRESS
4449 023464 010037 023514 MOV R0,2$ ;LOAD DATA
4450 023470 010237 023520 MOV R2,4$ ;LOAD BA
4451 023474 032702 000001 BIT #BIT0,R2 ;IS BA ODD?
4452 023500 001402 BEQ +4 ;BR IF NO
4453 023502 000337 023514 SWAB 2$ ;IF ODD PUT DATA IN HI-BYTE
4454 023506 004537 034634 JSR R5,NPRSET ;LOAD NPR REGISTERS
4455 023512 000000 0 ;IN DATA
4456 023514 000000 2$: 0 ;OUT DATA
4457 023516 000000 0 ;IN BA
4458 023520 000000 4$: 0 ;OUT BA
4459 023522 105012 CLRB (R2) ;CLEAR MEMORY LOCATION
4460 023524 012761 000221 300004 MOV #221,4(R1) ;LOAD PORT4
4461 023532 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
4462 023534 121110 121110 ;DO THE NPR
4463 023536 000240 NOP
4464 023540 010005 MOV R0,R5 ;PUT 'EXPECTED' IN R5
4465 023542 111204 MOVB (R2),R4 ;PUT 'FOUND' IN R4
4466 023544 120504 MPB R5,R4 ;IS DATA CORRECT?
4467 023546 001401 BEQ 3$ ;BR IF YES
4468 023550 104021 HLT 21 ;ERROR, DATA INCORRECT
4469 023552 104401 7$: SCOP;
4470 023554 005200 INC R0 ;NEXT CHARACTER
4471 023556 042700 177400 BIC #177400,R0 ;USE ONLY LOW BYTE
4472 023562 005737 023622 TST 5$ ;HAS MAX MEMORY BEEN REACHED YET?
4473 023566 001402 BEQ 6$ ;BR IF NO
4474 023570 005700 TST R0 ;DONE PATTERN?
4475 023572 001412 BEQ 7$ ;BR IF YES
4476 023574 005202 6$: INC R2 ;INC BA
4477 023576 023702 001304 CMP MEMLIM,R2 ;REACHED MEMORY LIMIT YET?
4478 023602 001330 BNE 1$ ;BR IF NOT
4479 023604 012702 036534 MCV #CORMAX,R2 ;RESTART BA AT FIRST ADDRESS
4480 023610 012737 177777 023622 MOV #-1,5$ ;SET FLAG TO END TEST AT END OF DATA PATTERN
4481 023616 000722 BR 1$ ;CONTINUE
4482 023620 104400 7$: SCOP; ;SCOPE THIS TEST
4483 023622 000000 5$: 0 ;THIS LOCATION IS A FLAG, IT STARTS AT 0,
4484 ;AND IS SET TO -1 WHEN LAST MEMORY ADDRESS
4485 ;IS USED, TEST IS THEN ENDED WHEN PATTERN IS FINISHED
4486
4487
4488      ;***** TEST 70 *****
4489      ;*MAIN MEMORY TEST

```

```

4490                                     ;*FLOAT A 1 THROUGH ALL MAIN MEMORY LOCATIONS
4491                                     ;*****
4492
4493                                     ; TEST 70
4494                                     ;-----
4495 023624 012737 000070 001226 TST70: MOV #70,TSTNO
4496 023632 012737 023752 001216 MOV #TST71,NEXT
4497 023640 012737 023656 001220 MOV #65$,LOCK
4498
4499 023646 104412 MSTCLR                                     ;R1 CONTAINS BASE DMC11 ADDRESS
4500 023650 005002 CLR R2                                     ;MASTER CLEAR DMC11
4501 023652 012700 000001 1$: MOV #1,R0                                     ;START WITH ADDRESS 0
4502 023656 042737 000377 023672 65$: BIC #377,66$                                     ;START WITH BIT 0
4503 023664 050237 023672 BIS R2,66$                                     ;CLEAR ADDRESS FIELD OF INSTRUCTION
4504 023670 104414 ROMCLK                                     ;ADD ADDRESS TO INSTRUCTION
4505 023672 010000 66$: 010000                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
4506 023674 010061 000004 MOV R0,4(R1)                                     ;LOAD MAR WITH ADDRESS IN R2
4507 023700 104414 ROMCLK                                     ;WRITE PATTERN IN PORT4
4508 023702 122500 122500                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
4509 023704 104414 ROMCLK                                     ;MOVE PORT4 TO MEMORY
4510 023706 040620 040620                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
4511 023710 104414 ROMCLK                                     ;MOVE MEMORY TO BR
4512 023712 061225 61225                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4513 023714 010005 MOV R0,R5                                     ;MOVE BR TO PORT5
4514 023716 116104 000005 MOVB 5(R1),R4                                     ;PUT 'EXPECTED' IN R5
4515 023722 120504 CMPB R5,R4                                     ;PUT 'FOUND' IN R4
4516 023724 001401 BEQ 67$                                     ;DATA CORRECT?
4517 023726 104013 HLT 13                                     ;BR IF YES
4518 023730 104401 67$: SCOP1                                     ;DATA ERROR
4519 023732 000241 CLC                                     ;SW09=1?
4520 023734 106100 ROLB R0                                     ;CLEAR CARRY
4521 023736 001347 BNE 65$                                     ;SHIFT BIT IN R0
4522 023740 005202 INC R2                                     ;DONE IF R0=0
4523 023742 022702 000400 CMP #400,R2                                     ;NEXT ADDRESS
4524 023746 001341 BNE 1$                                     ;LAST ADDRESS
4525 023750 104400 SCOPE 1$                                     ;BR IF NO
4526                                     ;SCOPE THIS TEST
4527
4528                                     ;***** TEST 71 *****
4529                                     ;*MAIN MEMORY TEST
4530                                     ;*FLOAT A 0 THROUGH ALL MAIN MEMORY LOCATIONS
4531                                     ;*****
4532
4533                                     ; TEST 71
4534                                     ;-----
4535 023752 012737 000071 001226 TST71: MOV #71,TSTNO
4536 023760 012737 024104 001216 MOV #TST72,NEXT
4537 023766 012737 024006 001220 MOV #65$,LOCK
4538
4539 023774 104412 MSTCLR                                     ;R1 CONTAINS BASE DMC11 ADDRESS
4540 023776 005002 CLR R2                                     ;MASTER CLEAR DMC11
4541 024000 012700 000001 1$: MOV #1,R0                                     ;START WITH ADDRESS 0
4542 024004 005100 64$: COM R0                                     ;START WITH BIT 0
4543 024006 042737 000377 024022 65$: BIC #377,66$                                     ;CHANGE TO FLOATING 0
4544 024014 050237 024022 BIS R2,66$                                     ;CLEAR ADDRESS FIELD OF INSTRUCTION
4545 024020 104414 ROMCLK                                     ;ADD ADDRESS TO INSTRUCTION
4546                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304

```

4546	024022	010000		66\$:	010000		:LOAD MAR WITH ADDRESS IN R2
4547	024024	010061	000004		MOV	R0,4(R1)	:WRITE PATTERN IN PORT4
4548	024030	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4549	024032	122500			122500		:MOVE PORT4 TO MEMORY
4550	024034	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4551	024036	040620			040620		:MOVE MEMORY TO BR
4552	024040	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4553	024042	061225			61225		:MOVE BR TO PORT5
4554	024044	010005			MOV	R0,R5	:PUT 'EXPECTED' IN R5
4555	024046	116104	000005		MOVB	5(R1),R4	:PUT 'FOUND' IN R4
4556	024052	120504			CMPB	R5,R4	:DATA CORRECT?
4557	024054	001401			BEQ	67\$	:BR IF YES
4558	024056	104013			HLT	13	:DATA ERROR
4559	024060	104401		67\$:	SCOP1		:SW09=1?
4560	024062	005100			COM	R0	:CHANGE TO FLOATING 1
4561	024064	000241			CLC		:CLEAR CARRY
4562	024066	106100			ROLB	R0	:SHIFT BIT IN R0
4563	024070	001345			BNE	64\$	:DONE IF R0=0
4564	024072	005202			INC	R2	:NEXT ADDRESS
4565	024074	022702	000400		CMP	#400,R2	:LAST ADDRESS
4566	024100	001337			BNE	1\$	:BR IF NO
4567	024102	104400			SCOPE		:SCOPE THIS TEST

***** TEST 72 *****
 :*MAIN MEMORY DUAL ADDRESSING TEST
 :*LOAD EACH MEMORY LOCATION WITH IT'S OWN ADDRESS
 :*READ BACK EACH LOCATION TO VERIFY CORRECT ADDRESSING
 :*****

: TEST 72

4578	024104	012737	000072	001226	1\$*72:	MOV	#72,TSTNO	
4579	024112	012737	024304	001216		MOV	#1\$*73,NEXT	
4580	024120	012737	024132	001220		MOV	#1\$,LOCK	
4581								:R* CONTAINS BASE DMC11 ADDRESS
4582	024126	104412				MSCLR		:MASTER CLEAR DMC11
4583	024130	005002				CLR	R2	:START AT ADDRESS 0
4584	024132	042737	000377	044146	1\$:	BIC	#377,2\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
4585	024140	050237	024146			BIS	R2,2\$	:ADD ADDRESS TO INSTRUCTION
4586	024144	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4587	024146	010000			1\$:	010000		:LOAD MAR
4588	024150	010261	000004			MOV	R2,4(R1)	
4589	024154	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4590	024156	122500				122500		:MOVE PORT4 TO MEMORY
4591	024160	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4592	024162	040620				040620		:MOVE MEMORY TO THE BR
4593	024164	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4594	024166	061225				61225		:MOVE BR TO PORT5
4595	024170	010205				MOV	R2,R5	:PUT 'EXPECTED' IN R5
4596	024172	116104	000005			MOVB	5(R1),R4	:PUT 'FOUND' IN R4
4597	024176	120504				CMPB	R5,R4	:DATA CORRECT?
4598	024200	001401				BEQ	3\$	:BR IF YES
4599	024202	104013				HLT	3	:DATA ERROR
4600	024204	104401			1\$:	SCOP1		:SW09=1?
4601	024206	005202				INC	R2	:NEXT ADDRESS

```
4602 024210 022702 000400      CMP      #400,R2      ;LAST ADDRESS
4603 024214 001346      BNE      1$      ;BR IF NO
4604 024216 012737 024226 001220  MOV      #4$,LOCK    ;NEW SCOPE 1
4605 024224 005002      CLR      R2      ;RESTART AT ADDRESS 0
4606 024226 042737 000377 024242 4$:  BIC      #377,5$     ;CLEAR ADDRESS FIELD OF INSTRUCTION
4607 024234 050237 024242      BIS      R2,5$      ;ADD ADDRESS TO INSTRUCTION
4608 024240 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4609 024242 010000 5$:      010000      ;LOAD THE MAR
4610 024244 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4611 024246 040620      040620      ;MOVE MEMORY TO THE BR
4612 024250 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4613 024252 061225      61225      ;MOV BR TO PORT5
4614 024254 010205      MOV      R2,R5      ;PUT 'EXPECTED' IN R5
4615 024256 116104 000605      MOVB     5(R1),R4    ;PUT 'FOUND' IN R4
4616 024262 120504      CMPB      R5,R4      ;DATA CORRECT?
4617 024264 001401      BEQ      6$      ;BR IF YES
4618 024266 104013      HLT      3      ;ADDRESSING ERROR
4619 024270 104401 6$:      SCOP1     ;SW09=1?
4620 024272 005202      INC      R2      ;NEXT ADDRESS
4621 024274 022702 000400      CMP      #400,R2    ;IS IT THE LAST
4622 024300 001352      BNE      4$      ;BR IF NO
4623 024302 104400      SCOPE     ;SCOPE THIS TEST
4624
4625
4626
4627
4628
4629
4630
4631
4632
4633
```

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

```
: TEST 73
```

```
4634 024304 012737 000673 001226 TEST73: MOV      #73,TSTNO
4635 024312 012737 024440 001216      MOV      #TST74,NEXT
4636
4637 024320 104412      MSTCLR     ;R1 CONTAINS BASE DMC11 ADDRESS
4638 024322 005002      CLR      R2      ;MASTER CLEAR DMC11
4639 024324 104414      ROMCLK     ;START WITH A ZERO
4640 024326 010000      010000      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4641 024330 032737 100000 001366      BIT      #BIT15,STAT1 ;LOAD MAR
4642 024336 001402      BEQ      .+6      ;DMC?
4643 024340 104414      ROMCLK     ;BR IF YES
4644 024342 004000      4000      ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4645 024344 010261 000004 1$:      MOV      R2,4(R1)    ;MAR HI = 0 (KMC ONLY)
4646 024350 104414      ROMCLK     ;WRITE DATA TO PORT4
4647 024352 136500      36500      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4648 024354 005202      INC      R2      ;LOAD MEM AUTO-INC MAR
4649 024356 022702 000400      CMP      #400,R2    ;INCREMENT DATA
4650 024362 001370      BNE      1$      ;DONE YET?
4651 024364 005002      CLR      R2      ;BR IF NO
4652 024366 104414      ROMCLK     ;RESTART WITH A ZERO
4653 024370 010000      010000      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4654 024372 032737 100000 001366      BIT      #BIT15,STAT1 ;LOAD MAR
4655 024400 001402      BEQ      .+6      ;DMC?
4656 024402 104414      ROMCLK     ;BR IF YES
4657 024404 004000      4000      ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
                                ;MAR HI = 0 (KMC ONLY)
```

4658	024406				2\$:	ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4659	024406	104414				055224		:MOVE MEM TO PORT4
4660	024410	055224				MOV	R2,R5	:PUT 'EXPECTED' IN R5
4661	024412	010205				MOV	4(R1),R4	:PUT 'FOUND' IN R4
4662	024414	016104	000004			CMPB	R5,R4	:DATA CORRECT?
4663	024420	120504				BEQ	3\$	:BR IF YES
4664	024422	001401				HLT	14	:MAR ERROR
4665	024424	104014			3\$:	INC	R2	:NEXT ADDRESS
4666	024426	005202				CMP	#400,R2	:DONE YET?
4667	024430	022702	000400			BNE	2\$	:BR IF NO
4668	024434	001364				SCOPE		:SCOPE THIS TEST
4669	024436	104400						
4670								
4671								
4672								
4673								
4674								
4675								
4676								
4677								
4678								

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

: TEST 74

4679	024440	012737	000074	001226	TST74:	MOV	#74,TSTNO	
4680	024446	012737	024552	001216		MOV	#TST75,NEXT	
4681	024454	012737	024500	001220		MOV	#1\$,LOCK	
4682								
4683	024462	104412				MSTCLR		:R1 CONTAINS BASE DMC11 ADDRESS
4684	024464	004737	034676			JSR	PC,MEMLD	:MASTER CLEAR DMC11
4685	024470	024542				TDATA		:LOAD MAINMEM DATA
4686	024472	004737	034732			JSR	PC,SPLD	:POINTER TO DATA
4687	024476	024542				TDATA		:LOAD SP DATA
4688	024500				1\$:			:POINTER TO DATA
4689	024500	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4690	024502	010000				010000		:MAR 0
4691	024504	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4692	024506	054400				054400.<0*20>		:ADD 377 AND 377, TO SET C BIT
4693	024510	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4694	024512	040421				040401.<1*20>		:ADD 0 AND 0 AND THE C BIT
4695	024514	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4696	024516	061224				61224		:PUT RESULTS IN PORT4
4697	024520	012705	000001			MOV	#1,R5	:PUT 'EXPECTED' IN R5
4698	024524	016104	000004			MOV	4(R1),R4	:PUT 'FOUND' IN R4
4699	024530	120504				CMPB	R5,R4	:DATA CORRECT?
4700	024532	001401				BEQ	2\$	:BR IF YES
4701	024534	104015				HLT	15	:ERROR C BIT NOT SET
4702	024536	104401			2\$:	SCOPE		:SW09=1?
4703	024540	104400				SCOPE		:SCOPE THIS TEST
4704	024542	377	000	000	TDA*A:	.BYTE	-1,0,0,0,0,0,0,0	
4705	024545	000	000	000				
4706	024550	000	000					
4707								
4708								
4709								
4710								
4711								
4712								
4713								

.EVEN

***** TEST 75 *****
 :*ALU TEST
 :*TEST OF ALU FUNCTION SEL B WITH C BIT CLEARED
 :*ALU FUNCTION (B) (ODE=11)


```

4714      ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4715      ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4716      ;*****
4717
4718      ; TEST 75
4719      ;-----
4720 024552 012737 000075 001226 TST75: MOV    #75,TSTNO
4721 024560 012737 024726 001216      MOV    #TST76,NEXT
4722 024566 012737 024620 001220      MOV    #1$,LOCK
4723
4724 024574 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
4725 024576 005000      CLR          ;MASTER CLEAR DMC11
4726 024600 012702 024716      CLR      R0      ;MEM + SP ADDRESS
4727 024604 004737 034676      MOV    #5$,R2      ;POINTER TO CORRECT DATA
4728 024610 035022      JSR      PC,MEMLD      ;LOAD 8 WORDS OF MAIN MEMORY
4729 024612 004737 034732      MEMDAT      ;POINTER TO DATA
4730 024616 035032      JSR      PC,SPLD      ;LOAD 8 WORDS OF SP
4731 024620 004737 034776      SPDAT      ;POINTER TO DATA
4732 024624 042737 000017 024640 1$: JSR      PC,CLRC      ;CLEAR C BIT!
4733 024632 050037 024640      BIC      #1$,2$      ;CLEAR ADDRESS FIELD OF INSTRUCTION
4734 024636 104414      BIS      R0,2$      ;ADD ADDRESS TO INSTRUCTION
4735 024640 010000      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
4736 024642 042737 000017 024656 2$: 010000      ;LOAD MAR
4737 024650 050037 024656      BIC      #17,3$      ;CLEAR ADDRESS OF INSTRUCTION
4738 024654 104414      BIS      R0,3$      ;ADD ADDRESS TO INSTRUCTION
4739 024656 040620      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4740 024660 104414      040400!<11*20>      ;BR    SEL B
4741 024662 061224      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4742 024664 111205      61224      ;MOVE BR TO PORT4
4743 024666 116104 000004      MOV      (R2),R5      ;PUT 'EXPECTED' IN R5
4744 024672 120504      MOV      4(R1),R4      ;PUT 'FOUND' IN R4
4745 024674 001401      CMP      R5,R4      ;DATA CORRECT?
4746 024676 104015      BEQ      4$      ;BR IF YES
4747 024700 104401      HLT      15      ;ALU ERROR
4748 024702 005202      SCOPE1      ;SW09=1?
4749 024704 005200      INC      R2      ;NEXT DATA
4750 024706 022700 000010      INC      R0      ;NEXT ADDRESS
4751 024712 001342      CMP      #10,R0      ;DONE YET?
4752 024714 104400      BNE      1$      ;BR IF NO
4753 024716      000      377      000 5$: SCOPE      ;SCOPE THIS TEST
4754 024721      377      125      252      .BYTE 0,-1,0,-1,125,252,125,252
4755 024724      125      252
4756      ;.EVEN
4757
4758
4759      ;***** TEST 76 *****
4760      ;*ALU TEST
4761      ;*TEST OF ALU FUNCTION SEL A WITH C BIT CLEARED
4762      ;*ALU FUNCTION (A) CODE=10
4763      ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4764      ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4765      ;*****
4766
4767      ; TEST 76
4768      ;-----
4769 024726 012737 000076 001226 TST76: MOV    #76,TSTNO

```

4770	024734	012737	025102	001216		MOV	#TST77,NEXT	
4771	024742	012737	024774	001220		MOV	#1\$,LOCK	
4772								:R1 CONTAINS BASE DMC11 ADDRESS
4773	024750	104412				MSTCLR		:MASTER CLEAR DMC11
4774	024752	005000				CLR	R0	:MEM + SP ADDRESS
4775	024754	012702	025072			MOV	#5\$,R2	:POINTER TO CORRECT DATA
4776	024760	004737	034676			JSR	PC,MEMLD	:LOAD 8 WORDS OF MAIN MEMORY
4777	024764	035022				MEMDAT		:POINTER TO DATA
4778	024766	004737	034732			JSR	PC,SPLD	:LOAD 8 WORDS OF SP
4779	024772	035032				SPDAT		:POINTER TO DATA
4780	024774	004737	034776		1\$:	JSR	PC,CLRC	:CLEAR C BIT.
4781	025000	042737	000017	025014		BIC	#17,2\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
4782	025006	050037	025014			BIS	R0,2\$	:ADD ADDRESS TO INSTRUCTION
4783	025012	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4784	025014	010000			2\$:	010000		:LOAD MAR
4785	025016	042737	000017	025032		BIC	#17,3\$	:CLEAR ADDRESS OF INSTRUCTION
4786	025024	050037	025032			BIS	R0,3\$	:ADD ADDRESS TO INSTRUCTION
4787	025030	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4788	025032	040600			3\$:	040400!<10*20>		:BR SEL A
4789	025034	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4790	025036	061224				61224		:MOVE BR TO PORT4
4791	025040	111205				MOVB	(R2),R5	:PUT 'EXPECTED' IN R5
4792	025042	116104	000004			MOVB	4(R1),R4	:PUT 'FOUND' IN R4
4793	025046	120504				CMPB	R5,R4	:DATA CORRECT?
4794	025050	001401				BEQ	4\$	:BR IF YES
4795	025052	104015				HLT	15	:ALU ERROR
4796	025054	104401			4\$:	SCOPI		:SW09=1?
4797	025056	005202				INC	R2	:NEXT DATA
4798	025060	005200				INC	R0	:NEXT ADDRESS
4799	025062	022700	000010			CMP	#10,R0	:DONE YET?
4800	025066	001342				BNE	1\$	:BR IF NO
4801	025070	104400				SCOPE		:SCOPE THIS TEST
4802	025072	000	000	377	5\$:	.BYTE	0,0,-1,-1,125,125,252,252	
4803	025075	377	125	125				
4804	025100	252	252					
4805								.FVEN

***** 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

4818	025102	012737	000077	001226	TST77:	MOV	#77,TSTNO	
4819	025110	012737	025256	001216		MOV	#TST100,NEXT	
4820	025116	012737	025150	001220		MOV	#1\$,LOCK	
4821								:R1 CONTAINS BASE DMC11 ADDRESS
4822	025124	104412				MSTCLR		:MASTER CLEAR DMC11
4823	025126	005000				CLR	R0	:MEM + SP ADDRESS
4824	025130	012702	025246			MOV	#5\$,R2	:POINTER TO CORRECT DATA
4825	025134	004737	034676			JSR	PC,MEMLD	:LOAD 8 WORDS OF MAIN MEMORY

4826	025140	035022				MEMDAT		:POINTER TO DATA
4827	025142	004737	034732			JSR	PC,SPLD	:LOAD 8 WORDS OF SP
4828	025146	035032				SPDAT		:POINTER TO DATA
4829	025150	004737	034776		1\$:	JSR	PC,CLRC	:CLEAR C BIT
4830	025154	042737	000017	025170		BIC	#17,2\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
4831	025162	050037	025170			BIS	R0,2\$	:ADD ADDRESS TO INSTRUCTION
4832	025166	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4833	025170	010000			2\$:	010000		:LOAD MAR
4834	025172	042737	000017	025206		BIC	#17,3\$	:CLEAR ADDRESS OF INSTRUCTION
4835	025200	050037	025206			BIS	R0,3\$	:ADD ADDRESS TO INSTRUCTION
4836	025204	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4837	025206	040640			3\$:	040400	<12*20>	:BR A OR NOTB
4838	025210	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4839	025212	061224				61224		:MOVE BR TO PORT4
4840	025214	111205				MOVB	(R2),R5	:PUT 'EXPECTED' IN R5
4841	025216	116104	000004			MOVB	4(R1),R4	:PUT 'FOUND' IN R4
4842	025222	120504				CMPB	R5,R4	:DATA CORRECT?
4843	025224	001401				BEQ	4\$	:BR IF YES
4844	025226	104015				HLT	1\$	:ALU ERROR
4845	025230	104401			4\$:	SCOP1		:SW09-1?
4846	025232	005202				INC	R2	:NEXT DATA
4847	025234	005200				INC	R0	:NEXT ADDRESS
4848	025236	022700	000010			CMP	#10,R0	:DONE YET?
4849	025242	001342				BNE	1\$	:BR IF NO
4850	025244	104400				SCOPE		:SCOPE THIS TEST
4851	025246	377	000	377	5\$:	.BYTE	-1,0,-1,-1,-1,125,252,-1	
4852	025251	377	377	125				
4853	025254	252	377					

.EVEN

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

: TEST 100

4867	025256	012737	000100	001226	TST100:	MOV	#100,TSTNO	
4868	025264	012737	025432	001216		MOV	#TST101,NEXT	
4869	025272	012737	025324	001220		MOV	#1\$,LOCK	
4870								:R1 CONTAINS BASE DMC11 ADDRESS
4871	025300	104412				MSTCLR		:MASTER CLEAR DMC11
4872	025302	005000				CLR	R0	:MEM + SP ADDRESS
4873	025304	012702	025422			MOV	#5\$,R2	:POINTER TO CORRECT DATA
4874	025310	004737	034676			JSR	PC,MEMLD	:LOAD 8 WORDS OF MAIN MEMORY
4875	025314	035022				MEMDAT		:POINTER TO DATA
4876	025316	004737	034732			JSR	PC,SPLD	:LOAD 8 WORDS OF SP
4877	025322	035032				SPDAT		:POINTER TO DATA
4878	025324	004737	034776		1\$:	JSR	PC,CLRC	:CLEAR C BIT
4879	025330	042737	000017	025344		BIC	#17,2\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
4880	025336	050037	025344			BIS	R0,2\$	:ADD ADDRESS TO INSTRUCTION
4881	025342	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```

4882 025344 010000 2$: 010000 ;LOAD MAR
4883 025346 042737 000017 025362 BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
4884 025354 050037 025362 BIS R0,3$ ;ADD ADDRESS TO INSTRUCTION
4885 025360 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4886 025362 040400 <13*20> 3$: 040400 <13*20> ;BR A AND B
4887 025364 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4888 025366 061224 61224 ;MOVE BR TO PORT4
4889 025370 111205 MOVB (R2),R5 ;PUT 'EXPECTED' IN R5
4890 025372 116104 000004 MOVB 4(R1),R4 ;PUT 'FOUND' IN R4
4891 025376 120504 CMPB R5,R4 ;DATA CORRECT?
4892 025400 001401 BEQ 4$ ;BR IF YES
4893 025402 104015 HLT 15 ;ALU ERROR
4894 025404 104401 4$: SCOP1 ;SW09=1?
4895 025406 005202 INC R2 ;NEXT DATA
4896 025410 005200 INC R0 ;NEXT ADDRESS
4897 025412 022700 000010 CMP #10,R0 ;DONE YET?
4898 025416 001342 BNE 1$ ;BR IF NO
4899 025420 104400 SCOPE ;SCOPE THIS TEST
4900 025422 000 000 000 5$: .BYTE 0,0,0,-1,125,0,0,252
4901 025425 377 125 000
4902 025430 000 252
4903 .EVEN
4904
4905
4906 ;***** TEST 101 *****
4907 ;*ALU TEST
4908 ;*TEST OF ALU FUNCTION A OR B WITH C BIT CLEARED
4909 ;*ALU FUNCTION (A OR B) CODE=14
4910 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4911 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4912 ;*****
4913
4914 ; TEST 101
4915 ;-----
4916 025432 012737 000101 001226 TST101: MOV #101,TSTNO
4917 025440 012737 025606 001216 MOV #TST102,NEXT
4918 025446 012737 025500 001220 MOV #1$,LOCK
4919
4920 025454 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
4921 025456 005000 CLR R0 ;MASTER CLEAR DMC11
4922 025460 012702 025576 MOV #5$,R2 ;MEM + SP ADDRESS
4923 025464 004737 034676 JSR PC,MEMLD ;POINTER TO CORRECT DATA
4924 025470 035022 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
4925 025472 004737 034732 JSR PC,SPLD ;POINTER TO DATA
4926 025476 035032 SPDAT ;LOAD 8 WORDS OF SP
4927 025500 004737 034776 1$: JSR PC,CLRC ;POINTER TO DATA
4928 025504 042737 000017 025520 BIC #17,2$ ;CLEAR C BIT
4929 025512 050037 025520 BIS R0,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
4930 025516 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
4931 025520 010000 2$: 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4932 025522 042737 000017 025536 BIC #17,3$ ;LOAD MAR
4933 025530 050037 025536 BIS R0,3$ ;CLEAR ADDRESS OF INSTRUCTION
4934 025534 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
4935 025536 040700 3$: 040400 <14*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4936 025540 104414 ROMCLK ;BR A OR B
4937 025542 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOVE BR TO PORT4

```

```

4938 025544 111205          MOVB (R2),R5          ;PUT 'EXPECTED' IN R5
4939 025546 116104 000004   MOVB 4(R1),R4        ;PUT 'FOUND' IN R4
4940 025552 120504          CMPB R5,R4            ;DATA CORRECT?
4941 025554 001401          BEQ 4$                ;BR IF YES
4942 025556 104015          HLT 15                ;ALU ERROR
4943 025560 104401          4$: SCOP1             ;SW09=1?
4944 025562 005202          INC R2                ;NEXT DATA
4945 025564 005200          INC R0                ;NEXT ADDRESS
4946 025566 022700 000010   CMP #10,R0           ;DONE YET?
4947 025572 001342          BNE 1$                ;BR IF NO
4948 025574 104400          SCOPE                 ;SCOPE THIS TEST
4949 025576 000 377 377 5$: .BYTE 0,-1,-1,-1,125,-1,-1,252
4950 025601 377 125 377
4951 025604 377 252
4952                                     .EVEN
4953
4954
4955                                     ;***** TEST 102 *****
4956                                     ;*ALU TEST
4957                                     ;*TEST OF ALL FUNCTION A XOR B WITH C BIT CLEARED
4958                                     ;*ALU FUNCTION (A XOR B) CODE=15
4959                                     ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4960                                     ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4961                                     ;*****
4962
4963                                     ; TEST 102
4964                                     ;-----
4965 025606 012737 000102 001226 TST102: MOV #102,TSTNO
4966 025614 012737 025762 001216 MOV #TST103,NEXT
4967 025622 012737 025654 001220 MOV #1$,LOCK
4968
4969 025630 104412          MSTCLR                 ;R1 CONTAINS BASE DMC11 ADDRESS
4970 025632 005000          CLR R0                ;MASTER CLEAR DMC11
4971 025634 012702 025752   MOV #5$,R2           ;MEM + SP ADDRESS
4972 025640 004737 034676   JSR PC,MEMLD         ;POINTER TO CORRECT DATA
4973 025644 035022          MEMDAT              ;LOAD 8 WORDS OF MAIN MEMORY
4974 025646 004737 034732   JSR PC,SPLD          ;POINTER TO DATA
4975 025652 035032          SPDAT              ;LOAD 8 WORDS OF SP
4976 025654 004737 034776   JSR PC,CLRC          ;POINTER TO DATA
4977 025660 042737 000017 025674 1$: BIC #17,2$  ;CLEAR C BIT
4978 025666 050037 025674   BIS R0,2$           ;CLEAR ADDRESS FIELD OF INSTRUCTION
4979 025672 104414          ROMCLK              ;ADD ADDRESS TO INSTRUCTION
4980 025674 010000          010000             ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4981 025676 042737 000017 025712 2$: BIC #17,3$  ;LOAD MAR
4982 025704 050037 025712   BIS R0,3$           ;CLEAR ADDRESS OF INSTRUCTION
4983 025710 104414          ROMCLK              ;ADD ADDRESS TO INSTRUCTION
4984 025712 040720          040400!<15*20>      ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4985 025714 104414          ROMCLK              ;BR A XOR B
4986 025716 061224          61224               ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
4987 025720 111205          MOVB (R2),R5          ;MOVE BR TO PORT4
4988 025722 116104 000004   MOVB 4(R1),R4        ;PUT 'EXPECTED' IN R5
4989 025726 120504          CMPB R5,R4            ;PUT 'FOUND' IN R4
4990 025730 001401          BEQ 4$                ;DATA CORRECT?
4991 025732 104015          HLT 15                ;BR IF YES
4992 025734 104401          4$: SCOP1             ;ALU ERROR
4993 025736 005202          IN R2                 ;SW09-1?
                                     ;NEXT DATA

```

```

4994 025740 005200          INC      R0          ;NEXT ADDRESS
4995 025742 022700 000010    CMP      #10,R0      ;DONE YET?
4996 025746 001342          BNE      1$          ;BR IF NO
4997 025750 104400          SCOPE          ;SCOPE THIS TEST
4998 025752      000      377      377 5$: .BYTE 0,-1,-1,0,0,-1,-1,0
4999 025755      000      000      377
5000 025760      377      000
5001                                     .EVEN
5002
5003
5004                                     :***** TEST 103 *****
5005                                     :*ALU TEST
5006                                     :*TEST OF ALU FUNCTION ADD WITH C BIT CLEARED
5007                                     :*ALU FUNCTION (A PLUS B) CODE=00
5008                                     :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5009                                     :*PERFORM THE FUNCTION, VERIFY THE RESULTS
5010                                     :*****
5011
5012                                     : TEST 103
5013                                     :-----
5014 025762 012737 000103 001226 TST103: MOV      #103,TSTNO
5015 025770 012737 026136 001216      MOV      #TST104,NEXT
5016 025776 012737 026030 001220      MOV      #1$,LOCK
5017
5018 026004 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
5019 026006 005000          CLR      R0          ;MASTER CLEAR DMC11
5020 026010 012702 026126      MOV      #5$,R2      ;MEM + SP ADDRESS
5021 026014 004737 034676      JSR      PC,MEMLD      ;POINTER TO CORRECT DATA
5022 026020 035022          MEMDAT          ;LOAD 8 WORDS OF MAIN MEMORY
5023 026022 004737 034732      JSR      PC,SPLD      ;POINTER TO DATA
5024 026026 035032          SPDAT          ;LOAD 8 WORDS OF SP
5025 026030 004737 034776      JSR      PC,CLRC      ;POINTER TO DATA
5026 026034 042737 000017 026050 1$: BIC      #1$,2$      ;CLEAR C BIT!
5027 026042 050037 026050      BIS      R0,2$      ;CLEAR ADDRESS FIELD OF INSTRUCTION
5028 026046 104414          ROMCLK          ;ADD ADDRESS TO INSTRUCTION
5029 026050 010000          010000          ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
5030 026052 042737 000017 026066 2$: BIC      #17,3$      ;LOAD MAR
5031 026060 050037 026066      BIS      R0,3$      ;CLEAR ADDRESS OF INSTRUCTION
5032 026064 104414          ROMCLK          ;ADD ADDRESS TO INSTRUCTION
5033 026066 040400 3$: 040400!<00*20>      ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
5034 026070 104414          ROMCLK          ;BR ADD
5035 026072 061224          61224          ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5036 026074 111205          MOV      (R2),R5      ;MOVE BR TO PORT4
5037 026076 116104 000004      MOV      4(R1),R4      ;PUT 'EXPECTED' IN R5
5038 026102 120504          CMP      R5,R4          ;PUT 'FOUND' IN R4
5039 026104 001401          BEQ      4$          ;DATA CORRECT?
5040 026106 104015          HLT      1$          ;BR IF YES
5041 026110 104401          4$: SCOP1          ;ALU ERROR
5042 026112 005202          INC      R2          ;SW09=1?
5043 026114 005200          INC      R0          ;NEXT DATA
5044 026116 022700 000010    CMP      #10,R0      ;NEXT ADDRESS
5045 026122 001342          BNE      1$          ;DONE YET?
5046 026124 104400          SCOPE          ;BR IF NO
5047 026126      000      377      377 5$: .BYTE 0,-1,-1,376,252,-1,-1,124
5048 026131      376      252      377
5049 026134      377      124

```

```

5050 .EVEN
5051
5052
5053 ***** TEST 104 *****
5054 :*ALU TEST
5055 :*TEST OF ALU FUNCTION 2A W/C WITH C BIT CLEARED
5056 :*ALU FUNCTION (A PLUS A PLUS C) CODE=6
5057 :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5058 :*PERFORM THE FUNCTION, VERIFY THE RESULTS
5059 :*****
5060
5061 : TEST 104
5062 :-----
5063 026136 012737 000104 001226 TST104: MOV #104,TSTNO
5064 026144 012737 026312 001216 MOV #TST105,NEXT
5065 026152 012737 026204 001220 MOV #1$,LOCK
5066
5067 026160 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
5068 026162 005000 CLR RC ;MASTER CLEAR DMC11
5069 026164 012702 026302 MOV #5$,R2 ;MEM + SP ADDRESS
5070 026170 004737 034676 JSR PC,MEMLD ;POINTER TO CORRECT DATA
5071 026174 035022 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
5072 026176 004737 034732 JSR PC,SPLD ;POINTER TO DATA
5073 026202 035032 SPDAT ;LOAD 8 WORDS OF SP
5074 026204 004737 034776 1$: JSR PC,CLRC ;POINTER TO DATA
5075 026210 042737 000017 026224 BIC #17,2$ ;CLEAR C BIT!
5076 026216 050037 026224 BIS R0,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
5077 026222 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5078 026224 010000 2$: 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
5079 026226 042737 000017 026242 BIC #17,3$ ;LOAD MAR
5080 026234 050037 026242 BIS R0,3$ ;CLEAR ADDRESS OF INSTRUCTION
5081 026240 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5082 026242 040540 3$: 040400!<6*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
5083 026244 104414 ROMCLK ;BR 2A W/C
5084 026246 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
5085 026250 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
5086 026252 116104 000004 MOVB 4(R1),R4 ;PUT 'EXPECTED' IN R5
5087 026256 120504 CMPS R5,R4 ;PUT 'FOUND' IN R4
5088 026260 001401 BEQ 4$ ;DATA CORRECT?
5089 026262 104015 HLT 15 ;BR IF YES
5090 026264 104401 4$: SCOP1 ;ALU ERROR
5091 026266 005202 INC R2 ;SW09=1?
5092 026270 005200 INC R0 ;NEXT DATA
5093 026272 022700 000010 CMP #10,R0 ;NEXT ADDRESS
5094 026276 001342 BNE 1$ ;DONE YET?
5095 026300 104400 SCOPE ;BR IF NO
5096 026302 000 000 376 5$: .BYTE 0,0,376,376,252,252,124,124
5097 026305 376 252
5098 026310 124 124
5099 .EVEN
5100
5101 ***** TEST 105 *****
5102 :*ALU TEST
5103 :*TEST OF ALU FUNCTION SUB WITH C BIT CLEARED
5104 :*ALU FUNCTION (A-H) CODE=6
5105

```

```

5106                                     ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5107                                     ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5108                                     ;*****
5109
5110                                     : TEST 105
5111                                     :-----
5112 026312 012737 000105 001226 TST105: MOV #105,TSTNO
5113 026320 012737 026466 001216 MOV #TST106,NEXT
5114 026326 012737 026360 001220 MOV #1$,LOCK
5115
5116 026334 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
5117 026336 005000 CLR R0 ;MASTER CLEAR DMC11
5118 026340 012702 026456 MOV #5$,R2 ;MEM + SP ADDRESS
5119 026344 004737 034676 JSR PC,MEMLD ;POINTER TO CORRECT DATA
5120 026350 035022 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
5121 026352 004737 034732 JSR PC,SPLD ;POINTER TO DATA
5122 026356 035032 SPDAT ;LOAD 8 WORDS OF SP
5123 026360 004737 034776 1$: JSR PC,CLRC ;POINTER TO DATA
5124 026364 042737 000017 026400 BIC #1,2$ ;CLEAR C BIT!
5125 026372 050037 026400 BIS R0,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
5126 026376 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5127 026400 010000 2$: 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5128 026402 042737 000017 026416 BIC #17,3$ ;LOAD MAR
5129 026410 050037 026416 BIS R0,3$ ;CLEAR ADDRESS OF INSTRUCTION
5130 026414 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5131 026416 040740 3$: 040400!<16*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5132 026420 104414 ROMCLK ;BR SUB
5133 026422 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5134 026424 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
5135 026426 116104 000004 MOVB 4(R1),R4 ;PUT 'EXPECTED' IN R5
5136 026432 120504 CMPB R5,R4 ;PUT 'FOUND' IN R4
5137 026434 001401 BEQ 4$ ;DATA CORRECT?
5138 026436 104015 HL 15 ;BR IF YES
5139 026440 104401 4$: SCOP1 ;ALU ERROR
5140 026442 005202 INC R2 ;SW09=1?
5141 026444 005200 INC R0 ;NEXT DATA
5142 026446 022700 000010 CMP #10,R0 ;NEXT ADDRESS
5143 026452 001342 BNE 1$ ;DONE YET?
5144 026454 104400 SCOPE ;BR IF NO
5145 026456 000 001 377 5$: .BYTE 0,1,-1,0,0,253,125,0 ;SCOPE THIS TEST
5146 026461 000 000 53
5147 026464 125 000
5148
5149 .EVEN
5150
5151 ***** TEST 106 *****
5152 ;*ALU TEST
5153 ;*TEST OF ALU FUNCTION ADD W/C WITH C BIT CLEARED
5154 ;*ALU FUNCTION (A PLUS B PLUS C) CODE=01
5155 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5156 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5157 ;*****
5158
5159 : TEST 106
5160 :-----
5161 026466 012737 000105 001226 TST106: MOV #106,TSTNO

```


5162	026474	012737	026642	001216		MOV	#TST107,NEXT	
5163	026502	012737	026534	001220		MOV	#1\$,LOCK	
5164								:R1 CONTAINS BASE DMC11 ADDRESS
5165	026510	104412				MSTCLR		:MASTER CLEAR DMC11
5166	026512	005000				CLR	R0	:MEM + SP ADDRESS
5167	026514	012702	026632			MOV	#5\$,R2	:POINTER TO CORRECT DATA
5168	026520	004737	034676			JSR	PC,MEMLD	:LOAD 8 WORDS OF MAIN MEMORY
5169	026524	035022				MEMDAT		:POINTER TO DATA
5170	026526	004737	034732			JSR	PC,SPLD	:LOAD 8 WORDS OF SP
5171	026532	035032				SPDAT		:POINTER TO DATA
5172	026534	004737	034776		1\$:	JSR	PC,CLRC	:CLEAR C BIT!
5173	026540	042737	000017	026554		BIC	#17,2\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
5174	026546	050037	026554			BIS	R0,2\$	:ADD ADDRESS TO INSTRUCTION
5175	026552	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5176	026554	010000			2\$:	010000		:LOAD MAR
5177	026556	042737	000017	026572		BIC	#17,3\$	:CLEAR ADDRESS OF INSTRUCTION
5178	026564	050037	026572			BIS	R0,3\$	:ADD ADDRESS TO INSTRUCTION
5179	026570	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5180	026572	040420			3\$:	040400!<01*20>		:BR ADD W/C
5181	026574	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5182	026576	061224				61224		:MOVE BR TO PORT4
5183	026600	111205				MOVB	(R2),R5	:PUT 'EXPECTED' IN R5
5184	026602	116104	000004			MOVB	4(R1),R4	:PUT 'FOUND' IN R4
5185	026606	120504				CMPS	R5,R4	:DATA CORRECT?
5186	026610	001401				BEQ	4\$	:BR IF YES
5187	026612	104015				HLT	15	:ALU ERROR
5188	026614	104401			4\$:	SCOPE		:SW09=1?
5189	026616	005202				INC	R2	:NEXT DATA
5190	026620	005200				INC	R0	:NEXT ADDRESS
5191	026622	022700	000010			CMPS	#10,R0	:DONE YET?
5192	026626	001342				BNE	1\$	:BR IF NO
5193	026630	104400				SCOPE		:SCOPE THIS TEST
5194	026632	000	377	377	5\$:	.BYTE	0,-1,-1,376,252,-1,-1,124	
5195	026635	376	252	377				
5196	026640	377	124					

.FLEN

***** TEST 107 *****
 :*ALU TEST
 :*TEST OF ALU FUNCTION SUB W/C WITH C BIT CLEARED
 :*ALU FUNCTION (A-B-C) CODE=2
 :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
 :*PERFORM THE FUNCTION, VERIFY THE RESULTS
 :*****

: TEST 107

5200								
5201								
5202								
5203								
5204								
5205								
5206								
5207								
5208								
5209								
5210	026642	012737	000107	001226	TST107:	MOV	#107,TSTNO	
5211	026650	012737	027016	001216		MOV	#TST110,NEXT	
5212	026656	012737	026710	001220		MOV	#1\$,LOCK	
5213								:R1 CONTAINS BASE DMC11 ADDRESS
5214	026664	104412				MSTCLR		:MASTER CLEAR DMC11
5215	026666	005000				CLR	R0	:MEM + SP ADDRESS
5216	026670	012702	027006			MOV	#5\$,R2	:POINTER TO CORRECT DATA
5217	026674	004737	034676			JSR	PC,MEMLD	:LOAD 8 WORDS OF MAIN MEMORY

PC	OPCODE	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
----	--------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------


```

5274 027104 010000 2$: 010000 ;LOAD MAR
5275 027106 042737 000017 027122 BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
5276 027114 050037 027122 BIS R0,3$ ;ADD ADDRESS TO INSTRUCTION
5277 027120 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5278 027122 040460 3$: 040400.<3*20> ;BR INC A
5279 027124 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5280 027126 061224 61224 ;MOVE BR TO PORT4
5281 027130 111205 MOVB (R2),R5 ;PUT 'EXPECTED' IN R5
5282 027132 116104 000004 MOVB 4(R1),R4 ;PUT 'FOUND' IN R4
5283 027136 120504 CMPB R5,R4 ;DATA CORRECT?
5284 027140 001401 BEQ 4$ ;BR IF YES
5285 027142 104015 HLT 15 ;ALU ERROR
5286 027144 104401 4$: SCOP1 ;SW09=1?
5287 027146 005202 INC R2 ;NEXT DATA
5288 027150 005200 INC R0 ;NEXT ADDRESS
5289 027152 022700 000010 CMP #10,R0 ;DONE YET?
5290 027156 001342 BNE 1$ ;BR IF NO
5291 027160 104400 SCOPE ;SCOPE THIS TEST
5292 027162 001 001 000 5$: .BYTE 1,1,0,0,126,126,253,253
5293 027165 000 126 126
5294 027170 253 253
5295 .EVEN
5296
5297
5298 ;***** TEST 111 *****
5299 ;*ALU TEST
5300 ;*TEST OF ALU FUNCTION 2A WITH C BIT CLEARED
5301 ;*ALU FUNCTION (A PLUS A) CODE-5
5302 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5303 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5304 ;*****
5305
5306 ; TEST 111
5307 ;-----
5308 027172 012737 000111 001226 TST111: MOV #111,TSTNO
5309 027200 012737 027346 001216 MOV #TST112,NEXT
5310 027206 012737 027240 001220 MOV #1$,LOCK
5311
5312 027214 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
5313 027216 005000 CLR R0 ;MASTER CLEAR DMC11
5314 027220 012702 027336 MOV #5$,R2 ;MEM + SP ADDRESS
5315 027224 004737 034676 JSR PC,MEMLD ;POINTER TO CORRECT DATA
5316 027230 035022 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
5317 027232 004737 034732 JSR PC,SPLD ;POINTER TO DATA
5318 027236 035032 SPDAT ;LOAD 8 WORDS OF SP
5319 027240 004737 034776 1$: JSR PC,CLRC ;POINTER TO DATA
5320 027244 042737 000017 027260 BIC #17,2$ ;CLEAR C BIT
5321 027252 050037 027260 BIS R0,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
5322 027256 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5323 027260 010000 2$: 010000 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5324 027262 042737 000017 027276 BIC #17,3$ ;LOAD MAR
5325 027270 050037 027276 BIS R0,3$ ;CLEAR ADDRESS OF INSTRUCTION
5326 027274 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5327 027276 040520 3$: 040400.<5*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5328 027300 104414 ROMCLK ;BR 2A
5329 027302 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;MOVE BR TO PORT4

```

```

5330 027304 111205      MOVB      (R2),R5      ;PUT 'EXPECTED' IN R5
5331 027306 116104 000004  MOVB      4(R1),R4      ;PUT 'FOUND' IN R4
5332 027312 120504      CMPB      R5,R4      ;DATA CORRECT?
5333 027314 001401      BEQ       4$          ;BR IF YES
5334 027316 104015      HLT        1$          ;ALU ERROR
5335 027320 104401      4$: SCOP1      ;SW09=1?
5336 027322 005202      INC        R2          ;NEXT DATA
5337 027324 005200      INC        R0          ;NEXT ADDRESS
5338 027326 022700 000010  CMP       #10,R0      ;DONE YET?
5339 027332 001342      BNE       1$          ;BR IF NC
5340 027334 104400      SCOPE      ;SCOPE THIS TEST
5341 027336      000      000      376 5$: .BYTE 0,0,376,376,252,252,124,124
5342 027341      376      252      252
5343 027344      124      124
5344                                     .EVEN
5345
5346
5347                                     :***** TEST 112 *****
5348                                     :*ALU TEST
5349                                     :*TEST OF ALU FUNCTION A PLUS C WITH C BIT CLEARED
5350                                     :*ALU FUNCTION (A PLUS C) CODE=4
5351                                     :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5352                                     :*PERFORM THE FUNCTION, VERIFY THE RESULTS
5353                                     :*****
5354
5355                                     : TEST 112
5356                                     :-----
5357 027346 012737 000112 001226 TST112: MOV      #112,TSTNO
5358 027354 012737 027522 001216 MOV      #TST113,NEXT
5359 027362 012737 027414 001220 MOV      #1$,LOCK
5360
5361 027370 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
5362 027372 005000      CLR        R0          ;MASTER CLEAR DMC11
5363 027374 012702 027512  MOV      #5$,R2      ;MEM + SP ADDRESS
5364 027400 004737 034676  JSR      PC,MEMLD    ;POINTER TO CORRECT DATA
5365 027404 035022      MEMDAT      ;LOAD 8 WORDS OF MAIN MEMORY
5366 027406 004737 034732  JSR      PC,SPLD    ;POINTER TO DATA
5367 027412 035032      SPDAT      ;LOAD 8 WORDS OF SP
5368 027414 004737 034776  JSR      PC,CLRC    ;POINTER TO DATA
5369 027420 042737 000017 027434 1$: BIC      #17,2$    ;CLEAR C BIT!
5370 027426 050037 027434  BIS      R0,2$      ;CLEAR ADDRESS FIELD OF INSTRUCTION
5371 027432 104414      ROMCLK      ;ADD ADDRESS TO INSTRUCTION
5372 027434 010000 010000 2$: ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5373 027436 042737 000017 027452  BIC      #17,3$    ;LOAD MAR
5374 027444 050037 027452  BIS      R0,3$      ;CLEAR ADDRESS OF INSTRUCTION
5375 027450 104414      ROMCLK      ;ADD ADDRESS TO INSTRUCTION
5376 027452 040500 040400 3$: ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5377 027454 104414      61224      ;BR A PLUS C
5378 027456 061224      MOVB      (R2),R5      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5379 027460 111205      MOVB      4(R1),R4      ;MOVE BR TO PORT4
5380 027462 116104 000004  CMPB      R5,R4      ;PUT 'EXPECTED' IN R5
5381 027466 120504      BEQ       4$          ;PUT 'FOUND' IN R4
5382 027470 001401      HLT        1$          ;DATA CORRECT?
5383 027472 104015      4$: SCOP1      ;BR IF YES
5384 027474 104401      INC        R2          ;ALU ERROR
5385 027476 005202      INC        R2          ;SW09 1?
                                     ;NEXT DATA

```

```

5386 027500 005200          INC      R0          ;NEXT ADDRESS
5387 027502 022700 000010  CMP      #10,R0      ;DONE YET?
5388 027506 001342          BNE      1$          ;BR IF NO
5389 027510 104400          SCOPE          ;SCOPE THIS TEST
5390 027512      000      000      377 5$: .BYTE 0,0,-1,-1,125,125,252,252
5391 027515      377      125      125
5392 027520      252      252
5393                                     .EVEN
5394
5395
5396                                     ;***** TEST 113 *****
5397                                     ;*ALU TEST
5398                                     ;*TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT CLEARED
5399                                     ;*ALU FUNCTION (A-B-1) CODE-17
5400                                     ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5401                                     ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5402                                     ;*****
5403
5404                                     ; TEST 113
5405                                     ;-----
5406 027522 012737 000113 001226 TST113: MOV      #113,TSTND
5407 027530 012737 027676 001216      MOV      #TST114,NEXT
5408 027536 012737 027570 001220      MOV      #1$,LOCK
5409
5410 027544 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
5411 027546 005000          CLR      RC          ;MASTER CLEAR DMC11
5412 027550 012702 027666  MOV      #5$,R2      ;MEM + SP ADDRESS
5413 027554 004737 034676  JSR      PC,MEMLD    ;POINTER TO CORRECT DATA
5414 027560 035022          MEMDAT          ;LOAD 8 WORDS OF MAIN MEMORY
5415 027562 004737 034732  JSR      PC,SPLD    ;POINTER TO DATA
5416 027566 035032          SPDAT          ;LOAD 8 WORDS OF SP
5417 027570 004737 034776  JSR      PC,CLRC    ;POINTER TO DATA
5418 027574 042737 000017 027610 1$: BIC      #1$,2$    ;CLEAR C BIT
5419 027602 050037 027610      BIS      R0,3$    ;CLEAR ADDRESS FIELD OF INSTRUCTION
5420 027606 104414          ROMCLK          ;ADD ADDRESS TO INSTRUCTION
5421 027610 010000          010000          ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5422 027612 042737 000017 027626 2$: BIC      #17,3$    ;LOAD MAR
5423 027620 050037 027626      BIS      R0,3$    ;CLEAR ADDRESS OF INSTRUCTION
5424 027624 104414          ROMCLK          ;ADD ADDRESS TO INSTRUCTION
5425 027626 040760          040400.<17*20>    ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
5426 027630 104414          ROMCLK          ;BR 2'S COMP SUB
5427 027632 061224          61224          ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
5428 027634 111205          MOVB      (R2),R5    ;MOVE BR TO PORT4
5429 027636 116104 000004  MOVB      4(R1),R4    ;PUT 'EXPECTED' IN R5
5430 027642 120504          CMPB      R5,R4      ;PUT 'FOUND' IN R4
5431 027644 001401          BEQ      4$          ;DATA CORRECT?
5432 027646 104015          HLT      1$          ;BR IF YES
5433 027650 104401          4$: SCOP1          ;ALU ERROR
5434 027652 005202          INC      R2          ;SW09-1?
5435 027654 005200          INC      R0          ;NEXT DATA
5436 027656 022700 000010  CMP      #10,R0      ;NEXT ADDRESS
5437 027662 001342          BNE      1$          ;DONE YET?
5438 027664 104400          SCOPE          ;BR IF NO
5439 027666      377      000      376 5$: .BYTE -1,0,376,-1,-1,252,124,-1
5440 027671      377      377      252
5441 027674      24      377

```

5442 .EVEN

5443
5444
5445
5446
5447
5448
5449
5450
5451
5452
5453
5454
5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465
5466
5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487
5488
5489
5490
5491
5492
5493
5494
5495
5496
5497

***** 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

```

-----
TST114: MOV    #114,TSTNO
        MOV    #TST115,NEXT
        MOV    #1$,LOCK

        MSTCLR
        CLR    RC
        MOV    #5$,R2
        JSR    PC,MEMLD
        MEMDAT
        JSR    PC,SPLD
        SPDAT
        JSR    PC,CLRC
        BIC    #17,2$
        BIS    R0,2$
        ROMCLK
        010000
        BIC    #17,3$
        BIS    R0,3$
        ROMCLK
        040400!<7*20>
        ROMCLK
        61224
        MOVB   (R2),R5
        MOVB   4(R1),R4
        CMPB   R5,R4
        BEQ    4$
        HLT    1$
        SCOPI
        INC    R2
        INC    R0
        CMP    #10,R0
        BNE    1$
        SCOPE
        .BYTE  -1,-1,376,376,124,124,251,251
  
```

: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
 :CLEAR C BIT!
 :CLEAR ADDRESS FIELD OF INSTRUCTION
 :ADD ADDRESS TO INSTRUCTION
 :NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
 :LOAD MAR
 :CLEAR ADDRESS OF INSTRUCTION
 :ADD ADDRESS TO INSTRUCTION
 :NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
 :BR DEC 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

.EVEN

***** TEST 115 *****
 : *ALU TEST
 : *TEST OF ALU FUNCTION SEL B WITH C BIT SET
 : *ALU FUNCTION (B) CODE=11

```

5498 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5499 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5500 ;*****
5501
5502 ; TEST 115
5503 ;-----
5504 030052 012737 000115 001226 TST115: MOV #115,TSTNO
5505 030060 012737 030226 001216 MOV #TST116,NEXT
5506 030066 012737 030120 001220 MOV #1$,LOCK
5507
5508 030074 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
5509 030076 005000 CLR ;MASTER CLEAR DMC11
5510 030100 012702 030216 MOV #5$,R2 ;MEM + SP ADDRESS
5511 030104 004737 034676 JSR PC,MEMLD ;POINTER TO CORRECT DATA
5512 030110 035022 MEMDAT ;LOAD 8 WORDS OF MAIN MEMORY
5513 030112 004737 034732 JSR PC,SPLD ;POINTER TO DATA
5514 030116 035032 SPDAT ;LOAD 8 WORDS OF SP
5515 030120 004737 035010 1$: JSR PC,SETC ;POINTER TO DATA
5516 030124 042737 000017 030140 BIC #1$,2$ ;SET C BIT
5517 030132 050037 030140 BIS R0,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
5518 030136 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5519 030140 010000 010000 2$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5520 030142 042737 000017 030156 BIC #17,3$ ;LOAD MAR
5521 030150 050037 030156 BIS R0,3$ ;CLEAR ADDRESS OF INSTRUCTION
5522 030154 104414 ROMCLK ;ADD ADDRESS TO INSTRUCTION
5523 030156 040620 3$: 040400!<11*20> ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
5524 030160 104414 ROMCLK ;BR SEL B
5525 030162 061224 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5526 030164 111205 MOVB (R2),R5 ;MOVE BR TO PORT4
5527 030166 116104 000004 MOVB 4(R1),R4 ;PUT 'EXPECTED' IN R5
5528 030172 120504 CMPB R5,R4 ;PUT 'FOUND' IN R4
5529 030174 001401 BEQ 4$ ;DATA CORRECT?
5530 030176 104015 HLT 1$ ;BR IF YES
5531 030200 104401 4$: SCOP1 ;ALU ERROR
5532 030202 005202 INC R2 ;SW09-1?
5533 030204 005200 INC R0 ;NEXT DATA
5534 030206 022700 000010 CMP #10,R0 ;NEXT ADDRESS
5535 030212 001342 BNE 1$ ;DONE YET?
5536 030214 104400 SCOPF ;BR IF NO
5537 030216 000 377 000 5$: .BYTE 0,-1,0,-1,125,252,125,252 ;SCOPE THIS TEST
5538 030221 377 125 252
5539 030224 125 252
5540 .EVEN
5541
5542
5543 ;***** TEST 116 *****
5544 ;*ALU TEST
5545 ;*TEST OF ALU FUNCTION SEL A WITH C BIT SET
5546 ;*ALU FUNCTION (A) CODE=10
5547 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5548 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5549 ;*****
5550
5551 ; TEST 116
5552 ;-----
5553 030226 012737 000116 001226 TST116: MOV #116,TSTNO

```

```

5554 030234 012737 030402 001216 MOV #TST117,NEXT
5555 030242 012737 030274 001220 MOV #1$,LOCK
5556                                     ;R1 CONTAINS BASE DMC11 ADDRESS
5557 030250 104412 MSTCLR ;MASTER CLEAR DMC11
5558 030252 005000 CLR R0 ;MEM + SP ADDRESS
5559 030254 012702 030372 MOV #5$,R2 ;POINTER TO CORRECT DATA
5560 030260 004737 034676 JSR PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMORY
5561 030264 035022 MEMDAT ;POINTER TO DATA
5562 030266 004737 034732 JSR PC,SPLD ;LOAD 8 WORDS OF SP
5563 030272 035032 SPDAT ;POINTER TO DATA
5564 030274 004737 035010 1$: JSR PC,SETC ;SET C BIT!
5565 030300 042737 000017 030314 BIC #17,2$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
5566 030306 050037 030314 BIS R0,2$ ;ADD ADDRESS TO INSTRUCTION
5567 030312 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5568 030314 010000 010000 2$: ;LOAD MAR
5569 030316 042737 000017 030332 BIC #17,3$ ;CLEAR ADDRESS OF INSTRUCTION
5570 030324 050037 030332 BIS R0,3$ ;ADD ADDRESS TO INSTRUCTION
5571 030330 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5572 030332 040600 3$: 040400.<10*20> ;BR SEL A
5573 030334 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5574 030336 061224 61224 ;MOVE BR TO PORT4
5575 030340 111205 MOVB (R2),R5 ;PUT 'EXPECTED' IN R5
5576 030342 116104 000004 MOVB 4(R1),R4 ;PUT 'FOUND' IN R4
5577 030346 120504 CMPB R5,R4 ;DATA CORRECT?
5578 030350 001401 BEQ 4$ ;BR IF YES
5579 030352 104015 HLT 15 ;ALU ERROR
5580 030354 104401 4$: SCOP1 ;SW09=1?
5581 030356 005202 INC R2 ;NEXT DATA
5582 030360 005200 INC R0 ;NEXT ADDRESS
5583 030362 022700 000010 CMP #10,R0 ;DONE YET?
5584 030366 001342 BNE 1$ ;BR IF NO
5585 030370 104400 SCOPE ;SCOPE THIS TEST
5586 030372 000 000 377 5$: .BYTE C,0,-1,-1,125,125,252,252
5587 030375 377 125
5588 030400 252 252
5589 .FVEN
5590
5591
5592 ;***** TEST 117 *****
5593 ;*ALU TEST
5594 ;*TEST OF ALU FUNCTION A OR NOTB WITH C BIT SET
5595 ;*ALU FUNCTION (A OR NOTB) CODE-12
5596 ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5597 ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5598 ;*****
5599
5600 ; TEST 117
5601 ;-----
5602 030402 012737 000117 001226 TST117: MOV #117,TSTNO
5603 030410 012737 030556 001216 MOV #TST120,NEXT
5604 030416 012737 030450 001220 MOV #1$,LOCK
5605                                     ;R1 CONTAINS BASE DMC11 ADDRESS
5606 030424 104412 MSTCLR ;MASTER CLEAR DMC11
5607 030426 005000 CLR R0 ;MEM + SP ADDRESS
5608 030430 012702 030546 MOV #5$,R2 ;POINTER TO CORRECT DATA
5609 030434 004737 034676 JSR PC,MEMLD ;LOAD 8 WORDS OF MAIN MEMORY

```


..EVEN

: TEST 120

5651	030556	012737	000120	001226	TST120:	MOV	#120,TSTNO			
5652	030564	012737	030732	001216		MOV	#TST121,NEXT			
5653	030572	012737	030624	001220		MOV	#1\$,LOCK			
5654										:R1 CONTAINS BASE DMC11 ADDRESS
5655	030600	104412				MSTCLR				:MASTER CLEAR DMC11
5656	030602	005000				CLR	R0			:MEM + SP ADDRESS
5657	030604	012702	030722			MOV	#5\$,R2			:POINTER TO CORRECT DATA
5658	030610	004737	034676			JSR	PC,MEMLD			:LOAD 8 WORDS OF MAIN MEMORY
5659	030614	035022				MEMDAT				:POINTER TO DATA
5660	030616	004737	034732			JSR	PC,SPLD			:LOAD 8 WORDS OF SP
5661	030622	035032				SPDAT				:POINTER TO DATA
5662	030624	004737	035010		1\$:	JSR	PC,SETC			:SET C BIT!
5663	030630	042737	000017	030644		BIC	#1\$,2\$			:CLEAR ADDRESS FIELD OF INSTRUCTION
5664	030636	050037	030644			BIS	R0,2\$			:ADD ADDRESS TO INSTRUCTION
5665	030642	104414				ROMCLK				:NEXT WORD IS INSTRUCTION. ROMCLK P- 5304

5666	030644	010000			2\$:	010000	:LOAD MAR
5667	030646	042737	000017	030662		BIC #17,3\$	:CLEAR ADDRESS OF INSTRUCTION
5668	030654	050037	030662			BIS R0,3\$	:ADD ADDRESS TO INSTRUCTION
5669	030660	104414				ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5670	030662	040660			3\$:	040400.<13*20>	:BR A AND B
5671	030664	104414				ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5672	030666	061224				61224	:MOVE BR TO PORT4
5673	030670	111205				MOVB (R2),R5	:PUT 'EXPECTED' IN R5
5674	030672	116104	000004			MOVB 4(R1),R4	:PUT 'FOUND' IN R4
5675	030676	120504				CMPB R5,R4	:DATA CORRECT?
5676	030700	001401				BEQ 4\$	:BR IF YES
5677	030702	104015				HLT 15	:ALU ERROR
5678	030704	104401			4\$:	SCOP1	:SW09=1?
5679	030706	005202				INC R2	:NEXT DATA
5680	030710	005200				INC R0	:NEXT ADDRESS
5681	030712	022700	000010			CMP #10,R0	:DONE YET?
5682	030716	001342				BNE 1\$	:BR IF NO
5683	030720	104400				SCOPE	:SCOPE THIS TEST
5684	030722	000	000	000	5\$:	.BYTE 0,0,0,-1,125,0,0,252	
5685	030725	377	125	000			
5686	030730	000	252				

.EVEN

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

: TEST 121

5700	030732	012737	000121	001226	TST121:	MOV #121,TSTNO	
5701	030740	012737	031106	001216		MOV #TST122,NEXT	
5702	030746	012737	031000	001220		MOV #1\$,LOCK	
5703							
5704	030754	104412				MSTCLR	:R1 CONTAINS BASE DMC11 ADDRESS
5705	030756	005000				CLR R0	:MASTER CLEAR DMC11
5706	030760	012702	031076			MOV #5\$,R2	:MEM + SP ADDRESS
5707	030764	004737	034676			JSR PC,MEMLD	:POINTER TO CORRECT DATA
5708	030770	035022				MEMDAT	:LOAD 8 WORDS OF MAIN MEMORY
5709	030772	004737	034732			JSR PC,SPLD	:POINTER TO DATA
5710	030776	035032				SPDAT	:LOAD 8 WORDS OF SP
5711	031000	004737	035010		1\$:	JSR PC,SETC	:POINTER TO DATA
5712	031004	042737	000017	031020		BIC #17,2\$	:SET C BIT!
5713	031012	050037	031020			BIS R0,2\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
5714	031016	104414				ROMCLK	:ADD ADDRESS TO INSTRUCTION
5715	031020	010000			2\$:	010000	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5716	031022	042737	000017	031036		BIC #17,3\$	:LOAD MAR
5717	031030	050037	031036			BIS R0,3\$	:CLEAR ADDRESS OF INSTRUCTION
5718	031034	104414				ROMCLK	:ADD ADDRESS TO INSTRUCTION
5719	031036	040700			3\$:	040400.<14*PC>	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5720	031040	104414				ROMCLK	:BR A OR B
5721	031042	061224				61224	:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304

:MOVE BR TO PORT4


```

5722 031044 111205          MOVB      (R2),R5          ;PUT 'EXPECTED' IN R5
5723 031046 116104 000004    MOVB      4(R1),R4        ;PUT 'FOUND' IN R4
5724 031052 120504          CMPB      R5,R4            ;DATA CORRECT?
5725 031054 001401          BEQ       4$              ;BR IF YES
5726 031056 104015          HLT       15              ;ALU ERROR
5727 031060 104401          4$: SCOP1      ;SW09=1?
5728 031062 005202          INC       R2              ;NEXT DATA
5729 031064 005200          INC       R0              ;NEXT ADDRESS
5730 031066 022700 000010    CMP       #10,R0          ;DONE YET?
5731 031072 001342          BNE       1$              ;BR IF NC
5732 031074 104400          SCOPE     ;SCOPE THIS TEST
5733 031076      000      377      377 5$: .BYTE      0,-1,-1,-1,125,-1,-1,252
5734 031101      377      125      377
5735 031104      377      252
5736                                     .EVEN
5737
5738
5739                                     ;***** TEST 122 *****
5740                                     ;*ALU TEST
5741                                     ;*TEST OF ALU FUNCTION A XOR B WITH C BIT SET
5742                                     ;*ALU FUNCTION (A XOR B) CODE 15
5743                                     ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5744                                     ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5745                                     ;*****
5746
5747                                     ; TEST 122
5748                                     ;-----
5749 031106 012737 000122 001226 TST122: MOV      #122,TSTNO
5750 031114 012737 031262 001216    MOV      #TST123,NEXT
5751 031122 012737 031154 001220    MOV      #1$,LOCK
5752
5753 031130 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
5754 031132 005000          CLR          ;MASTER CLEAR DMC11
5755 031134 012702 031252    MOV      #5$,R2          ;MEM + SP ADDRESS
5756 031140 004737 034676    JSR      PC,MEMLD        ;POINTER TO CORRECT DATA
5757 031144 035022          MEMDAT        ;LOAD 8 WORDS OF MAIN MEMORY
5758 031146 004737 034732    JSR      PC,SPLD        ;POINTER TO DATA
5759 031152 035032          SPDAT        ;LOAD 8 WORDS OF SP
5760 031154 004737 035010    JSR      PC,SETC        ;POINTER TO DATA
5761 031160 042737 000017 031174 1$: BIC      #17,2$      ;SET C BIT
5762 031166 050037 031174    BIS      R0,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5763 031172 104414          ROMCLK        ;ADD ADDRESS TO INSTRUCTION
5764 031174 010000 010000 4$: ROMCLK        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5765 031176 042737 000017 031212    BIC      #17,3$      ;LOAD MAR
5766 031204 050037 031212    BIS      R0,3$          ;CLEAR ADDRESS OF INSTRUCTION
5767 031210 104414          ROMCLK        ;ADD ADDRESS TO INSTRUCTION
5768 031212 040400 <15*20> 3$: ROMCLK        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5769 031214 104414          ROMCLK        ;BR A XOR B
5770 031216 061224          61224          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5771 031220 111205          MOVB      (R2),R5          ;MOVE BR TO PORT4
5772 031222 116104 000004    MOVB      4(R1),R4        ;PUT 'EXPECTED' IN R5
5773 031226 120504          CMPB      R5,R4            ;PUT 'FOUND' IN R4
5774 031230 001401          BEQ       4$              ;DATA CORRECT?
5775 031232 104015          HLT       15              ;BR IF YES
5776 031234 104401          SCOPE     ;ALU ERROR
5777 031236 005202          INC       R2              ;SW09=1?
                                     ;NEXT DATA

```

```

5778 031240 005200          INC      R0          ;NEXT ADDRESS
5779 031242 022700 000010    CMP      #10,R0      ;DONE YET?
5780 031246 001342          BNE      1$          ;BR IF NO
5781 031250 104400          SCOPE          ;SCOPE THIS TEST
5782 031252 000 377 377 5$: .BYTE 0,-1,-1,0,0,-1,-1,0
5783 031255 000 000 377
5784 031260 377 000
5785                                     .EVEN
5786
5787
5788                                     ***** TEST 123 *****
5789                                     ;*ALU TEST
5790                                     ;*TEST OF ALU FUNCTION ADD WITH C BIT SET
5791                                     ;*ALU FUNCTION (A PLUS B) CODE=00
5792                                     ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5793                                     ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5794                                     ;*****
5795
5796                                     ; TEST 123
5797                                     ;-----
5798 031262 012737 000123 001226 1$123: MOV      #123,TSTNG
5799 031270 012737 031436 001216    MOV      #TST124,NEXT
5800 031276 012737 031330 001220    MOV      #1$,LOCK
5801
5802 031304 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
5803 031306 005000          CLR      R0          ;MASTER CLEAR DMC11
5804 031310 012702 031426    MOV      #5$,R2      ;MEM + SP ADDRESS
5805 031314 004737 034676    JSR      PC,MEMLD     ;POINTER TO CORRECT DATA
5806 031320 035022          MEMDAT          ;LOAD 8 WORDS OF MAIN MEMORY
5807 031322 004737 034732    JSR      PC,SPLD     ;POINTER TO DATA
5808 031326 035032          SPDAT          ;LOAD 8 WORDS OF SP
5809 031330 004737 035010    JSR      PC,SETC     ;POINTER TO DATA
5810 031334 042737 000017 031350 1$: JSR      PC,SETC     ;SET C BIT
5811 031342 050037 031350    BIC      #1$,2$      ;CLEAR ADDRESS FIELD OF INSTRUCTION
5812 031346 104414          ROMCLK          ;ADD ADDRESS TO INSTRUCTION
5813 031350 010000          BIC      #17,3$      ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5814 031352 042737 000017 031366 2$: BIC      #17,3$      ;LOAD MAR
5815 031360 050037 031366    BIS      R0,3$      ;CLEAR ADDRESS OF INSTRUCTION
5816 031364 104414          ROMCLK          ;ADD ADDRESS TO INSTRUCTION
5817 031366 040400 3$: 040400!<00*20>      ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5818 031370 104414          ROMCLK          ;BR ADD
5819 031372 061224          61224          ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
5820 031374 111205          MOV      (R2),R5      ;MOVE BR TO PORT4
5821 031376 116104 000004    MOV      4(R1),R4     ;PUT 'EXPECTED' IN R5
5822 031402 120504          CMP      R5,R4        ;PUT 'FOUND' IN R4
5823 031404 001401          BEQ      4$          ;DATA CORRECT?
5824 031406 104015          HLT      1$          ;BR IF YES
5825 031410 104401          4$: SCOP1          ;ALU ERROR
5826 031412 005202          INC      R2          ;SW09-1?
5827 031414 005200          INC      R0          ;NEXT DATA
5828 031416 022700 000010    CMP      #10,R0      ;NEXT ADDRESS
5829 031422 001342          BNE      1$          ;DONE YET?
5830 031424 104400          SCOPE          ;BR IF NO
5831 031426 000 377 377 5$: .BYTE 0,-1,-1,0,0,-1,-1,0 ;SCOPE THIS TEST
5832 031431 376 252 377
5833 031434 377 252

```

5834 .EVEN

```

5836:***** TEST 124 *****
5837:*ALU TEST
5838:*TEST OF ALU FUNCTION 2A W/C WITH C BIT SET
5839:*ALU FUNCTION (A PLUS A PLUS C) CODE=6
5840:*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5841:*PERFORM THE FUNCTION, VERIFY THE RESULTS
5842:*****
5843:

```

5845						: TEST 124	
5846						-----	
5847	031436	012737	000124	001226	TST124:	MOV #124,TSTNO	
5848	031444	012737	031612	001216		MOV #TST125,NEXT	
5849	031452	012737	031504	001220		MOV #1\$,LOCK	
5850							:R1 CONTAINS BASE DMC11 ADDRESS
5851	031460	104412			MSTCLR		:MASTER CLEAR DMC11
5852	031462	005000			CLR RC		:MEM + SP ADDRESS
5853	031464	012702	031602		MOV #5\$,R2		:POINTER TO CORRECT DATA
5854	031470	004737	034676		JSR PC,MEMLD		:LOAD 8 WORDS OF MAIN MEMORY
5855	031474	035022			MEMDAT		:POINTER TO DATA
5856	031476	004737	034732		JSR PC,SPLD		:LOAD 8 WORDS OF SP
5857	031502	035032			SPDAT		:POINTER TO DATA
5858	031504	004737	035010		JSR PC,SETC		:SET C BIT!
5859	031510	042737	000017	031524	BIC #17,2\$		:CLEAR ADDRESS FIELD OF INSTRUCTION
5860	031516	050037	031524		BIS R0,2\$		:ADD ADDRESS TO INSTRUCTION
5861	031522	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5862	031524	010000			010000		:LOAD MAR
5863	031526	042737	000017	031542	BIC #17,3\$		:CLEAR ADDRESS OF INSTRUCTION
5864	031534	050037	031542		BIS R0,3\$		:ADD ADDRESS TO INSTRUCTION
5865	031540	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5866	031542	040540			040400!<6*20>		:BR 2A W/C
5867	031544	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5868	031546	061224			61224		:MOVE BR TO PORT4
5869	031550	111205			MOVB (R2),R5		:PUT 'EXPECTED' IN R5
5870	031552	116104	000004		MOVB 4(R1),R4		:PUT 'FOUND' IN R4
5871	031556	120504			CMPB R5,R4		:DATA CORRECT?
5872	031560	001401			BEQ 4\$		:BR IF YES
5873	031562	104015			HIT 1\$		:ALU ERROR
5874	031564	104401			SCOPI		:SW09=1?
5875	031566	005202			INC R2		:NEXT DATA
5876	031570	005200			INC R0		:NEXT ADDRESS
5877	031572	022700	000017		CMP #10,R0		:DONE YET?
5878	031576	001342			BNE 1\$		:BR IF NO
5879	031600	104400			SCOPE		:SCOPE THIS TEST
5880	031602	001	001	377	58:	.BYTE 1,1,-1,-1,253,253,25,125	
5881	031605	377	053	377			
5882	031610	125	053	125			

5883 .F.4A

```

5886      : ..... TEST 125 .....
5887      : *ALU TEST
5888      : *TEST OF ALU FUNCTION SUB WITH BIT SET
5889      : *ALU FUNCTION (A-B) CODE 16

```

```

5890      : *LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5891      : *PERFORM THE FUNCTION, VERIFY THE RESULTS
5892      : *****
5893
5894      : TEST 125
5895      : -----
5896 031612 012737 000125 001226 TST125: MOV #125,TSTNO
5897 031620 012737 031766 001216      MOV #TST126,NEXT
5898 031626 012737 031660 001220      MOV #1$,LOCK
5899
5900 031634 104412      MSTCLR      :R1 CONTAINS BASE DMC11 ADDRESS
5901 031636 005000      CLR R0      :MASTER CLEAR DMC11
5902 031640 012702 031756      MOV #5$,R2      :MEM + SP ADDRESS
5903 031644 004737 034676      JSR PC,MEMLD      :POINTER TO CORRECT DATA
5904 031650 035022      MEMDAT      :LOAD 8 WORDS OF MAIN MEMORY
5905 031652 004737 034732      JSR PC,SPLD      :POINTER TO DATA
5906 031656 035032      SPDAT      :LOAD 8 WORDS OF SP
5907 031660 004737 035010      JSR PC,SETC      :POINTER TO DATA
5908 031664 042737 000017 031700 1$: BIC #1,2$      :SET C BIT!
5909 031672 050037 031700      BIS R0,2$      :CLEAR ADDRESS FIELD OF INSTRUCTION
5910 031676 104414      ROMCLK      :ADD ADDRESS TO INSTRUCTION
5911 031700 010000      010000      :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5912 031702 042737 000017 031716 2$: BIC #17,3$      :LOAD MAR
5913 031710 050037 031716      BIS R0,3$      :CLEAR ADDRESS OF INSTRUCTION
5914 031714 104414      ROMCLK      :ADD ADDRESS TO INSTRUCTION
5915 031716 040740      040400.<16*20> 3$: ROMCLK      :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5916 031720 104414      61224      :BR SUB
5917 031722 061224      MOVB (R2),R5      :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5918 031724 111205      MOVB 4(R1),R4      :MOVE BR TO PORT4
5919 031726 116104 000004      CMPB R5,R4      :PUT 'EXPECTED' IN R5
5920 031732 120504      BEQ 4$      :PUT 'FOUND' IN R4
5921 031734 001401      HLT 1$      :DATA CORRECT?
5922 031736 104015      4$: SCOP1      :BR IF YES
5923 031740 104401      INC R2      :ALU ERROR
5924 031742 005202      INC R0      :SW09-1?
5925 031744 005200      CMP #10,R0      :NEXT DATA
5926 031746 022700 000010      BNE 1$      :NEXT ADDRESS
5927 031752 001342      SCOPE      :DONE YET?
5928 031754 104400      5$: .BYTE 0,1,-1,0,0,253,125,0 :BR IF NO
5929 031756 000 001 377 :SCOPE THIS TEST
5930 031761 000 000 353
5931 031764 125 000
5932      .EVEN
5933
5934
5935      : ***** TEST 126 *****
5936      : *ALU TEST
5937      : *TEST OF ALU FUNCTION ADD W/C WITH C BIT SET
5938      : *ALU FUNCTION (A PLUS B PLUS C) CODE=01
5939      : *LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5940      : *PERFORM THE FUNCTION, VERIFY THE RESULTS
5941      : *****
5942
5943      : TEST 126
5944      : -----
5945 031766 012737 000126 001226 TST126: MOV #126,TSTNO

```

5946	031774	012737	032142	001216		MOV	#TST127,NEXT	
5947	032002	012737	032034	001220		MOV	#1\$,LOCK	
5948								:R1 CONTAINS BASE DMC11 ADDRESS
5949	032010	104412				MSTCLR		:MASTER CLEAR DMC11
5950	032012	005000				CLR	R0	:MEM + SP ADDRESS
5951	032014	012702	032132			MOV	#5\$,R2	:POINTER TO CORRECT DATA
5952	032020	004737	034676			JSR	PC,MEMLD	:LOAD 8 WORDS OF MAIN MEMORY
5953	032024	035022				MEMDAT		:POINTER TO DATA
5954	032026	004737	034732			JSR	PC,SPLD	:LOAD 8 WORDS OF SP
5955	032032	035032				SPDAT		:POINTER TO DATA
5956	032034	004737	035010		1\$:	JSR	PC,SETC	:SET C BIT!
5957	032040	042737	000017	032054		BIC	#17,2\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
5958	032046	050037	032054			BIS	R0,2\$	:ADD ADDRESS TO INSTRUCTION
5959	032052	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5960	032054	010000			2\$:	010000		:LOAD MAR
5961	032056	042737	000017	032072		BIC	#17,3\$	:CLEAR ADDRESS OF INSTRUCTION
5962	032064	050037	032072			BIS	R0,3\$	:ADD ADDRESS TO INSTRUCTION
5963	032070	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5964	032072	040420			3\$:	040400.<01+20>		:BR ADD W/C
5965	032074	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5966	032076	061224				61224		:MOVE BR TO PORT4
5967	032100	111205				MOV	(R2),R5	:PUT 'EXPECTED' IN R5
5968	032102	116104	000004			MOV	4(R1),R4	:PUT 'FOUND' IN R4
5969	032106	120504				CMPS	R5,R4	:DATA CORRECT?
5970	032110	001401				BEQ	4\$	:BR IF YES
5971	032112	104015				HLT	1\$	:ALU ERROR
5972	032114	104401			4\$:	SCOPE1		:SW09=1?
5973	032116	005202				INC	R2	:NEXT DATA
5974	032120	005200				INC	R0	:NEXT ADDRESS
5975	032122	022700	000010			CMP	#10,R0	:DONE YET?
5976	032126	001342				BNE	1\$	:BR IF NO
5977	032130	104400				SCOPE		:SCOPE THIS TEST
5978	032132	001	000	000	5\$:	.BYTE	1,0,0,-1,255,0,0,125	
5979	032135	377	253	000				
5980	032140	000	125					

.EVEN

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

: TEST 127

5994	032142	012737	000127	001226	TST127:	MOV	#127,TSTNO	
5995	032150	012737	032316	001216		MOV	#TST130,NEXT	
5996	032156	012737	032210	001220		MOV	#1\$,LOCK	
5997								:R1 CONTAINS BASE DMC11 ADDRESS
5998	032164	104412				MSTCLR		:MASTER CLEAR DMC11
5999	032166	005000				CLR	R0	:MEM + SP ADDRESS
6000	032170	012702	032306			MOV	#7\$,R2	:POINTER TO CORRECT DATA
6001	032174	004737	034676			JSR	PC,MEMLD	:LOAD 8 WORDS OF MAIN MEMORY

6002	032200	035022				MEMDAT		: POINTER TO DATA
6003	032202	004737	034732			JSR	PC, SPLD	: LOAD 8 WORDS OF SP
6004	032206	035032				SPDAT		: POINTER TO DATA
6005	032210	004737	035010		1\$:	JSR	PC, SETC	: SET C BIT!
6006	032214	042737	000017	032230		BIC	#1, 2\$	: CLEAR ADDRESS FIELD OF INSTRUCTION
6007	032222	050037	032230			BIS	R0, 2\$	: ADD ADDRESS TO INSTRUCTION
6008	032226	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
6009	032230	010000			2\$:	010000		: LOAD MAR
6010	032232	042737	000017	032246		BIC	#17, 3\$	: CLEAR ADDRESS OF INSTRUCTION
6011	032240	050037	032246			BIS	R0, 3\$	: ADD ADDRESS TO INSTRUCTION
6012	032244	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6013	032246	040440			3\$:	040440!<2*20>		: BR SUB W/C
6014	032250	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6015	032252	061224				61224		: MOVE BR TO PORT4
6016	032254	111205				MOVB	(R2), R5	: PUT 'EXPECTED' IN R5
6017	032256	116104	000004			MOVB	4(R1), R4	: PUT 'FOUND' IN R4
6018	032262	120504				CMPB	R5, R4	: DATA CORRECT?
6019	032264	001401				BEQ	4\$	: BR IF YES
6020	032266	104015				HLT	1\$	: ALU ERROR
6021	032270	104401			4\$:	SCOP1		: SW09-1?
6022	032272	005202				INC	R2	: NEXT DATA
6023	032274	005200				INC	R0	: NEXT ADDRESS
6024	032276	022700	000010			CMP	#10, R0	: DONE YET?
6025	032302	001342				BNE	1\$	: BR IF NO
6026	032304	104400				SCOPE		: SCOPE THIS TEST
6027	032306	000	001	377	5\$:	.BYTE	0, 1, -1, 0, 0, 253, 125, 0	
6028	032311	000	000	253				
6029	032314	125	000					

.EVEN

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

: TEST 130

6043	032316	012737	000130	001226	TST'30:	MOV	#130, TSTNO	
6044	032324	012737	032472	001216		MOV	#TST131, NEXT	
6045	032332	012737	032364	001220		MOV	#1\$, LOCK	
6046								: R1 CONTAINS BASE DMC11 ADDRESS
6047	032340	104412				MSTCLR		: MASTER CLEAR DMC11
6048	032342	005000				CLR	R0	: MEM + SP ADDRESS
6049	032344	012702	032462			MOV	#5\$, R2	: POINTER TO CORRECT DATA
6050	032350	004737	034676			JSR	PC, MEMLD	: LOAD 8 WORDS OF MAIN MEMORY
6051	032354	035022				MEMDAT		: POINTER TO DATA
6052	032356	004737	034732			JSR	PC, SPLD	: LOAD 8 WORDS OF SP
6053	032362	035032				SPDAT		: POINTER TO DATA
6054	032364	004737	035010		1\$:	JSR	PC, SETC	: SET C BIT!
6055	032370	042737	000017	032404		BIC	#1\$, 2\$	: CLEAR ADDRESS FIELD OF INSTRUCTION
6056	032376	050037	032404			BIS	R0, 2\$	: ADD ADDRESS TO INSTRUCTION
6057	032402	104414				ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC 5304

6058	032404	010000			2\$:	010000	:LOAD MAR
6059	032406	042737	000017	032422		BIC #17,3\$	:CLEAR ADDRESS OF INSTRUCTION
6060	032414	050037	032422			BIS R0,3\$	:ADD ADDRESS TO INSTRUCTION
6061	032420	104414				ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6062	032422	040460			3\$:	040400.<3*20>	:BR INC A
6063	032424	104414				ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6064	032426	061224				61224	:MOVE BR TO PORT4
6065	032430	111205				MOVB (R2),R5	:PUT 'EXPECTED' IN R5
6066	032432	116104	000004			MOVB 4(R1),R4	:PUT 'FOUND' IN R4
6067	032436	120504				CMPB R5,R4	:DATA CORRECT?
6068	032440	001401				BEQ 4\$	:BR IF YES
6069	032442	104015				HLT 15	:ALU ERROR
6070	032444	104401			4\$:	SCOP1	:SW09=1?
6071	032446	005202				INC R2	:NEXT DATA
6072	032450	005200				INC R0	:NEXT ADDRESS
6073	032452	022700	000010			CMP #10,R0	:DONE YET?
6074	032456	001342				BNE 1\$	:BR IF NO
6075	032460	104400				SCOPE	:SCOPE THIS TEST
6076	032462	001	001	000	5\$:	.BYTE 1,1,0,0,126,126,253,253	
6077	032465	000	126	126			
6078	032470	253	253				
6079						.EVEN	

6080
 6081
 6082 :***** TEST 131 *****
 6083 :*ALU TEST
 6084 :*TEST OF ALU FUNCTION 2A WITH C BIT SET
 6085 :*ALU FUNCTION (A PLUS A) CODE=5
 6086 :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
 6087 :*PERFORM THE FUNCTION, VERIFY THE RESULTS
 6088 :*****
 6089

6090 : TEST 131
 6091 :-----
 6092 032472 012737 000131 001226 TST131: MOV #131,TSTNO
 6093 032500 012737 032646 001216 MOV #TST132,NEXT
 6094 032506 012737 032540 001220 MOV #1\$,LOCK
 6095 :R1 CONTAINS BASE DMC11 ADDRESS
 6096 032514 104412 MSTCLR :MASTER CLEAR DMC11
 6097 032516 005000 CLR R0 :MEM + SP ADDRESS
 6098 032520 012702 032636 MOV #5\$,R2 :POINTER TO CORRECT DATA
 6099 032524 004737 034676 JSR PC,MEMLD :LOAD 8 WORDS OF MAIN MEMORY
 6100 032530 035022 MEMDAT :POINTER TO DATA
 6101 032532 004737 034732 JSR PC,SPLD :LOAD 8 WORDS OF SP
 6102 032536 035032 SPDAT :POINTER TO DATA
 6103 032540 004737 035010 JSR PC,SETC :SET C BIT!
 6104 032544 042737 000017 032560 BIC #17,2\$:CLEAR ADDRESS FIELD OF INSTRUCTION
 6105 032552 050037 032560 BIS R0,2\$:ADD ADDRESS TO INSTRUCTION
 6106 032556 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 6107 032560 010000 2\$: 010000 :LOAD MAR
 6108 032562 042737 000017 032576 BIC #17,3\$:CLEAR ADDRESS OF INSTRUCTION
 6109 032570 050037 032576 BIS R0,3\$:ADD ADDRESS TO INSTRUCTION
 6110 032574 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 6111 032576 040520 3\$: 040400.<5*2> :BR 2A
 6112 032600 104414 ROMCLK :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 6113 032602 061224 61224 :MOVE BR TO PORT4

```

6114 032604 111205          MOVB      (R2),R5          :PUT 'EXPECTED' IN R5
6115 032606 116104 000004    MOVB      4(R1),R4        :PUT 'FOUND' IN R4
6116 032612 120504          CMPB      R5,R4          :DATA CORRECT?
6117 032614 001401          BEQ       4$              :BR IF YES
6118 032616 104015          HLT       1$              :ALU ERROR
6119 032620 104401          4$: SCOP1      :SW09=1?
6120 032622 005202          INC       R2              :NEXT DATA
6121 032624 005200          INC       R0              :NEXT ADDRESS
6122 032626 022700 000010    CMP       #10,R0         :DONE YET?
6123 032632 001342          BNE      1$              :BR IF NC
6124 032634 104400          SCOPE     :SCOPE THIS TEST
6125 032636      000      000      376 5$: .BYTE 0,0,376,376,252,252,124,124
6126 032641      376      252      252
6127 032644      124      124
6128                                     .EVEN
6129
6130
6131                                     :***** TEST 132 *****
6132                                     :*ALU TEST
6133                                     :*TEST OF ALU FUNCTION A PLUS C WITH C BIT SET
6134                                     :*ALU FUNCTION (A PLUS C) CODE 4
6135                                     :*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
6136                                     :*PERFORM THE FUNCTION, VERIFY THE RESULTS
6137                                     :*****
6138
6139                                     : TEST 132
6140                                     :-----
6141 032646 012737 000132 001226 TST132: MOV      #132,TSTNO
6142 032654 012737 033022 001216 MOV      #TST133,NEXT
6143 032662 012737 032714 001220 MOV      #1$,LOCK
6144
6145 032670 104412          MSTCLR          :R1 CONTAINS BASE DMC11 ADDRESS
6146 032672 005000          CLR       R0            :MASTER CLEAR DMC11
6147 032674 012702 033012    MOV      #5$,R2        :MEM + SP ADDRESS
6148 032700 004737 034676    JSR      PC,MEMLD      :POINTER TO CORRECT DATA
6149 032704 035022          MEMDAT          :LOAD 8 WORDS OF MAIN MEMORY
6150 032706 004737 034732    JSR      PC,SPLD      :POINTER TO DATA
6151 032712 035032          SPDAT          :LOAD 8 WORDS OF SP
6152 032714 004737 035010    JSR      PC,SETC      :POINTER TO DATA
6153 032720 042737 000017 032734 1$: BIC      #17,2$ :SET C BIT
6154 032726 050037 032734    BIS      R0,2$        :CLEAR ADDRESS FIELD OF INSTRUCTION
6155 032732 104414          ROMCLK          :ADD ADDRESS TO INSTRUCTION
6156 032734 010000          010000          :NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
6157 032736 042737 000017 032752 2$: BIC      #17,3$ :LOAD MAR
6158 032744 050037 032752    BIS      R0,3$        :CLEAR ADDRESS OF INSTRUCTION
6159 032750 104414          ROMCLK          :ADD ADDRESS TO INSTRUCTION
6160 032752 040500          040400!<4*20> :NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
6161 032754 104414          ROMCLK          :BR A PLUS C
6162 032756 061224          61224          :NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
6163 032760 111205          MOVB      (R2),R5        :MOVE BR TO PORT4
6164 032762 116104 000004    MOVB      4(R1),R4      :PUT 'EXPECTED' IN R5
6165 032766 120504          CMPB      R5,R4          :PUT 'FOUND' IN R4
6166 032770 001401          BEQ       4$              :DATA CORRECT?
6167 032772 104015          HLT       1$              :BR IF YES
6168 032774 104401          4$: SCOP1      :ALU ERROR
6169 032776 005202          INC       R2              :SW09-1?
                                     :NEXT DATA

```



```

6170 033000 005200          INC      R0          ;NEXT ADDRESS
6171 033002 022700 000010    CMP      #10,R0      ;DONE YET?
6172 033006 001342          BNE      1$          ;BR IF NO
6173 033010 104400          SCOPE          ;SCOPE THIS TEST
6174 033012      001      001      000 5$: .BYTE 1,1,0,0,126,126,253,253
6175 033015      000      126      126
6176 033020      253      253
6177                                     .EVEN
6178
6179
6180                                     ;***** TEST 133 *****
6181                                     ;*ALU TEST
6182                                     ;*TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT SET
6183                                     ;*ALU FUNCTION (A-B-1) CODE-17
6184                                     ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
6185                                     ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
6186                                     ;*****
6187
6188                                     ; TEST 133
6189                                     ;-----
6190 033022 012737 000133 001226 TST133: MOV      #133,TSTND
6191 033030 012737 033176 001216    MOV      #TST134,NEXT
6192 033036 012737 033070 001220    MOV      #1$,LOCK
6193
6194 033044 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
6195 033046 005000          CLR      R0      ;MASTER CLEAR DMC11
6196 033050 012702 033166    MOV      #5$,R2    ;MEM + SP ADDRESS
6197 033054 004737 034676    JSR      PC,MEMLD  ;POINTER TO CORRECT DATA
6198 033060 035022          MEMDAT        ;LOAD 8 WORDS OF MAIN MEMORY
6199 033062 004737 034732    JSR      PC,SPLD   ;POINTER TO DATA
6200 033066 035032          SPDAT        ;LOAD 8 WORDS OF SP
6201 033070 004737 035010 1$: JSR      PC,SETC  ;POINTER TO DATA
6202 033074 042737 000017 033110    BIC      #17,2$ ;SET C BIT!
6203 033102 050037 033110    BIS      R0,2$    ;CLEAR ADDRESS FIELD OF INSTRUCTION
6204 033106 104414          ROMCLK        ;ADD ADDRESS TO INSTRUCTION
6205 033110 010000          010000        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6206 033112 042737 000017 033126 2$: BIC      #17,3$ ;LOAD MAR
6207 033120 050037 033126    BIS      R0,3$    ;CLEAR ADDRESS OF INSTRUCTION
6208 033124 104414          ROMCLK        ;ADD ADDRESS TO INSTRUCTION
6209 033126 040760          040400 <17*20> 3$: ROMCLK 61224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
6210 033130 104414          ROMCLK 61224    ;BR 2'S COMP SUB
6211 033132 061224          MOVB      (R2),R5  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6212 033134 111205          MOVB      4(R1),R4 ;MOVE BR TO PORT4
6213 033136 116104 000004    CMPB      R5,R4  ;PUT 'EXPECTED' IN R5
6214 033142 120504          BEQ      4$      ;PUT 'FOUND' IN R4
6215 033144 001401          HLT      15      ;DATA CORRECT?
6216 033146 104015          SCOPE1        ;BR IF YES
6217 033150 104401          INC      R2      ;ALU ERROR
6218 033152 005202          INC      R0      ;SW09=1?
6219 033154 005200          CMP      #10,R0  ;NEXT DATA
6220 033156 022700 000010    BNE      1$      ;NEXT ADDRESS
6221 033162 001342          SCOPE          ;DONE YET?
6222 033164 104400          .BYTE -1,0,376,-1,-1,252,124,-1 ;BR IF NO
6223 033166      377      000      376 5$:
6224 033171      377      377      252
6225 033174      124      377

```

6226 .EVEN

6227
6228
6229
6230
6231
6232
6233
6234
6235
6236
6237
6238
6239
6240
6241
6242
6243
6244
6245
6246
6247
6248
6249
6250
6251
6252
6253
6254
6255
6256
6257
6258
6259
6260
6261
6262
6263
6264
6265
6266
6267
6268
6269
6270
6271
6272
6273
6274
6275
6276
6277
6278
6279
6280
6281

```

***** 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
-----
TST134: MOV #134,TSTNO
        MOV #TST135,NEXT
        MOV #1$,LOCK

        ;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 DEC 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

1$:     MOV #17,2$
        BIC #17,2$
        BIS R0,2$
        ROMCLK 010000
        BIC #17,3$
        BIS R0,3$
        ROMCLK 040400!<7*20>
        ROMCLK 61224
        MCVB (R2),R5
        MOVB 4(R1),R4
        CMPB R5,R4
        BEQ 4$
        HLT 1$
        SCOP1
        INC R2
        INC R0
        CMP #10,R0
        BNE 1$
        SCOPE
        .BYTE -1,-1,376,376,124,124,251,251

2$:     ROMCLK 010000
        BIC #17,3$
        BIS R0,3$
        ROMCLK 040400!<7*20>
        ROMCLK 61224
        MCVB (R2),R5
        MOVB 4(R1),R4
        CMPB R5,R4
        BEQ 4$
        HLT 1$
        SCOP1
        INC R2
        INC R0
        CMP #10,R0
        BNE 1$
        SCOPE
        .BYTE -1,-1,376,376,124,124,251,251

3$:     ROMCLK 040400!<7*20>
        ROMCLK 61224
        MCVB (R2),R5
        MOVB 4(R1),R4
        CMPB R5,R4
        BEQ 4$
        HLT 1$
        SCOP1
        INC R2
        INC R0
        CMP #10,R0
        BNE 1$
        SCOPE
        .BYTE -1,-1,376,376,124,124,251,251

4$:     SCOP1
        INC R2
        INC R0
        CMP #10,R0
        BNE 1$
        SCOPE
        .BYTE -1,-1,376,376,124,124,251,251

5$:     .BYTE -1,-1,376,376,124,124,251,251

```

.EVEN

```

***** TEST 135 *****
*TEST OF PROGRAM CLOCK BIT
*DO A MASTER CLEAR, WRITE THE PROGRAM CLOCK BIT TO A ONE,
*VERIFY THAT IT CLEARS, AND THEN SETS SOME TIME LATER

```

```

6282      ;:*****
6283
6284      ; TEST 135
6285      ;-----
6286 033352 012737 000135 001226 TST135: MOV #135,TSTNO
6287 033360 012737 033544 001216 MOV #TST136,NEXT
6288
6289 033366 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
6290 033370 005037 001416 CLR TEMP ;MASTER CLEAR DMC11
6291 033374 005037 001246 CLR TEMP1 ;PREPARE FOR
6292 033400 012702 000011 MOV #11,R2 ;DELAY
6293 033404 012761 000020 000004 1$: MOV #20,4(R1) ;SAVE FOR TYPEOUT
6294 033412 152761 000002 000001 BISB #BIT1,1(R1) ;LOAD PORT 4
6295 033420 012761 121111 000006 MOV #121111,6(R1) ;SET ROMI
6296 033426 152761 000003 000001 BISB #BIT1!BIT0,1(R1) ;SEL6 INSTRUCTION
6297 033434 012761 121224 000006 MOV #121224,6(R1) ;SET CLOCK BIT
6298 033442 152761 000003 000001 BISB #BIT1.BIT0,1(R1) ;LOAD NEXT INSTRUCTION
6299 033450 142761 000003 000001 BICB #BIT1!BIT0,1(R1) ;READ CLOCK BIT
6300 033456 016104 000004 MOV 4(R1),R4 ;CLEAR MAINT BITS
6301 033462 005005 CLR R5 ;PUT 'FOUND' IN R4
6302 033464 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
6303 033466 001401 BEQ 2$ ;IS PGM CLOCK CLEAR?
6304 033470 104016 HLT 16 ;ERROR, PGM CLOCK IS NOT CLEAR
6305 033472 2$:
6306 033472 1044 4 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
6307 033474 121224 121224 ;PORT4 LUI1
6308 033476 122761 000020 000004 CMPB #20,4(R1) ;IS PGM CLOCK SET?
6309 033504 001416 BEQ 3$ ;BR IF YES
6310 033506 062737 000001 001416 ADD #1,TEMP ;INCREMENT DELAY
6311 033514 005537 001246 ADC TEMP1 ;INCREMENT DELAY
6312 033520 022737 000006 001246 CMP #6,TEMP1 ;IS DELAY DONE
6313 033526 001361 BNE 2$ ;BR IF NO
6314 033530 012705 000020 MOV #20,R5 ;PUT 'EXPECTED' IN R5
6315 033534 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
6316 033540 104016 HLT 16 ;ERROR PGM CLOCK NOT SET
6317 033542 104400 3$: SCOPE ;SCOPE THIS TEST
6318
6319
6320      ;***** TEST 136 *****
6321      ;*FORCE POWER FAIL TEST
6322      ;*SET FORCE POWER FAIL BIT VERIFY THAT PROCESSOR TRAPS TO 24
6323      ;*GOING DOWN AND COMING UP. VERIFY ALSO THAT BUS INIT WAS
6324      ;*BLOCKED FROM GETTING TO THE DMC DURING THE POWER FAIL
6325      ;*THIS TEST MAY HANG ON SOME PROCESSORS IF AN M9301 IS PRESENT.
6326      ;*TO AVOID HANGING SW02 (POWER ON REBOOT ENABLE) ON THE M9301
6327      ;*MUST BE IN THE OFF POSITION. THIS TEST WILL ALSO FAIL IF THE
6328      ;*CPU POWER FAIL VECTOR IS SET TO ANY LOCATION OTHER THAN 24.
6329      ;*IF THIS TEST HANGS OR FAILS DUE TO EITHER REASON ABOVE THE
6330      ;*FOLLOWING PATCH MAY BE INSTALLED TO SKIP THIS TEST:
6331      ;*
6332      ;* LOC 33362 WAS 33544 SB 33736
6333      ;:*****
6334
6335      ; TEST 136
6336      ;-----
6337 033544 012737 000136 001226 TST136: MOV #136,TSTNO

```

```

6338 033552 012737 033736 001216      MOV      #TST137,NEXT
6339                                     ;R1 CONTAINS BASE DMC11 ADDRESS
6340 033560 104412      MSTCLR     ;MASTER CLEAR DMC11
6341 033562 005037 001416      CLR      TEMP      ;PREPARE FOR DELAY
6342 033566 013746 000024      MOV      @#24,-(SP)    ;STORE POWER FAIL ADDRESS
6343 033572 012737 033636 000024      MOV      #1$,@#24      ;SET UP FOR FORCE POWER FAIL
6344 033600 012761 000002 000004      MOV      #2,4(R1)    ;LOAD PORT4
6345 033606 012711 001000      MOV      #BIT9,(R1)    ;SET ROMI
6346 033612 012761 121111 000006      MOV      #121111,6(R1) ;LOAD INSTRUCTION
6347 033620 012711 001400      MOV      #BIT9.BIT8,(R1) ;CLOCK INSTRUCTION
6348 033624 005237 001416      5$:      INC      TEMP      ;WAIT FOR POWER FAIL
6349 033630 001375      BNE      5$      ;BR IF DELAY NOT DONE
6350 033632 104017      HLT      17      ;ERROR, NO POWER FAIL
6351 033634 000426      BR       4$
6352 033636 012737 033654 000024 1$:      MOV      #3$,@#24      ;POWER UP ADDRESS
6353 033644 010637 033652      MOV      SP,2$      ;STORE STACK
6354 033650 000000      HALT
6355 033652 000000      2$:      0      ;WAIT FOR POWER UP SEQUENCE
6356 033654 013706 033652      3$:      MOV      2$,SP      ;RESTORE STACK
6357 033660 022626      POP2SP     ;POP STACK TWICE
6358 033662 012637 000024      MOV      (SP)+,@#24    ;RESTORE TRUE POWER FAIL ADDRESS
6359 033666 022737 005336 000024      CMP      #.PFAIL,@#24    ;IS IT CORRECT?
6360 033674 001406      BEQ      4$      ;BR IF YES
6361 033676 104017      HLT      17      ;ERROR, STACK IS INCORRECT
6362 033700 012737 005336 000024      MOV      #.PFAIL,@#24    ;RESTORE TRUE POWER FAIL ADDRESS
6363 033706 012706 001200      MOV      #STACK,SP    ;RESTORE STACK
6364 033712 012711 003000      4$:      MOV      #BIT9.BIT10,(R1) ;SEL6 = MAINT IR
6365 033716 012705 121111      MOV      #121111,R5    ;R5 = EXPECTED
6366 033722 016104 000004      MOV      4(R1),R4    ;R4 = FOUND
6367 033726 020504      CMP      R5,R4      ;MAINT IR SHOULD 12111
6368 033730 001401      BEQ      +4      ;BR IF OK
6369 033732 104025      HLT      25      ;IF - 0 THEN BUS INIT WAS
6370                                     ;NOT BLOCKED FROM CLEARING
6371                                     ;THE DMC-11
6372 033734 104400      SCOPE      ;SCOPE THIS TEST
6373
6374
6375      ;***** TEST 137 *****
6376      ;*MICRO-PROCESSOR NOISE TEST
6377      ;*WRITE ALL ZERO'S THEN ALL ONE'S THEN A DATA PATTERN
6378      ;*TO THE IBUS* AND IBUS REGISTERS AND TO THE SP AND MAIN MEM
6379      ;*THEN GO BACK AND READ THE DATA PATTERNS TO VERIFY THAT
6380      ;*READING AND WRITING OF OTHER LOCATIONS AND REGISTERS
6381      ;*DID NOT CHANGE THE DATA.
6382      ;*****
6383
6384      ; TEST 137
6385      ;-----
6386 033736 012737 000137 001226 TST137: MOV      #137,TSTNO
6387 033744 012737 003364 001216      MOV      #.EOP,NEXT
6388
6389 033752 104412      MSTCLR     ;R1 CONTAINS BASE DMC11 ADDRESS
6390 033754 005002      CLR      R2      ;MASTER CLEAR DMC11
6391 033756 042737 000017 034002 1$:      BIC      #7,2$      ;R2 IS INDEX REGISTER
6392 033764 156237 034576 034002      BISR      30$(R2),2$    ;CLEAR ADDRESS FIELD
6393 033772 116261 034604 000004      MOV      31$(R2),4(R1) ;ADD IBUS* REG ADDRESS TO INSTRUCTION
                                     ;LOAD PORT4

```

6394	034000	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
6395	034002	121100			2\$:	121100		:WRITE IBUS* REGISTER
6396	034004	005202				INC	R2	:INC INDEX REGISTER
6397	034006	022702	000005			CMP	#5,R2	:DONE YET?
6398	034012	001361				BNE	1\$	:BR IF NO
6399	034014	005002				CLR	R2	:R2 IS IBUS REGISTER ADDRESS
6400	034016	042737	000017	034062	3\$:	BIC	#17,4\$	:CLEAR ADDRESS FIELD OF INSTRUCTIONS
6401	034024	042737	000017	034074		BIC	#17,5\$	
6402	034032	042737	000017	034104		BIC	#17,6\$	
6403	034040	050237	034062			BIS	R2,4\$	:ADD IBUS REG ADDRESS TO INSTRUCTION
6404	034044	050237	034074			BIS	R2,5\$	
6405	034050	050237	034104			BIS	R2,6\$	
6406	034054	105061	000004			CLRB	4(R1)	:CLEAR PORT4
6407	034060	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6408	034062	122100			4\$:	122100		:WRITE 0 TO IBUS REG
6409	034064	112761	000377	000004		MOVB	#377,4(R1)	:LOAD PORT4
6410	034072	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6411	034074	122100			5\$:	122100		:WRITE ALL ONES TO IBUS REG
6412	034076	110261	000004			MOVB	R2,4(R1)	:LOAD PORT4
6413	034102	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6414	034104	122100			6\$:	122100		:WRITE ITS OWN ADDRESS TO IBUS REG
6415	034106	005202				INC	R2	:NEXT ADDRESS
6416	034110	022702	000010			CMP	#10,R2	:DONE YET?
6417	034114	001340				BNE	3\$	:BR IF NO
6418	034116	005002				CLR	R2	:START AT SP ADDRESS 0
6419	034120	042737	000017	034164	7\$:	BIC	#17,8\$	:CLEAR ADDRESS FIELD
6420	034126	042737	000017	034176		BIC	#17,9\$	
6421	034134	042737	000017	034206		BIC	#17,10\$	
6422	034142	050237	034164			BIS	R2,8\$	:ADD ADDRESS TO INSTRUCTION
6423	034146	050237	034176			BIS	R2,9\$	
6424	034152	050237	034206			BIS	R2,10\$	
6425	034156	105061	000004			CLRB	4(R1)	:CLEAR PORT4
6426	034162	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6427	034164	123100			8\$:	123100		:WRITE ZERO TO SP
6428	034166	112761	000377	000004		MOVB	#377,4(R1)	:LOAD PORT4
6429	034174	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6430	034176	123100			9\$:	123100		:WRITE ALL ONES TO SP
6431	034200	110261	000004			MOVB	R2,4(R1)	:LOAD PORT4
6432	034204	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6433	034206	123100			10\$:	123100		:WRITE SP ADDRESS TO ITSELF
6434	034210	005202				INC	R2	:NEXT SP ADDRESS
6435	034212	022702	000020			CMP	#20,R2	:DONE YET?
6436	034216	001340				BNE	7\$	:BR IF NO
6437	034220	005002				CLR	R2	:R2 = MAIN MEM ADDRESS
6438	034222	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6439	034224	010000				010000		:MAR 0
6440	034226	105061	000004		11\$:	CLRB	4(R1)	:CLEAR PORT4
6441	034232	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6442	034234	122500				122500		:WRITE ZEROS TO MEM
6443	034236	112761	000377	000004		MOVB	#377,4(R1)	:LOAD PORT4
6444	034244	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6445	034246	122500				122500		:WRITE ONES TO MEM
6446	034250	110261	000004			MOVB		

6450	034262	022702	000400		CMP	#400,R2		:DONE YET?
6451	034266	001357			BNE	11\$		:BR IF NO
6452								
6453								:NOW GO BACK AND READ EVERYTHING
6454								
6455	034270	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6456	034272	010000			010000			:MAR 0
6457	034274	032737	100000	001366	BIT	#BIT15,STAT1		:DMC?
6458	034302	001402			BEQ	.+6		:BR IF YES
6459	034304	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6460	034306	004000			4000			:MAR HI 0 (KMC ONLY)
6461	034310	005000			CLR	R0		:R0 IS INDEX REGISTER
6462	034312	042737	000360	034350	12\$:	BIC	#360,13\$	:CLEAR ADDRESS FIELD
6463	034320	116002	034576		MOVB	30\$(R0),R2		:R2 = IBUS* ADDRESS
6464	034324	010203			MOV	R2,R3		:PUT IBUS* ADDRESS IN R3
6465	034326	006303			ASL	R3		:SHIFT ADDRESS TO BITS 4-7
6466	034330	006303			ASL	R3		
6467	034332	006303			ASL	R3		
6468	034334	006303			ASL	R3		
6469	034336	050337	034350		BIS	R3,13\$		:ADD ADDRESS TO INSTRUCTION
6470	034342	116005	034604		MOVB	31\$(R0),R5		:R5 = 'EXPECTED'
6471	034346	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
6472	034350	121004		13\$:	121004			:PORT4 IBUS* REGISTER
6473	034352	016104	000004		MOV	4(R1),R4		:R4 = 'FOUND'
6474	034356	120504			CMPB	R5,R4		:IBUS* CONTENTS OK?
6475	034360	001401			BEQ	.+4		:BR IF YES
6476	034362	104004			HLT	4		:IBUS* DATA ERROR
6477	034364	005200			INC	R0		:INC COUNTER
6478	034366	022700	000005		CMP	#5,R0		:DONE YET?
6479	034372	001347			BNE	12\$		:BR IF NO
6480	034374	005002			CLR	R2		:R2 = IBUS REG ADDRESS
6481	034376	042737	000360	034426	14\$:	BIC	#360,15\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
6482	034404	010203			MOV	R2,R3		:R3 = IBUS ADDRESS
6483	034406	006303			ASL	R3		:SHIFT ADDRESS TO BITS 4-7
6484	034410	006303			ASL	R3		
6485	034412	006303			ASL	R3		
6486	034414	006303			ASL	R3		
6487	034416	050337	034426		BIS	R3,15\$		:ADD ADDRESS TO INSTRUCTION
6488	034422	010205			MOV	R2,R5		:R5 = 'EXPECTED'
6489	034424	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6490	034426	021004		15\$:	021004			:PORT4 IBUS REG
6491	034430	016104	000004		MOV	4(R1),R4		:R4 = 'FOUND'
6492	034434	120504			CMPB	R5,R4		:IBUS CONTENTS OK?
6493	034436	001401			BEQ	.+4		:BR IF YES
6494	034440	104005			HLT	5		:IBUS DATA ERROR
6495	034442	005202			INC	R2		:NEXT IBUS REGISTER
6496	034444	022702	000010		CMP	#10,R2		:DONE YET?
6497	034450	001352			BNE	14\$		:BR IF NO
6498	034452	005002			CLR	R2		:R2 = SP ADDRESS
6499	034454	042737	000017	034470	16\$:	BIC	#17,17\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
6500	034462	050237	034470		BIS	R2,17\$		:ADD ADDRESS TO INSTRUCTION
6501	034466	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6502	034470	040600		17\$:	040600			:BR SP
6503	034472	010205			MOV	R2,R5		:R5 = 'EXPECTED'
6504	034474	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6505	034476	061224			061224			:PORT4 BR

6506	034500	016104	000004		MOV	4(R1),R4	:R4 - 'FOUND'
6507	034504	120504			CMPB	R5,R4	:SP CONTENTS OK?
6508	034506	001401			BEQ	.+4	:BR IF YES
6509	034510	104007			HLT	7	:SP DATA ERROR
6510	034512	005202			INC	R2	:NEXT SP LOCATION
6511	034514	022702	000020		CMP	#20,R2	:DONE YET?
6512	034520	001355			BNE	16\$	:BR IF NO
6513	034522	005002			CLR	R2	:R2 - MEMORY ADDRESS
6514	034524	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
6515	034526	010000			010000		:MAR - 0
6516	034530	032737	100000	001366	BIT	#BIT15,STAT1	:DMC?
6517	034536	001402			BEQ	.+6	:BR IF YES
6518	034540	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
6519	034542	004000			4000		:MAR HI - 0 (KMC ONLY)
6520	034544	010205			MOV	R2,R5	:R5 - 'EXPECTED'
6521	034546	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
6522	034550	055224			055224		:PORT4 - MAIN MEM
6523	034552	016104	000004		MOV	4(R1),R4	:R4 = 'FOUND'
6524	034556	120504			CMPB	R5,R4	:MAIN MEM CONTENTS OK?
6525	034560	001401			BEQ	.+4	:BR IF YES
6526	034562	104013			HLT	13	:MAIN MEM DATA ERROR
6527	034564	005202			INC	R2	:NEXT MEM ADDRESS
6528	034566	022702	000400		CMP	#400,R2	:DONE YET?
6529	034572	001364			BNE	18\$	:BR IF NO
6530	034574	104400			SCOPE		:SCOPE THIS TEST
6531	034576	000	002	007	.BYTE	0,2,3,5,10	
6532	034601	005	010				
6533		034604			.EVEN		
6534	034604	001	003	004	.BYTE	1,3,4,6,10	
6535	034607	006	010				
6536		034612			.EVEN		
6537							
6538							
6539					:SUBROUTINES		
6540					:-----		
6541							
6542	034612				SETVEC:		
6543							
6544						:THIS SUBROUTINE LOADS THE VECTORS AND VECTOR LEVELS	
6545	034612	012577	144556		MOV	(R5)+,ADMVLC	:LOAD BASE VECTOR
6546	034616	012577	144556		MOV	(R5)+,ADMTVEC	:LOAD VECTOR + 2
6547	034622	112577	144550		MOVB	(R5)+,ADMRLVL	:LOAD VECTOR + 4
6548	034626	112577	144550		MOVB	(R5)+,ADMTLVL	:LOAD VECTOR + 6
6549	034632	000205			RTS	R5	:RETURN
6550							
6551							
6552	034634				NPRSET:		
6553						:THIS SUBROUTINE LOADS IBUS REGISTERS 0-7	
6554						:WITH NPR INFORMATION (INBA, OUTBA, OUT DATA)	
6555							
6556	034634	010246			MOV	R2,-(SP)	:SAVE R2
6557	034636	005002			CLR	R2	:START AT IBUS REG 0
6558	034640	112561	000004		MOVB	(R5)+,4(R1)	:LOAD PORT4
6559	034644	042737	000017		BIT	#17,2\$	:CLEAR ADDRESS FIELD OF INSTRUCTION
6560	034652	050237	034660		BIT	#18,2\$	:ADD ADDRESS TO INSTRUCTION
6561	034656	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC-5304

```

6562 034660 122100      2$: 122100      ;MOVE PORT4 TO IBUS REG
6563 034662 005202      INC R2          ;NEXT ADDRESS
6564 034664 022702 000010 CMP #10,R2      ;ALL DONE?
6565 034670 001363      BNE 1$          ;BR IF NO
6566 034672 012602      MOV (SP)+,R2    ;RESTORE R2
6567 034674 000205      RTS R5          ;RETURN
6568
6569
6570 034676      MEMLD:
6571      ;THIS SUBROUTINE LOADS THE FIRST 8 LOCATIONS OF MAIN
6572      ;MEMORY WITH THIS DATA: 0,-1,0,-1,125,252,125,252
6573
6574 034676 013605      MOV @ (SP)+,R5    ;PUT POINTER TO DATA IN R5
6575 034700 062746 000001 ADD #2,-(SP)    ;ADJUST STACK
6576 034704 012704 000010 MOV #10,R4     ;DO 8 LOADS
6577 034710 104414      ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6578 034712 010000 010000 ;MAR < 0
6579 034714 112577 144414 1$: MOVB (R5)+,@DMP04 ;LOAD PORT4
6580 034720 104414      ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6581 034722 136500 136500 ;MOV DATA TO MEM, AUTO INC MAR
6582 034724 005304      DEC R4          ;DECREMENT COUNT
6583 034726 001372      BNE 1$          ;BR IF NOT DONE
6584 034730 000207      RTS PC          ;RETURN
6585
6586
6587 034732      SPLD:
6588      ;THIS SUBROUTINE LOADS THE FIRST 8 SCRATCH PAD
6589      ;LOCATIONS WITH: 0,0,-1,-1,125,125,252,252
6590
6591 034732 013605      MOV @ (SP)+,R5    ;PUT POINTER TO DATA IN R5
6592 034734 062746 000002 ADD #2,-(SP)    ;ADJUST STACK
6593 034740 005004      CLR R4          ;START AT SP ADDRESS 0
6594 034742 112577 144444 1$: MOVB (R5)+,@DMP04 ;LOAD PORT4 WITH DATA
6595 034744 042737 000017 034762 BIC #17,2$    ;CLEAR ADDRESS FIELD OF INSTRUCTION
6596 034754 050437 034762 BIS R4,2$      ;ADD ADDRESS TO INSTRUCTION
6597 034760 104414      ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6598 034762 123100 123100 2$: MOV R4,SP    ;MOVE DATA TO SP
6599 034764 005204      INC R4          ;INCREMENT COUNT
6600 034766 022704 000010 CMP #10,R4     ;DONE YET?
6601 034772 001363      BNE 1$          ;BR IF NO
6602 034774 000207      RTS PC          ;RETURN
6603
6604
6605 034776      CLR:
6606      ;THIS SUBROUTINE CLEARS THE MICRO PROCESSOR C BIT
6607
6608 034776 104414      ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6609 035000 010000 010000 ;MAR 0
6610 035002 104414      ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6611 035004 040400 040400 ;CLEAR C BIT
6612 035006 000207      RTS PC          ;RETURN
6613
6614
6615 035010      SET:
6616      ;THIS SUBROUTINE SETS THE MICRO PROCESSOR C BIT
6617

```


CZDMC MACY11 30A(1052) 24-AUG-78 12:19 PAGE 124
 CZDMCC.P11 11-AUG-78 08:21 DMC11 ALU TESTS

SEQ 0124

6618	035010	104414	ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
6619	035012	010003	010003		:MAR 3
6620	035014	104414	ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
6621	035016	040403	040403	<0*20>	:SET C BIT
6622	035020	000207	RTS	PC	:RETURN

6623					
6624					
6625	035022	000	377	000	MEMDAT: .BYTE 0,-1,0,-1,125,252,125,252
6626	035025	377	125	252	
6627	035030	125	252		
6628	035032	000	000	377	SPDAT: .BYTE 0,0,-1,-1,125,125,252,252
6629	035035	377	125	125	
6630	035040	252	252		

6631					.EVEN
6632	035042	052777	044516	052502	EM1: .ASCIZ <377>/UNIBUS REGISTER ADDRESSING TIME-OUT/
	035107	377	047125	041111	EM2: .ASCIZ <377>/UNIBUS REGISTER WRITE/READ TEST^
	035150	046777	041511	047522	EM3: .ASCIZ <377>/MICRO PROCESSOR TEST/
	035176	046777	041511	047522	EM4: .ASCIZ <377>/MICRO PROCESSOR WRITE/READ TEST^
	035237	377	051102	051040	EM5: .ASCIZ <377>/BR REGISTER TEST/
	035261	377	041523	040522	EM6: .ASCIZ <377>/SCRATCH PAD TEST/
	035303	377	042504	044526	EM7: .ASCIZ <377>/DEVICE FAILED TO INTERRUPT/
	035337	377	042504	044526	EM10: .ASCIZ <377>/DEVICE INTERRUPTED TO WRONG VECTOR/
	035403	377	050116	020122	EM11: .ASCIZ <377>/NPR TEST/
	035415	377	040515	047111	EM'2: .ASCIZ <377>/MAIN MEMORY TEST/
	035437	377	040515	020122	EM'3: .ASCIZ <377>/MAR TEST/
	035451	377	046101	020125	EM'4: .ASCIZ <377>/ALU TEST/
	035463	377	051120	043517	EM15: .ASCIZ <377>/PROGRAM CLOCK TEST/
	035507	377	047506	041522	EM16: .ASCIZ <377>/FORCE POWER FAIL ERROR/
	035537	377	047125	054105	EM17: .ASCIZ <377>/UNEXPECTED INTERRUPT/
	035565	377	046504	030503	EM20: .ASCIZ <377>/DMC11 CONFIGURATION ERROR/
	035620	046777	044501	052116	EM21: .ASCIZ <377>/MAINTENANCE INSTRUCTION REGISTER TEST/
	035667	377	047520	042527	EM22: .ASCIZ <377>/POWER FAIL INITIALIZE FAILURE/
	035726	051377	043505	051511	DH1: .ASCIZ <377>/REGISTER
	035767	377	054105	042520	DH2: .ASCIZ <377>/EXPECTED FOUND TRAPPED FROM/
	036025	377	054105	042520	DH3: .ASCIZ <377>/EXPECTED FOUND REGISTER/
	036046	042777	050130	041505	DH4: .ASCIZ <377>/EXPECTED FOUND IBUS* REGISTER/
	036110	042777	050130	041505	DH5: .ASCIZ <377>/EXPECTED FOUND IBUS REGISTER/
	036151	377	054105	042520	DH6: .ASCIZ <377>/EXPECTED FOUND ADDRESS/

036206	000002				
036210	006	005			
036212	001262				
036214	006	002			
036216	001264				
036220	000003				
036222	006	004			
036224	001272				
036226	006	004			
036230	001270				
036232	006	002			
036234	001262				
036236	000002				
036240	006	004			
036242	001272				

036244	006	002		.BYTE	6.2	
036246	001270			SAVR4		
036250	000003		DT4:	3		
036252	003	007		.BYTE	3.7	
036254	001272			SAVR5		
036256	003	011		.BYTE	3.11	
036260	001270			SAVR4		
036262	002	002		.BYTE	2.2	
036264	001264			SAVR2		
036266	000003		DT5:	3		
036270	003	007		.BYTE	3.7	
036272	001272			SAVR5		
036274	003	007		.BYTE	3.7	
036276	001270			SAVR4		
036300	006	002		.BYTE	6.2	
036302	001264			SAVR2		
036304	000002		DT6:	2		
036306	003	007		.BYTE	3.7	
036310	001272			SAVR5		
036312	003	002		.BYTE	3.2	
036314	001270			SAVR4		
036316	000002		DT7:	2		
036320	006	004		.BYTE	6.4	
036322	001262			SAVR1		
036324	006	002		.BYTE	6.2	
036326	001404			DMCSR		
036330			.ERRTAB:			
036330	000000			0		
036332	000000			0		
036334	000000			0		
036336	035042			EM1		
036340	035726			DH1	:HLT	1
036342	036206			DT1		
036344	035107			EM2		
036346	035767			DH2	:HLT	2
036350	036220			DT2		
036352	035150			EM3		
036354	036025			DH3	:HLT	3
036356	036236			DT3		
036360	035176			EM4		
036362	036046			DH4	:HLT	4
036364	036250			DT4		
036366	035176			EM4		
036370	036110			DH5	:HLT	5
036372	036250			DT4		
036374	035237			EM5		
036376	036025			DH3	:HLT	6
036400	036304			DT6		
036402	035261			EM6		
036404	036151			DH6	:HLT	7
036406	036266			DT5		
036410	035303			EM7		
036412	000000			0	:HLT	10
036414	000000			0		
036416	035337			EM10		
036420	000000			0	:HLT	11

036422	000000	0		
036424	035403	EM11		
036426	036025	DH3	:HLT	12
036430	036236	DT3		
036432	035415	EM12		
036434	036151	DH6	:HLT	13
036436	036266	DT5		
036440	035437	EM13		
036442	036151	DH6	:HLT	14
036444	036266	DT5		
036446	035451	EM14		
036450	036025	DH3	:HLT	15
036452	036304	DT6		
036454	035463	EM15		
036456	036046	DH4	:HLT	16
036460	036250	DT4		
036462	035507	EM16		
036464	000000	0	:HLT	17
036466	000000	0		
036470	035537	EM17		
036472	000000	0	:HLT	20
036474	000000	0		
036476	035403	EM11		
036500	036151	DH6	:HLT	21
036502	036266	DT5		
036504	035565	EM20		
036506	036025	DH3	:HLT	22
036510	036316	DT7		
036512	035620	EM21		
036514	036025	DH3	:HLT	23
036516	036236	DT3		
036520	035565	EM20		
036522	000000	0	:HLT	24
036524	000000	0		
036526	035667	EM22		
036530	036025	DH3	:HLT	25
036532	036236	DT3		

C36534

000001

ORMAX:
END

1907*	1908*	6548*							
1905*	1906*	1907	6546*						
1959									
865#	1070	1123	1133	1249	1877	1879	1957	1962	2192
1373*	1393#	1803*							

LSTERR	001234	751#	1077*	1319*	1621	1623*	1711*								
MASKX	001244	759#													
MASTEK	006142	1645	1774#												
MCRLF	005672	1418	1541	1641	1642	1650	1774#	1915	1977						
MCSRX	006072	1324	1774#												
MDATA	007544	1571	1581	1791#											
MEMDAT	035022	4728	4777	4826	4875	4924	4973	5022	5071	5120	5169	5218	5267	5316	
		5365	5414	5463	5512	5561	5610	5659	5708	5757	5806	5855	5904	5953	
		6002	6051	6100	6149	6198	6247	6625#							
MEMLD	034676	4684	4727	4776	4825	4874	4923	4972	5021	5070	5119	5168	5217	5266	
		5315	5364	5413	5462	5511	5560	5609	5658	5707	5756	5805	5854	5903	
		5952	6001	6050	6099	6148	6197	6246	6570#						
MEMLIM	001304	775#	1287*	4477											
MEPASS	005733	1323	1774#												
MERRPC	006217	1648	1774#												
MERRX	006117	1330	1774#												
MERR2	005760	1774#	2177												
MERR3	006005	1260	1774#												
MILK	001322	783#	1071*	1332	1875*	1580*	1884								
MLOCK	006043	1301	1774#												
MNEW	006144	1255	1774#												
MODU	006704	1774#	2040												
MPASSX	006106	1328	1774#												
MPFAIL	005675	1708	1774#												
MOM	005666	1448	1774#	1939	2014	2024	2034	2049	2063						
MR	005755	1310	1774#	1933											
MRESET	004000	683#													
MSTCLR	104412	822#	1713	2311	2352	2382	2412	2442	2472	2502	2532	2562	2592	2622	
		2652	2682	2712	2742	2772	2802	2832	2875	2919	2961	3002	3044	3070	
		3126	3182	3234	3287	3345	3405	3457	3509	3561	3613	3665	3717	3769	
		3821	3882	3933	3999	4086	4115	4153	4229	4265	4298	4337	4373	4408	
		4444	4499	4539	4582	4637	4683	4724	4773	4822	4871	4920	4969	5018	
		5067	5116	5165	5214	5263	5312	5361	5410	5459	5508	5557	5606	5655	
		5704	5753	5802	5851	5900	5949	5998	6047	6096	6145	6194	6243	6289	
		6340	6389												
MTITLE	001000	723#	1098												
MTSTN	006130	1646	1774#	1917											
MTSTPC	006031	1774#													
MVECX	006100	1326	1774#												
NEXT	001216	744#	1386	1679	2261*	2289*	2308*	2350*	2380*	2400*	2440*	2470*	2500*	2530*	
		2560*	2590*	2620*	2650*	2680*	2710*	2740*	2770*	2800*	2830*	2873*	2916*	2958*	
		2999*	3042*	3067*	3123*	3179*	3231*	3284*	3342*	3402*	3454*	3506*	3558*	3610*	
		3662*	3714*	3766*	3818*	3879*	3930*	3996*	4055*	4084*	4113*	4151*	4194*	4227*	
		4263*	4296*	4335*	4371*	4406*	4442*	4496*	4536*	4579*	4635*	4680*	4721*	4770*	
		4819*	4868*	4917*	4966*	5015*	5064*	5113*	5162*	5211*	5260*	5309*	5358*	5407*	
		5456*	5505*	5554*	5603*	5652*	5701*	5750*	5799*	5848*	5897*	5946*	5995*	6044*	
		6093*	6142*	6191*	6240*	6287*	6338*	6387*							
NDACT	007154	1110	1774#	1869											
NODEV	002674	1164	1203#												
NPRSET	034634	4199	4231	4267	4301	4340	4374	4404	4454	6552#					
NUM	006450	1774#	1969												
OK	002646	1190	1197#	1226											
ONE	001302	774#													
PACT00	001702	952#													
PACT01	001706	955#													
PACT02	001712	958#													

CZDMC MACY11 30A(1052) 24-AUG-78 12:19 PAGE 136

CZDMCC.P11 11-AUG-78 08:21

CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0135

TST12	013372	2500	2529#
TST120	030556	5603	5651#
TST121	030732	5652	5700#
TST122	031106	5701	5749#
TST123	031262	5750	5798#
TST124	031436	5799	5847#
TST125	031612	5848	5896#
TST126	031766	5897	5945#
TST127	032142	5946	5994#
TST13	013470	2530	2559#
TST130	032316	5995	6043#
TST131	032472	6044	6092#
TST132	032646	6093	6141#
TST133	033022	6142	6190#
TST134	033176	6191	6239#
TST135	033352	6240	6286#
TST136	033544	6287	6337#
TST137	033736	6338	6386#
TST14	013566	2560	2589#
TST140=	***** U	6387	
TST15	013664	2590	2619#
TST16	013762	2620	2649#
TST17	014060	2650	2679#
TST2	012426	2261	2288#
TST20	014156	2680	2709#
TST21	014254	2710	2739#
TST22	014352	2740	2769#
TST23	014450	2770	2799#
TST24	014546	2800	2829#
TST25	014672	2830	2872#
TST26	015016	2873	2915#
TST27	015146	2916	2957#
TST3	012456	2289	2307#
TST30	015306	2958	2998#
TST31	015450	2999	3041#
TST32	015534	3042	3066#
TST33	015734	3067	3122#
TST34	016134	3123	3178#
TST35	016310	3179	3230#
TST36	016464	3231	3283#
TST37	016664	3284	3341#
TST4	012606	2308	2349#
TST40	017104	3342	3401#
TST41	017260	3402	3453#
TST42	017434	3454	3505#
TST43	017610	3506	3557#
TST44	017764	3558	3609#
TST45	020140	3610	3661#
TST46	020314	3662	3713#
TST47	020470	3714	3765#
TST5	012704	2350	2379#
TST50	020644	3766	3817#
TST51	021072	3818	3878#
TST52	021242	3879	3929#
TST53	021510	3930	3995#
TST54	021736	3996	4054#

CROSS REFERENCE TABLE -- USER SYMBOLS

SENDAD

4187#	4189#	4217#	4220	4222#	4253#	4256	4258#	4285#	4288	4291#	4324#	4327
4330#	4360#	4363	4366#	4395#	4398	4401#	4430#	4433	4436#	4486#	4489	4491#
4526#	4529	4531#	4568#	4571	4574#	4624#	4627	4630#	4670#	4673	4675#	4708#
4711	4716#	4757#	4760	4765#	4806#	4809	4814#	4855#	4858	4863#	4904#	4907
4712#	4953#	4956	4961#	5002#	5005	5010#	5051#	5054	5059#	5100#	5103	5108#
5149#	5152	5157#	5198#	5201	5206#	5247#	5250	5255#	5296#	5299	5304#	5345#
5348	5353#	5394#	5397	5402#	5443#	5446	5451#	5492#	5495	5500#	5541#	5544
5549#	5590#	5593	5598#	5639#	5642	5647#	5688#	5691	5696#	5737#	5740	5745#
5786#	5789	5794#	5835#	5838	5843#	5884#	5887	5892#	5933#	5936	5941#	5982#
5985	5990#	6031#	6034	6039#	6080#	6083	6088#	6129#	6132	6137#	6178#	6181
6186#	6227#	6230	6235#	6276#	6279	6282#	6318#	6321	6333#	6373#	6376	6382#
710	1096	1342#	1666									
588#	2251	2256	2258	2264#	2280	2284	2286	2291#	2297	2303	2305	2311
2312#	2339	2345	2347	2352	2353#	2369	2375	2377	2382	2383#	2399	2405
2407	2412	2413#	2429	2435	2437	2442	2443#	2459	2465	2467	2472	2473#
2489	2495	2497	2502	2503#	2519	2525	2527	2532	2533#	2549	2555	2557
2562	2563#	2579	2585	2587	2592	2593#	2609	2615	2617	2622	2623#	2639
2645	2647	2652	2653#	2669	2675	2677	2682	2683#	2699	2705	2707	2712
2713#	2729	2735	2737	2742	2743#	2759	2765	2767	2772	2773#	2789	2795
2797	2802	2803#	2810	2825	2827	2832	2833#	2862	2868	2870	2875	2876#
2905	2911	2913	2919	2920#	2947	2953	2955	2961	2962#	2988	2994	2996
3002	3003#	3029	3037	3039	3044	3045#	3056	3062	3064	3070	3071#	3112
3118	3120	3126	3127#	3168	3174	3176	3182	3183#	3220	3226	3228	3234
3235#	3272	3279	3281	3287	3288#	3329	3337	3339	3345	3346#	3391	3397
3399	3405	3406#	3443	3449	3451	3457	3458#	3495	3501	3503	3509	3510#
3547	3553	3555	3561	3562#	3599	3605	3607	3613	3614#	3651	3657	3659
3665	3666#	3703	3709	3711	3717	3718#	3755	3761	3763	3769	3770#	3807
3813	3815	3821	3822#	3868	3874	3876	3882	3883#	3919	3925	3927	3933
3934#	3985	3991	3993	3999	4000#	4045	4050	4052	4057	4059#	4074	4079
4081	4086	4087#	4102	4108	4110	4115	4116#	4140	4146	4148	4153	4154#
4184	4189	4191	4196	4198#	4217	4222	4224	4229	4230#	4253	4258	4260
4265	4266#	4285	4291	4293	4298	4299#	4324	4330	4332	4337	4338#	4360
4366	4368	4373	4374#	4395	4401	4403	4408	4409#	4430	4436	4438	4444</

\$RAMCL	588#	1718													
\$RCLK	588#	1721	1724	1761	1766	3076	3078	3096	3098	3132	3134	3152	3154	3187	3189
	3205	3207	3239	3241	3257	3259	3293	3295	3313	3315	3351	3353	3373	3375	3410
	3412	3428	3430	3462	3464	3480	3482	3514	3516	3532	3534	3566	3568	3584	3586
	3618	3620	3636	3638	3670	3672	3688	3690	3722	3724	3740	3742	3774	3776	3792
	3794	3828	3836	3854	3886	3888	3904	3906	3939	3943	3945	3962	3966	3968	4006
	4010	4012	4030	4032	4065	4093	4129	4169	4206	4238	4242	4244	4274	4307	4315
	4317	4351	4353	4380	4383	4386	4415	4418	4421	4461	4504	4507	4509	4511	4545
	4548	4550	4552	4586	4589	4591	4593	4608	4610	4612	4639	4643	4646	4652	4656
	4659	4689	4691	4693	4695	4734	4738	4740	4783	4787	4789	4832	4836	4838	4881
	4885	4887	4930	4934	4936	4979	4983	4985	5028	5032	5034	5077	5081	5083	5126
	5130	5132	5175	5179	5181	5224	5228	5230	5273	5277	5279	5322	5326	5328	5371
	5375	5377	5420	5424	5426	5469	5473	5475	5518	5522	5524	5567	5571	5573	5616
	5620	5622	5665	5669	5671	5714	5718	5720	5763	5767	5769	5812	5816	5818	5861
	5865	5867	5910	5914	5916	5959	5963	5965	6008	6012	6014	6057	6061	6063	6106
	6110	6112	6155	6159	6161	6204	6208	6210	6253	6257	6259	6306	6394	6407	6410
	6413	6426	6429	6432	6438	6441	6444	6447	6455	6459	6471	6489	6501	6504	6514
	6518	6521	6561	6577	6580	6597	6608	6610	6618	6620					
\$SCOPE	588#	1362													
\$SIMBC	588#														
\$SOFTC	588#	1794													
\$SPF	588#	3935	3957												
\$SP1	588#	3919													
\$SP2	588#	3985													
\$TIMER	588#	6276													
\$TRPDE	588#	802	804	806	808	810	812	814	816	818	820	822	824	826	828
	830														
\$TSTN	588#	2258	2286	2305	2347	2377	2407	2437	2467	2497	2527	2557	2587	2617	2647
	2677	2707	2737	2767	2797	2827	2870	2913	2955	2996	3039	3064	3120	3176	3228
	3281	3339	3399	3451	3503	3555	3607	3659	3711	3763	3815	3876	3927	3993	4052
	4081	4110	4148	4191	4224	4260	4293	4332	4368	4403	4438	4493	4533	4576	4632
	4677	4718	4767	4816	4865	4914	4963	5012	5061	5110	5159	5208	5257	5306	5355
	5404	5453	5502	5551	5600	5649	5698	5747	5796	5845	5894	5943	5992	6041	6090
	6139	6188	6237	6284	6335	6384									
\$VARIA	588#	721													
\$WRFLT	588#	2834	2847	2877	2890										
\$WR46	588#	2819	2862												
\$XZ	588#	2251	2256	2280	2284	2297	2303	2339	2345	2369	2375	2399	2405	2429	2435
	2459	2465	2489	2495	2519	2525	2549	2555	2579	2585	2609	2615	2639	2645	2669
	2675	2699	2705	2729	2735	2759	2765	2789	2795	2819	2825	2862	2868	2905	2911
	2947	2953	2988	2994	3029	3037	3056	3062	3112	3118	3168	3174	3220	3226	3272
	3279	3329	3337	3391	3397	3443	3449	3495	3501	3547	3553	3599	3605	3651	3657
	3703	3709	3755	3761	3807	3813	3868	3874	3919	3925	3985	3991	4045	4050	4074
	4079	4102	4108	4140	4146	4184	4189	4217	4222	4253	4258	4285	4291	4324	4330
	4360	4366	4395	4401	4430	4436	4486	4491	4526	4531	4568	4574	4624	4630	4670
	4675	4708	4716	4757	4765	4806	4814	4855	4863	4904	4912	4953	4961	5002	5010
	5051	5059	5100	5108	5149	5157	5198	5206	5247	5255	5296	5304	5345	5353	5394
	5402	5443	5451	5492	5500	5541	5549	5590	5598	5639	5647	5688	5696	5737	5745
	5786	5794	5835	5843	5884	5892	5933	5941	5982	5990	6031	6037	6080	6088	6129
	6137	6178	6186	6227	6235	6276	6282	6318	6333	6373	6382				

. ABS. C36534 000

ERRORS DETECTED: C

L 11

CZDMC MACY11 30A(1052) 24-AUG-78 12:19 PAGE 143

CZDMCC.P11 11-AUG-78 08:21

CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0141

CZDMCC/I,CZDMCC.SEO/SOL/CRF=CZDMCC.P11

RUN-TIME: 16 23 1 SECONDS

RUN-TIME RATIO: 55/41-1.5

CORE USED: 28K (55 PAGES)