

DMC-11

MODE LINE UNIT TEST
CZDMECO

AH-8553C-MC

COPYRIGHT 76-78

FICHE 1 OF 1

JAN 1979

digital

MADE IN USA

.REM 8

IDENTIFICATION

PRODUCT CODE: AC-8552C-MC
PRODUCT NAME: CZDMECO DDCMP MD LN UNIT 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
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43

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.

CZDME TESTS THE DMC-11 LINE UNIT (M8201 OR M8202). IT PERFORMS WRITE/READ TESTS ON THE DMC LINE UNIT REGISTERS. IT CHECKS FOR PROPER TRANSMITTER, RECEIVER, AND BCC OPERATION IN DDCMP MODE. THE MODEM SIGNALS ARE ALSO CHECKED. CZDME REQUIRES A DMC MICRO-PROCESSOR (M8200 OR M8204) TO RUN. FOR BEST DIAGNOSIS A TURN-AROUND CONNECTOR SHOULD BE INSTALLED, HOWEVER THE DIAGNOSTIC WILL RUN WITHOUT IT (SOME TESTS ARE SKIPPED).

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 WITH DMC11-DA OR DMC11-FA OR
DMC11-AL WITH DMC11-MA OR DMC11-MD

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 CROM? (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

190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218

ANSWERED.

4.1 CONTROL SWITCH SETTINGS

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

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

4.1.2 SWITCH REGISTER OPTIONS (AT START UP)

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

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

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

METHOD: A: LOAD ADDRESS 200
B: START WITH SW 00=1
C: PROGRAM WILL TYPE MESSA
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 SET UP 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.

LINE UNIT(M8201)- JUMPERS W1, W2, AND W4 MUST BE IN. JUMPERS W3, AND W5 MUST BE OUT. SW8 OF E26 MUST BE IN THE ON POSITION.

LINE UNIT (M8202)- JUMPER W1 MUST BE IN. SW8 OF E26 MUST BE IN THE OFF POSITION.

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:.

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.

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

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

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

DMCSR (1402) 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

470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498

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

CSR: CONTAINS DMC11 CSR ADDRESS

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

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

STAT3: BIT0=1 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 HAS BEEN FOUND, IF NOT, THE ADDRESS IS UPDATED BY 10 AND THE SEARCH CONTINUES. A -1 INDICATES A DMC11 WITH NO CROM OR CROM, A 125252 INDICATES A KMC11 WITH CROM, A 626 INDICATES A DMC11-AL AND A 16520 INDICATES A DMC11-AR. FURTHER TESTS ARE PERFORMED AT THIS POINT TO DETERMINE WHICH LINE UNIT, IF ANY, IS INSTALLED, IF A LOOP-BACK CONNECTOR IS INSTALLED AND VARIOUS SWITCH SETTINGS ON THE LINE UNIT. THIS IS WHY THE STATUS TABLE MUST BE VERIFIED BY THE USER AND IF ANY OF THE INFORMATION DOES NOT AGREE WITH THE HARDWARE THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS MUST BE ANSWERED. ALL DMC11'S IN THE SYSTEM WILL BE FOUND BY THE AUTO-SIZER. IF IT DOES NOT FIND A DMC11 THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS ANSWERED.

8.5.2 FINDING THE VECTOR AND BR LEVEL

THE VECTOR AREA (ADDRESS 300-776) IS FILLED WITH THE INSTRUCTION IOT AND '+2' (NEXT ADDRESS). THE PROCESSOR STATUS IS STARTED AT 7 AND THE DMC IS PROGRAMMED TO INTERRUPT. THE PS IS LOWERED BY 1 UNTIL THE DMC INTERRUPTS, A DELAY IS MADE AND IF NO INTERRUPT OCCURES AT PS LEVEL 3 (BECAUSE OF A BAD DMC11) THE PROGRAM ASSUMES VECTOR ADDRESS 300 AT BR LEVEL 5 AND THE PROBLEM SHOULD BE FIXED IN THE DIAGNOSTIC. ONCE THE PROBLEM IS FIXED; THE PROGRAM SHOULD BE RE-SETUP AGAIN TO GET CORRECT VECTOR. IF AN INTERRUPT OCCURED; THE ADDRESS TO WHICH THE DMC11 INTERRUPTED TO IS PICKED UP AND REPORTED AS THE VECTOR. NOTE: IF THE VECTOR REPORTED IS NOT THE VECTOR SET UP BY YOU; THERE IS A PROBLEM AND AUTO SIZING SHOULD NOT BE DONE.

8.6 SOFTWARE SWITCH REGISTER

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

CONTROL:

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

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
588
589
590
591
592
593
594

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.

```

595
596
597
598
599
600      ;*AC-8552C-MC   DMC11 DDCMP MODE LINE UNIT TESTS
601      ;*COPYRIGHT 1976,1978, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
602      ;-----
603
604      ;STARTING PROCEDURE
605      ;LOAD PROGRAM
606      ;LOAD ADDRESS 000200
607      ;SWR=0  AUTOSIZE DMC11
608      ;SW07=1  USE CURRENT DMC11 PARAMETERS
609      ;SW00=1  INPUT NEW DMC11 PARAMETERS
610      ;PRESS START
611      ;PROGRAM WILL TYPE 'AC-8552C-MC   DMC11 DDCMP MODE LINE UNIT TESTS'
612      ;PROGRAM WILL TYPE STATUS MAP
613      ;PROGRAM WILL TYPE 'R' TO INDICATE THAT TESTING HAS STARTED
614      ;AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
615      ;AND THEN RESUME TESTING
616      ;SUBSEQUENT RESTARTS WILL NOT TYPE PROGRAM TITLE
617
618
619
620
621      ;SWITCH REGISTER OPTIONS
622      ;-----
623
624      100000      SW15=100000      ;=1,HALT ON ERROR
625      040000      SW14=40000       ;=1,LOOP ON CURRENT TEST
626      020000      SW13=20000      ;=1,INHIBIT ERROR TYPEOUT
627      010000      SW12=10000      ;=1,DELETE TYPEOUT/BELL ON ERROR.
628      004000      SW11=4000       ;=1,INHIBIT ITERATIONS
629      002000      SW10=2000       ;=1,ESCAPE TO NEXT TEST ON ERROR
630      001000      SW09=1000       ;=1,LOOP WITH CURRENT DATA
631      000400      SW08=400        ;=1,LOOP ON ERROR
632      000200      SW07=200        ;=1,USE CURRENT DMC11 PARAMETERS, =0,AUTOSIZE DMC11
633      000100      SW06=100        ;=1, HALT BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION
634      000040      SW05=40
635      000020      SW04=20
636      000010      SW03=10
637      000004      SW02=4
638      000002      SW01=2
639      000001      SW00 1
                    ;RESELECT DMC11'S TO BE TESTED (ACTIVE)
                    ;LOCK ON TEST SELECT
                    ;RESTART PROGRAM AT SELECTED TEST
                    ;INPUT DMC11 PARAMETERS

```

```

640
641
642      ;REGISTER DEFINITIONS
643      ;-----
644
645      000000      R0=%0      ;GENERAL REGISTER
646      000001      R1=%1      ;GENERAL REGISTER
647      000002      R2=%2      ;GENERAL REGISTER
648      000003      R3=%3      ;GENERAL REGISTER
649      000004      R4=%4      ;GENERAL REGISTER
650      000005      R5=%5      ;GENERAL REGISTER
651      000006      SP=%6      ;PROCESSOR STACK POINTER
652      000007      PC=%7      ;PROGRAM COUNTER
653
654      ;LOCATION EQUIVALENCIES
655      ;-----
656
657      177776      PS=177776    ;PROCESSOR STATUS WORD
658      001200      STACK=1200  ;START OF PROCESSOR STACK
659
660      ;INSTRUCTION DEFINITIONS
661      ;-----
662
663      005746      PUSH1SP=5746 ;DECREMENT PROCESSOR STACK 1 WORD
664      005726      POP1SP=5726  ;INCREMENT PROCESSOR STACK 1 WORD
665      010046      PUSHRO=10046  ;SAVE R0 ON STACK
666      012600      POPRO=12600   ;RESTORE R0 FROM STACK
667      024646      PUSH2SP=24646 ;DECREMENT STACK TWICE
668      022626      POP2SP=22626  ;INCREMENT STACK TWICE
669      .EQUIV EMT.HLT ;BASIC DEFINITION OF ERROR CALL
670
671      ;BIT DEFINITIONS
672      ;-----
673
674      100000      BIT15=100000
675      040000      BIT14=40000
676      020000      BIT13=20000
677      010000      BIT12=10000
678      004000      BIT11=4000
679      002000      BIT10=2000
680      001000      BIT9=1000
681      000400      BIT8=400
682      000200      BIT7=200
683      000100      BIT6=100
684      000040      BIT5=40
685      000020      BIT4=20
686      000010      BIT3=10
687      000004      BIT2=4
688      000002      BIT1=2
689      000001      BIT0=1
690
691
  
```

```

692
693
694
695
696
697
698
699
700
701
702      000000
703
704
705
706      000024
707      000024 005336
708      000026 000340
709      000030 004750
710      000032 000340
711      000034 004716
712      000036 000340
713
714      000040 000000
715      000042 000000
716      000044 000000
717      000046 003522
718
719      000052 000000
720
721      000174
722      000174 000000
723      000176 000000
724
725      000200
726      000200 000137 002002
727
728
729
730      001000 005377 041501 034055
731      (2) 001016 046504 030503 020061
732      (2)
733      001200
734
735
736      001200 177570
737      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-8552C-MC/<377>
        .ASCIIZ /DMC11 DDCMP MODE LINE UNIT TESTS/<377>
.=1200
;INDIRECT POINTERS TO SWITCH REGISTER AND LIGHT DISPLAY
;-----
DISPLAY:177570
SWR: 177570
  
```

```

738
739 ;INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS
740 ;-----
741
742 001204 177560 TKCSR: 177560 ;TELETYPE KEYBOARD CONTROL REGISTER
743 001206 177562 TKDBR: 177562 ;TELETYPE KEYBOARD DATA BUFFER
744 001210 177564 TPCSR: 177564 ;TELEPRINTER CONTROL REGISTER
745 001212 177566 TPDBR: 177566 ;TELEPRINTER DATA BUFFER
746
747 ;PROGRAM CONTROL PARAMETERS
748 ;-----
749
750 001214 000000 RETURN: 0 ;SCOPE ADDRESS FOR LOOP ON TEST
751 001216 000000 NEXT: 0 ;ADDRESS OF NEXT TEST TO BE EXECUTED
752 001220 000000 LOCK: 0 ;ADDRESS FOR LOCK ON CURRENT DATA
753 001222 000003 ICOUNT: 3 ;NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE EXECUTED
754 001224 000000 LPCNT: 0 ;NUMBER OF ITERATIONS COMPLETED
755 001226 000000 TSTNO: 0 ;NUMBER OF TEST IN PROGRESS
756 001230 000000 PASCNT: 0 ;NUMBER OF PASSES COMPLETED
757 001232 000000 ERRCNT: 0 ;TOTAL NUMBER OF ERRORS
758 001234 000000 LSTERR: 0 ;PC OF LAST ERROR CALL
759
760 ;PROGRAM VARIABLES
761 ;-----
762
763 001236 000000 STRTSW: 0 ;SWITCHES AT START OF PROGRAM
764 001240 000000 STAT: 0 ;DM STATUS WORD STORAGE
765 001242 000000 CLKX: 0
766 001244 000000 MASKX: 0
767 001246 000000 TEMP1: 0 ;TEMPORARY STORAGE
768 001250 000000 TEMP2: 0 ;TEMPORARY STORAGE
769 001252 000000 TEMP3: 0 ;TEMPORARY STORAGE
770 001254 000000 TEMP4: 0 ;TEMPORARY STORAGE
771 001256 000000 TEMP5: 0 ;TEMPORARY STORAGE
772 001260 000000 SAVR0: 0 ;R0 STORAGE
773 001262 000000 SAVR1: 0 ;R1 STORAGE
774 001264 000000 SAVR2: 0 ;R2 STORAGE
775 001266 000000 SAVR3: 0 ;R3 STORAGE
776 001270 000000 SAVR4: 0 ;R4 STORAGE
777 001272 000000 SAVR5: 0 ;R5 STORAGE
778 001274 000000 SAVSP: 0 ;STACK POINTER STORAGE
779 001276 000000 SAVPC: 0 ;PROGRAM COUNTER STORAGE
780 001300 000000 ZERO: 0
781 001302 000001 ONE: 1
782 001304 000000 MEMLIM: 0
783 001306 000001 DMACTV: .BLKW 1 ;HIGHEST LOCATION FOR NPR'S
784 001310 000001 DMNUM: .BLKW 1 ;DMC11'S SELECTED ACTIVE.
785 001312 000001 SAVACT: .BLKW 1 ;OCTAL NUMBER OF DMC11'S.
786 001314 000001 SAVNUM: .BLKW 1 ;ORIGINAL ACTV DEVICES
787 001316 000000 RUN: 0 ;WORKABLE NUMBER
788 ;EVEN ;POINTER TO RUNNING DEVICE.
789 001320 001472 CREAM: DM.MAP-6 ;TABLE POINTER.
790 001322 001676 MILK: CNT.MAP-4 ;TABLE POINTER

```

```

791
792      ;PROGRAM CONTROL FLAGS
793      ;-----
794
795      001324      000      INIFLG: .BYTE      0      ;PROGRAM INITIALIZATION FLAG
796      001325      000      ERRFLG: .BYTE      0      ;ERROR OCCURED FLAG
797      001326      000      LOKFLG: .BYTE      0      ;LOCK ON CURRENT TEST FLAG
798      001327      000      QV.FLG: .BYTE      0      ;QUICK VERIFY FLAG.
799                                     ;ON FIRST PASS OF EACH DMC11 ITERATIONS WILL BE SUPPRESS
800      .EVEN
801
802      ;DEFINITIONS FOR TRAP SUBROUTINE CALLS
803      ;POINTERS TO SUBROUTINES CAN BE FOUND
804      ;IN THE TABLE IMMEDIATLY FOLLOWING THE DEFINITIONS
805
806      ;:*****
807      ;-----
808      001330      .TRPTAB:
809      001330      104400      SCOPE=TRAP+0      ;CALL TO SCOPE LOOP AND ITERATION HANDLER
810      001330      003576      .SCOPE
811      001332      104401      SCOP1=TRAP+1      ;CALL TO LOOP ON CURRENT DATA HANDLER
812      001332      003736      .SCOP1
813      001334      104402      TYPE=TRAP+2      ;CALL TO TELETYPE OUTPUT ROUTINE
814      001334      003766      .TYPE
815      001336      104403      INSTR=TRAP+3      ;CALL TO ASCII STRING INPUT ROUTINE
816      001336      004050      .INSTR
817      001340      104404      INSTER=TRAP+4      ;CALL TO INPUT ERROR HANDLER
818      001340      004154      .INSTER
819      001342      104405      PARAM=TRAP+5      ;CALL TO NUMERICAL DATA INPUT ROUTINE
820      001342      004174      .PARAM
821      001344      104406      SAV05=TRAP+6      ;CALL TO REGISTER SAVE ROUTINE
822      001344      004374      .SAV05
823      001346      104407      RES05=TRAP+7      ;CALL TO REGISTER RESTORE ROUTINE
824      001346      004434      .RES05
825      001350      104410      CONVRT=TRAP+10      ;CALL TO DATA OUTPUT ROUTINE
826      001350      004466      .CONVRT
827      001352      104411      CNVRT=TRAP+11      ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
828      001352      004472      .CNVRT
829      001354      104412      MSTCLR=TRAP+12      ;CALL TO ISUE A MASTER CLEAR
830      001354      005466      .MSTCLR
831      001356      104413      DELAY=TRAP+13      ;CALL TO DELAY
832      001356      005436      .DELAY
833      001360      104414      ROMCLK=TRAP+14      ;CALL TO CLOCK ROM ONCE
834      001360      005504      .ROMCLK
835      001362      104415      DATACLK=TRAP+15      ;CALL TO CLK DATA
836      001362      005552      .DATACLK
837      001364      104416      TIMER=TRAP+16      ;CALL TO DELAY A CLOCK TICK
838      001364      005616      .TIMER
839
840      ;-----
841      ;:*****

```

```

842 ;DMC11 CONTROL INDICATORS FOR CURRENT DMC11 UNDER TEST
843 ;-----
844
845 001366 000000 STAT1: 0
846 001370 000000 STAT2: 0
847 001372 000000 STAT3: 0
848
849 ;DMC11 VECTOR AND REGISTER INDIRECT POINTERS
850 ;-----
851
852 001374 000000 DMRVEC: 0 ;POINTER TO DMC11 RECEIVER INTERRUPT VECTOR
853 001376 000000 DMRLVL: 0 ;POINTER TO DMC11 RECEIVER INTERRUPT SERVICE PS
854 001400 000000 DMTVEC: 0 ;POINTER TO DMC11 TRANSMITTER INTERRUPT VECTOR
855 001402 000000 DMTLVL: 0 ;POINTER TO DMC11 TRANSMITTER INTERRUPT SERVICE PS
856 001404 000000 DMCSR: 0 ;POINTER TO DMC11 CONTROL STATUS REGISTER
857 001406 000000 DMCSRH: 0 ;POINTER TO DMC11 CONTROL STATUS REGISTER HIGH BYTE.
858 001410 000000 DMCTL: 0 ;POINTER TO DMC11 CONTROL OUT REGISTER
859 001412 000000 DMP04: 0 ;POINTER TO DMC11 PORT REGISTER(SEL 4)
860 001414 000000 DMP06: 0 ;POINTER TO DMC11 PORT REGISTER(SEL 6)
861
862 ;TEMP STORAGE
863 ;-----
864
865 001416 000000 TEMP: 0
866 001460 .-.+40
867
868 ;DMC11 STATUS TABLE AND ADDRESS ASSIGNMENTS
869 ;-----
870
871 001500 001500 .-1500
872 001500 000001 DM.MAP:
873 001500 000001 DMC00: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 00
874 001502 000001 DMS100: .BLKW 1 ;VECTOR FOR DMC11 NUMBER 00
875 001504 000001 DMS200: .BLKW 1 ;DDCMP LINE# FOR DMC11 NUMBER 00
876 001506 000001 DMS300: .BLKW 1 ;3RD STATUS WORD
877
878 001510 000001 DMC01: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 01
879 001512 000001 DMS101: .BLKW 1 ;VECTOR FOR DMC11 NUMBER 01
880 001514 000001 DMS201: .BLKW 1 ;DDCMP LINE# FOR DMC11 NUMBER 01
881 001516 000001 DMS301: .BLKW 1 ;3RD STATUS WORD
882
883 001520 000001 DMC02: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 02
884 001522 000001 DMS102: .BLKW 1 ;VECTOR FOR DMC11 NUMBER 02
885 001524 000001 DMS202: .BLKW 1 ;DDCMP LINE# FOR DMC11 NUMBER 02
886 001526 000001 DMS302: .BLKW 1 ;3RD STATUS WORD
887
888 001530 000001 DMC03: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 03
889 001532 000001 DMS103: .BLKW 1 ;VECTOR FOR DMC11 NUMBER 03
890 001534 000001 DMS203: .BLKW 1 ;DDCMP LINE# FOR DMC11 NUMBER 03
891 001536 000001 DMS303: .BLKW 1 ;3RD STATUS WORD
892
893 001540 000001 DMC04: .BLKW 1 ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 04
894 001542 000001 DMS104: .BLKW 1 ;VECTOR FOR DMC11 NUMBER 04
895 001544 000001 DMS204: .BLKW 1 ;DDCMP LINE# FOR DMC11 NUMBER 04
896 001546 000001 DMS304: .BLKW 1 ;3RD STATUS WORD
897

```

898	001550	000001	DMCR05: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 05
899	001552	000001	DMS105: .BLKW	1	:VECTOR FOR DMC11 NUMBER 05
900	001554	000001	DMS205: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 05
901	001556	000001	DMS305: .BLKW	1	:3RD STATUS WORD
902					
903	001560	000001	DMCR06: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 06
904	001562	000001	DMS106: .BLKW	1	:VECTOR FOR DMC11 NUMBER 06
905	001564	000001	DMS206: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 06
906	001566	000001	DMS306: .BLKW	1	:3RD STATUS WORD
907					
908	001570	000001	DMCR07: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 07
909	001572	000001	DMS107: .BLKW	1	:VECTOR FOR DMC11 NUMBER 07
910	001574	000001	DMS207: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 07
911	001576	000001	DMS307: .BLKW	1	:3RD STATUS WORD
912					
913	001600	000001	DMCR10: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 10
914	001602	000001	DMS110: .BLKW	1	:VECTOR FOR DMC11 NUMBER 10
915	001604	000001	DMS210: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 10
916	001606	000001	DMS310: .BLKW	1	:3RD STATUS WORD
917					
918	001610	000001	DMCR11: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 11
919	001612	000001	DMS111: .BLKW	1	:VECTOR FOR DMC11 NUMBER 11
920	001614	000001	DMS211: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 11
921	001616	000001	DMS311: .BLKW	1	:3RD STATUS WORD
922					
923	001620	000001	DMCR12: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 12
924	001622	000001	DMS112: .BLKW	1	:VECTOR FOR DMC11 NUMBER 12
925	001624	000001	DMS212: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 12
926	001626	000001	DMS312: .BLKW	1	:3RD STATUS WORD
927					
928	001630	000001	DMCR13: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 13
929	001632	000001	DMS113: .BLKW	1	:VECTOR FOR DMC11 NUMBER 13
930	001634	000001	DMS213: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 13
931	001636	000001	DMS313: .BLKW	1	:3RD STATUS WORD
932					
933	001640	000001	DMCR14: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 14
934	001642	000001	DMS114: .BLKW	1	:VECTOR FOR DMC11 NUMBER 14
935	001644	000001	DMS214: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 14
936	001646	000001	DMS314: .BLKW	1	:3RD STATUS WORD
937					
938	001650	000001	DMCR15: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 15
939	001652	000001	DMS115: .BLKW	1	:VECTOR FOR DMC11 NUMBER 15
940	001654	000001	DMS215: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 15
941	001656	000001	DMS315: .BLKW	1	:3RD STATUS WORD
942					
943	001660	000001	DMCR16: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 16
944	001662	000001	DMS116: .BLKW	1	:VECTOR FOR DMC11 NUMBER 16
945	001664	000001	DMS216: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 16
946	001666	000001	DMS316: .BLKW	1	:3RD STATUS WORD
947					
948	001670	000001	DMCR17: .BLKW	1	:CONTROL STATUS REGISTER FOR DMC11 NUMBER 17
949	001672	000001	DMS117: .BLKW	1	:VECTOR FOR DMC11 NUMBER 17
950	001674	000001	DMS217: .BLKW	1	:DDCMP LINE# FOR DMC11 NUMBER 17
951	001676	000001	DMS317: .BLKW	1	:3RD STATUS WORD
952					
953	001700	000000	DM.END: 000000		

```

954
955      ;DMC11 PASS COUNT AND ERROR COUNT TABLE
956      ;-----
957
958      CNT.MAP:
959      PACT00: 0      ;PASS COUNT FOR DMC11 NUMBER 00
960      ERCT00: 0      ;ERROR COUNT FOR DMC11 NUMBER 00
961
962      PACT01: 0      ;PASS COUNT FOR DMC11 NUMBER 01
963      ERCT01: 0      ;ERROR COUNT FOR DMC11 NUMBER 01
964
965      PACT02: 0      ;PASS COUNT FOR DMC11 NUMBER 02
966      ERCT02: 0      ;ERROR COUNT FOR DMC11 NUMBER 02
967
968      PACT03: 0      ;PASS COUNT FOR DMC11 NUMBER 03
969      ERCT03: 0      ;ERROR COUNT FOR DMC11 NUMBER 03
970
971      PACT04: 0      ;PASS COUNT FOR DMC11 NUMBER 04
972      ERCT04: 0      ;ERROR COUNT FOR DMC11 NUMBER 04
973
974      PACT05: 0      ;PASS COUNT FOR DMC11 NUMBER 05
975      ERCT05: 0      ;ERROR COUNT FOR DMC11 NUMBER 05
976
977      PACT06: 0      ;PASS COUNT FOR DMC11 NUMBER 06
978      ERCT06: 0      ;ERROR COUNT FOR DMC11 NUMBER 06
979
980      PACT07: 0      ;PASS COUNT FOR DMC11 NUMBER 07
981      ERCT07: 0      ;ERROR COUNT FOR DMC11 NUMBER 07
982
983      PACT10: 0      ;PASS COUNT FOR DMC11 NUMBER 10
984      ERCT10: 0      ;ERROR COUNT FOR DMC11 NUMBER 10
985
986      PACT11: 0      ;PASS COUNT FOR DMC11 NUMBER 11
987      ERCT11: 0      ;ERROR COUNT FOR DMC11 NUMBER 11
988
989      PACT12: 0      ;PASS COUNT FOR DMC11 NUMBER 12
990      ERCT12: 0      ;ERROR COUNT FOR DMC11 NUMBER 12
991
992      PACT13: 0      ;PASS COUNT FOR DMC11 NUMBER 13
993      ERCT13: 0      ;ERROR COUNT FOR DMC11 NUMBER 13
994
995      PACT14: 0      ;PASS COUNT FOR DMC11 NUMBER 14
996      ERCT14: 0      ;ERROR COUNT FOR DMC11 NUMBER 14
997
998      PACT15: 0      ;PASS COUNT FOR DMC11 NUMBER 15
999      ERCT15: 0      ;ERROR COUNT FOR DMC11 NUMBER 15
1000
1001      PACT16: 0      ;PASS COUNT FOR DMC11 NUMBER 16
1002      ERCT16: 0      ;ERROR COUNT FOR DMC11 NUMBER 16
1003
1004      PACT17: 0      ;PASS COUNT FOR DMC11 NUMBER 17
1005      ERCT17: 0      ;ERROR COUNT FOR DMC11 NUMBER 17
1006
  
```

1007

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	STAT2
I	*	B	M		A	D	D	*	I	*	L	I	N	E	#	
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

- CSR: CONTAINS DMC11 CSR ADDRESS
- STAT1: BITS 00-08 IS DMC11 VECTOR ADDRESS
BIT15=1 MICRO-PROCESSOR HAS CRAM
BIT15=0 MICRO-PROCESSOR HAS CROM
BIT14=1 ??? TURNAROUND CONNECTOR IS ON
BIT14=0 NO TURNAROUND CONNECTOR
BIT13=0 LINE UNIT IS AN M8201
BIT13=1 LINE UNIT IS AN M8202
BIT12=1 NO LINE UNIT
BITS 09-11 IS DMC11 BR PRIORITY LEVEL
- STAT2: LOW BYTE IS SWITCH PAC#1 (DDCMP LINE NUMBER)
HIGH BYTE IS SWITCH PAC#2 (BM873 BOOT ADD)
- STAT3: BIT0=1 DO FREE RUNNING TESTS ON KMC
(MUST BE SET TO A ONE MANUALLY [PROGRAM DZDMI ONLY])
KMC MUST HAVE MICRO-CODE WRITTEN FROM RUNNING
DZDMG TEST 2 FIRST
BIT1=1 DMC11-AL LOCAL HIGH SPEED MICRO-CODE
BIT1=0 DMC11-AR REMOTE LOW SPEED MICRO-CODE

```

1062
1063
1064      ;PROGRAM INITIALIZATION
1065      ;LOCK OUT INTERRUPTS
1066      ;SET UP PROCESSOR STACK
1067      ;SET UP POWER FAIL VECTOR
1068      ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
1069      ;TYPE TITLE MESSAGE
1070 002002 012737 000340 177776 .START: MOV #340,PS      ;LOCK OUT INTERRUPTS
1071 002010 012706 001200      MOV #STACK,SP      ;SET UP STACK
1072 002014 012737 005336 000024      MOV #.PFAIL,@#24      ;SET UP POWER FAIL VECTOR
1073 002022 013737 001310 001314      MOV DMNUM,SAVNUM      ;SAVE NUMBER OF DEVICES IN SYSTEM.
1074 002030 005037 010016      CLR SWFLG      ;CLEAR SOFT TYPEOUT FLAG
1075 002034 105037 001325      CLRB ERRFLG      ;CLEAR ERROR FLAG
1076 002040 105037 001327      CLRB QV.FLG      ;ZERO QUICK VERIFY FLAG
1077 002044 012737 001470 001320      MOV #DM.MAP-10,CREAM      ;GET MAP POINTER.
1078 002052 012737 001676 001322      MOV #CNT.MAP-4,MILK      ;GET PASS COUNT MAP POINTER
1079 002060 012737 100000 001316      MOV #BIT15,RUN      ;POINT POINTER TO FIRST DEVICE.
1080 002066 012700 001702      MOV #CNT.MAP,R0      ;PASS COUNT POINTER TO R0
1081 002072 005020      23$: CLR (R0)+      ;CLEAR TABLE
1082 002074 022700 002002      CMP #CNT.MAP+100,R0      ;DONE YET?
1083 002100 001374      BNE 23$      ;KEEP GOING
1084 002102 005037 001234      CLR LSTERR      ;CLEAR LAST ERROR POINTER
1085 002106 012737 000001 001226      MOV #1,TSTNO      ;SET UP FOR TEST 1
1086 002114 012737 002002 001214      MOV #.START,RETURN      ;SET UP FOR POWER FAIL BEFORE
1087      ;TESTING STARTS
1088      ;SAVE CURRENT VECTORS
1089 002122 013746 000006      MOV @#6,-(SP)
1090 002126 013746 000004      MOV @#4,-(SP)
1091 002132 012737 002166 000004      MOV #6$,@#4      ;SET UP FOR TIMEOUT
1092 002140 012737 177570 001202      MOV #177570,SWR      ;SET SWR TO HARD SWR ADDRESS
1093 002146 012737 177570 001200      MOV #177570,DISPLAY      ;SET DISPLAY TO HARD SWR ADDRESS
1094 002154 022777 177777 177020      CMP #-1,@SWR      ;REFERENCE HARDWARE SWITCH REGISTER
1095 002162 001402      BEQ 6$+2      ;IF = -1 USE SOFT SWR ANYWAY
1096 002164 000407      BR 7$      ;IF IT EXISTS AND NOT = -1 USE HARD SWR
1097 002166 022625      6$: CMP (SP)+,(SP)+      ;ADJUST STACK
1098 002170 012737 000176 001202      MOV #SWREG,SWR      ;POINTER TO SOFT SWR
1099 002176 012737 000174 001200      MOV #DISPREG,DISPLAY      ;POINTER TO SOFT DISPLAY REG
1100 002204 012637 000004      7$: MOV (SP)+,@#4      ;RESTORE VECTORS
1101 002210 012637 000006      MOV (SP)+,@#6
1102 002214 105737 001324      TSTB INIFLG      ;HAS INITIALIZATION BEEN PERFORMED
1103 002220 001006      BNE 20$      ;BR IF YES
1104 002222 022737 003522 000042      CMP #SENDAD,@#42      ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
1105 002230 001402      BEQ 20$
1106 002232 104402 001000      TYPE ,MTITLE      ;TYPE TITLE MESSAGE
1107 002236 004737 007606      JSR PC,CKSWR      ;CHECK FOR SOFT SWR
1108 002242 017737 176734 001236      MOV @SWR,STRTSW      ;STORE STARTING SWITCHES
1109 002250 005737 000042      TST @#42      ;IS IT RUNNING IN AUTO MODE?
1110 002254 001402      BEQ .+6      ;BR IF NO
1111 002256 005037 001236      CLR STRTSW      ;IF YES, CLEAR SWITCHES
1112 002262 032737 000001 001236      BIT #SW00,STRTSW      ;IF SW00=1, QUESTIONS ARE ASKED.
1113 002270 001012      BNE 17$      ;BR IF SW00=1
1114 002272 105737 001236      TSTB STRTSW      ;BIT7=1??
1115 002276 100007      BPL 17$      ;BR IF SW07=0
1116 002300 005737 001306      TST DMACTV      ;ARE ANY DEVICES SELECTED?
1117 002304 001006      BNE 10$      ;BR IF YES
1118 002306 104402 007154      TYPE ,NOACT      ;NO DEVICES SELECTED.

```

```

1118 002312 000000          HALT                ;STOP THE SHOW
1119 002314 000776          BR                    ;DISQUALIFY CONTINUE SWITCH
1120 002316 004737 010512    17$: JSR            PC,AUTO.SIZE ;GO DO THE AUTO SIZE
1121 002322 105737 001324    16$: TSTB          INIFLG        ;FIRST TIME?
1122 002326 001410          BEQ            21$          ;BR IF YES
1123 002330 105737 001236    TSTB          STRTSW        ;IF USING SAME PARAMETERS DONT TYPE MAP
1124 002334 100431          BMI            1$
1125 002336 032737 000006 001236 BIT          #BIT1!BIT2,STRTSW;IS TEST NO. OR LOCK SELECTED
1126 002344 001403          BEQ            24$          ;IF NO THEN TYPE STATUS
1127 002346 000424          BR            1$            ;IF YES DO NOT TYPE STATUS
1128 002350 005137 001324    21$: COM          INIFLG        ;SET FLAG
1129 002354 104402 006224    24$: TYPE          ,XHEAD      ;TYPE HEADER
1130 002360 012704 001500    MOV          #DM.MAP,R4      ;SET POINTER
1131 002364 010437 001246    5$: MOV          R4,TEMP1     ;SET ADDRESS
1132 002370 012437 001250    MOV          (R4)+,TEMP2     ;SET CSR
1133 002374 001411          BEQ            1$            ;ALL DONE IF ZERO
1134 002376 012437 001252    MOV          (R4)+,TEMP3     ;SET STAT1
1135 002402 012437 001254    MOV          (R4)+,TEMP4     ;SET STAT2
1136 002406 012437 001256    MOV          (R4)+,TEMP5     ;SET STAT3
1137 002412 104410          CONVPT          ;TYPE OUT STATUS MAP
1138 002414 007454          XSTATQ
1139 002416 000762          BR            5$
1140 002420 012700 001500    1$: MOV          #DM.MAP,R0    ;R0 POINTS TO STATUS TABLE
1141
1142 ;*****
1143 ;*AUTO SIZE TEST
1144 ;*THIS TEST VERIFYS THAT THE DMC11S AND/OR KMC11S ARE AT THE CORRECT FLOATING
1145 ;*ADDRESSES FOR YOUR SYSTEM. IF THIS TEST FAILS, IT IS NOT A HARDWARE ERROR.
1146 ;*CHECK THE ADDRESSES OF ALL FLOATING DEVICES (DJ,DH,DG,DU,DUP,LK,DMC,DZ,KMC).
1147 ;*IF THERE ARE NO OTHER FLOATING DEVICES BEFORE THE DMC11, THE FIRST
1148 ;*DMC11 ADDRESS IS 760070, KMC11 IS 760110. NO DEVICE SHOULD EVER BE AT
1149 ;*ADDRESS 760000. THIS TEST MAY REQUIRE 2 OR MORE ATTEMPTS TO GET THE
1150 ;*RIGHT ADDRESSES. AFTER YOU HAVE CHANGED THE ADDRESS TO WHAT IT TOLD
1151 ;*YOU THE FIRST TIME, IT MAY COME BACK AND TELL YOU A DIFFERENT ADDRESS
1152 ;*THE NEXT TIME YOU RUN IT. PLEASE HAVE PATIENCE, THE FINAL ADDRESS
1153 ;*WILL BE CORRECT (AS LONG AS ALL DEVICES IN FRONT OF THE DMC'S ARE
1154 ;*CORRECT).
1155 ;*****
1156
1157 002424 013746 000004          MOV          @#4,-(SP)      ;SAVE LOC 4
1158 002430 013746 000006          MOV          @#6,-(SP)      ;SAVE LOC 6
1159 002434 005037 000006          CLR          @#6            ;CLEAR VEC+2
1160 002440 005037 001252          CLR          TEMP3         ;CLEAR FLAG
1161 002444 005005          CLR          R5                  ;R5=0=DMC, R5=-1=KMC
1162 002446 011037 001404    AUSTRT: MOV        (R0),DMCSR     ;GET NEXT DMC CSR
1163 002452 001564          BEQ          AUDONE             ;BR IF DONE
1164 002454 005705          TST          R5                 ;DMC OR KMC?
1165 002456 001005          BNE          1$                 ;BR IF KMC
1166 002460 032760 100000 000002 BIT          #BIT15,2(R0)   ;CHECK FOR DMC CSR
1167 002466 001061          FNE          SKIP              ;SKIP IF NOT DMC
1168 002470 000404          BR            2$                 ;ITS A DMC SO CONTINUE
1169 002472 032760 100000 000002 1$: BIT        #BIT15,2(R0) ;CHECK FOR KMC CSR
1170 002500 001454          BEQ          SKIP              ;SKIP IF NOT KMC
1171 002502 012737 002674 000004 2$: MOV        #NODEV,@#4   ;SET UP FOR TIMEOUT
1172 002510 005705          TST          R5                 ;DMC OR KMC?
1173 002512 001003          BNE          3$                 ;BR IF KMC

```

1174	002514	012703	000006		MOV	#6,R3	:R3 IS COUNT OF DEVICES BEFORE DMC
1175	002520	000402			BR	4\$	:GO ON
1176	002522	012703	000010	3\$:	MOV	#10,R3	:R3 IS COUNT OF DEVICES BEFORE KMC
1177	002526	012702	003010	4\$:	MOV	#DEVTAB,R2	:R2 IS DEVICE TABLE PONTER
1178	002532	012701	160010		MOV	#160010,R1	:START WITH ADDRESS 160010
1179	002536	005711		FLOAT:	TST	(R1)	:CHECK ADDRESS IN R1
1180	002540	111204			MOVB	(R2),R4	:IF NO TIMEOUT, GET NEXT ADDRESS
1181	002542	060401			ADD	R4,R1	:IN R1
1182	002544	005201			INC	R1	:
1183	002546	040401			BIC	R4,R1	:
1184	002550	005703			TST	R3	:ANY MORE DEVICES TO CHECK FOR?
1185	002552	001371			BNE	FLOAT	:BR IF YES
1186	002554	012737	002700	000004	MOV	#ERR,2#4	:OK ONLY DMC'S ARE LEFT, SET UP FOR TIMEOUT
1187	002562	010137	003022		MOV	R1,XLOC	:SAVE FIRST DMC/KMC ADDRESS
1188	002566	005705		FY:	TST	R5	:DMC OR KMC?
1189	002570	001005			BNE	1\$	:BR IF KMC
1190	002572	032760	100000	000002	BIT	#BIT15,2(R0)	:CHECK FOR DMC CSR
1191	002600	001014			BNE	SKIP	:SKIP IF NOT DMC
1192	002602	000404			BR	2\$	:ITS A DMC SO CONTINUE
1193	002604	032760	100000	000002	1\$:	BIT	#BIT15,2(R0)
1194	002612	001407			BEQ	SKIP	:CHECK FOR KMC CSR
1195	002614	005711		2\$:	TST	(R1)	:SKIP IF NOT KMC
1196	002616	020137	001404		CMP	R1,DMCSR	:CHECK DMC ADDRESS
1197	002622	001411			BEQ	OK	:DOES IT MATCH
1198	002624	062701	000010		ADD	#10,R1	:BR IF YES
1199	002630	000756			BR	FY	:GET NEXT DMC ADDRESS
1200	002632	062700	000010		ADD	#10,R0	:DO IT AGAIN
1201	002636	011037	001404	SKIP:	MOV	(R0),DMCSR	:SKIP TO NEXT CSR IN TABLE
1202	002642	001470			BEQ	AUDONE	:GET NEXT CSR
1203	002644	000750			BR	FY	:BR IF DONE
1204	002646	062700	000010		ADD	#10,R0	:ELSE CONTINUE
1205	002652	062737	000010	003022	ADD	#10,XLOC	:SKIP TO NEXT DMC CSR
1206	002660	011037	001404		MOV	(R0),DMCSR	:UPDATE EXPECTED DMC/KMC ADDRESS
1207	002664	001457			BEQ	AUDONE	:GET NEXT DMC/KMC CSR
1208	002666	013701	003022		MOV	XLOC,R1	:BR IF DONE
1209	002672	000735			BR	FY	:GET EXPECTED DMC/KMC ADDRESS
1210	002674	122243		NODEV:	CMPB	(R2)+,-(R3)	:CONTINUE
1211	002676	000002			RTI		:ON TIMEOUT, INC R2, DEC R3
1212	002700	005737	001252	ERR:	TST	TEMP3	:RETURN
1213	002704	001014			BNE	1\$	:CHECK FLAG IF = 0 TYPE HEADER
1214	002706	104402			TYPE		:SKIP HEADER
1215	002710	007223			CONERR		:TYPEOUT HEADER MESSAGE
1216	002712	012737	002700	001276	MOV	#ERR,SAVPC	:CONFIGURATION ERROR!!!!
1217	002720	104411			CONVRT		:SAVE PC FOR TYPEOUT
1218	002722	002770			ERRPC		:TYPE OUT ERROR PC
1219	002724	104402			TYPE		:
1220	002726	007277			CNERR		:TYPE REST OF HEADER
1221	002730	012737	177777	001252	MOV	#-1,TEMP3	:
1222	002736	010137	001262	1\$:	MOV	R1,SAVR1	:SET FLAG SO IT ONLY GETS TYPED ONCE
1223	002742	104410			CONVRT		:SAVE R1 FOR TYPEOUT
1224	002744	002776			CONTAB		:
1225	002746	005705			TST	R5	:TYPE CSR VALUES
1226	002750	001003			BNE	3\$	:DMC OR KMC ?
1227	002752	104402			TYPE		:BR IF KMC
1228	002754	007320			DMCM		:
1229	002756	000402			BR	4\$	:CONTINUE

PROGRAM INITIALIZATION AND START UP.

SEQ 0026

1230	002760	104402		3\$:	TYPE			
1231	002762	007330			KMCM			
1232	002764	022626		4\$:	CMP	(SP)+,(SP)+		;ADJUST STACK
1233	002766	000727			BR	OK		;BR TO GET OUT
1234	002770	000001		ERRPC:	1			
1235	002772	006	002		.BYTE	6,2		
1236	002774	001276			SAVPC			
1237	002776	000002		CONTAB:	2			
1238	003000	006	004		.BYTE	6,4		
1239	003002	003022			XLOC			
1240	003004	006	002		.BYTE	6,2		
1241	003006	001404			DMCSR			
1242	003010	007		DEVTAB:	.BYTE	7		;DJ
1243	003011	017			.BYTE	17		;DH
1244	003012	007			.BYTE	7		;DQ
1245	003013	007			.BYTE	7		;DU
1246	003014	007			.BYTE	7		;DUP
1247	003015	007			.BYTE	7		;LK
1248	003016	007			.BYTE	7		;DMC
1249	003017	007			.BYTE	7		;DZ
1250	003020	007			.BYTE	7		;KMC
1251		003022		.EVEN				
1252	003022	000000		XLOC:	0			
1253	003024	005705		AUDONE:	TST	R5		;DMC?
1254	003026	001005			BNE	1\$		;BR IF KMC AND ALL DONE
1255	003030	012705	177777		MOV	#-1,R5		;SET R5 TO -1 (KMC)
1256	003034	012700	001500		MOV	#DM,MAP,R0		;RESET R0 TO START OF TABLE
1257	003040	000602			BR	AUSTRT		;GO DO KMC'S
1258	003042	012637	000006	1\$:	MOV	(SP)+,@#6		;RESTORE LOC 6
1259	003046	012637	000004		MOV	(SP)+,@#4		;RESTORE LOC 4
1260	003052	032737	000010	001236	BIT	#SW03,STRTSW		;SELECT SPECIFIC DEVICES??
1261	003060	001422			BEQ	3\$		;BR IF NO.
1262	003062	104402	006144		TYPE	,MNEW		;TYPE THE MESSAGE.
1263	003066	005000			CLR	R0		;ZERO DATA LIGHTS
1264	003070	000000			HALT			;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
1265	003072	027737	176104	001312	CMP	@SWR,SAVACT		;IS THE NUMBER VALID?
1266	003100	101404			BLOS	2\$		;BR IF NUMBER IS OK.
1267	003102	104402	006005		TYPE	,MERR3		;TELL USER OF INVALID NUMBER.
1268	003106	000000			HALT			;STOP EVERY THING.
1269	003110	000776			BR	.-2		;RESTART THE PROGRAM AGAIN.
1270	003112	017737	176064	001306	2\$:	MOV	@SWR,DMACTV	;GET NEW DEVICE PATTERN
1271	003120	013700	001306		MOV	DMACTV,R0		;SHOW THE USER WHAT HE SELECTED.
1272	003124	000000			HALT			;CONTINUE DYNAMIC SWITCHES.
1273	003126	012700	000300		3\$:	MOV	#300,R0	;PREPARE TO CLEAR THE FLOATING
1274	003132	012701	000302		MOV	#302,R1		;VECTOR AREA. 300-776
1275	003136	010120			4\$:	MOV	R1,(R0)+	;START PUTTING 'PC+2 - HALT'
1276	003140	005021			CLR	(R1)+		;IN VECTOR AREA.
1277	003142	022021			CMP	(R0)+,(R1)+		;POP POINTERS
1278	003144	022700	001000		CMP	#1000,R0		;ALL DONE??
1279	003150	001372			BNE	4\$		;BR IF NO.
1280								
1281					;TEST START AND RESTART			
1282					-----			
1283								
1284	003152	012706	001200		.BEGIN: MOV	#CTACK,SP		;SET UP STACK
1285	003156	013746	000006		MOV	@#6,-(SP)		;SAVE LOC 6

1286	003162	013746	000004		MOV	@#4,-(SP)	:SAVE LOC 4	
1287	003166	005000			CLR	R0	:START AT 0	
1288	003170	012737	003234	000004	MOV	#2\$,@#4	:SET UP FOR TIME OUT	
1289	003176	005037	000006		CLR	@#6	:TO AUTOSIZE MEMORY	
1290	003202	005720		6\$:	TST	(R0)+	:CHECK ADDRESS IN R0	
1291	003204	022700	157776		CMP	#157776,R0	:IS IT AT LEAST 28K	
1292	003210	001374			BNE	6\$	:BR IF NO	
1293	003212	162700	007776		SUB	#7776,R0	:SAVE 2K FOR MONITORS	
1294	003216	010037	001304	7\$:	MOV	R0,MEMLIM	:STORE MEMORY LIMIT	
1295	003222	012637	000004		MOV	(SP)+,@#4	:RESTORE LOC 4	
1296	003226	012637	000006		MOV	(SP)+,@#6	:RESTORE LOC 6	
1297	003232	000413			BR	10\$	:CONTINUE	
1298	003234	022626		2\$:	CMP	(SP)+,(SP)+	:ADJUST STACK	
1299	003236	162700	000004		SUB	#4,R0	:GET LAST GOOD ADDRESS	
1300	003242	162700	007776		SUB	#7776,R0	:SAVE 2K FOR MONITORS	
1301	003246	022700	030000		CMP	#30000,R0	:IS IT 8K?	
1302	003252	001361			BNE	7\$	:BR IF NO	
1303	003254	012700	037400		MOV	#37400,R0	:IF 8K DON'T SAVE 2K	
1304	003260	000756			BR	7\$	:	
1305	003262	012737	000340	177776	10\$:	MOV	#340,PS	:LOCK OUT INTERRUPTS
1306	003270	032737	000004	001236	BIT	#BIT2,STRTSW	:CHECK FOR LOCK ON TEST	
1307	003276	001411			BEQ	1\$	:BR IF NO LOCK DESIRED.	
1308	003300	104402	006043		TYPE	,MLOCK	:TYPE LOCK SELECTED.	
1309	003304	012737	000240	003612	MOV	#NOP,TTST	:ADJUST SCOPE ROUTINE.	
1310	003312	012737	000240	003614	MOV	#NOP,TTST+2	:SET UP TO LOCK	
1311	003320	000406			BR	3\$	:CONTINUE ALONG.	
1312	003322	013737	003730	003612	1\$:	MOV	BRW,TTST	:PREPARE NORMAL SCOPE ROUTINE
1313	003330	013737	003732	003614	MOV	BRX,TTST+2	:LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP	
1314	003336	012737	010060	001214	3\$:	MOV	#CYCLE,RETURN	:START AT 'CYCLE' FIND WHICH DEVICE TO TEST
1315	003344	032737	000002	001236	4\$:	BIT	#SW01,STRTSW	:IS TEST NO. SELECTED?
1316	003352	001002			BNE	5\$	:BR IF YES	
1317	003354	104402	005755		TYPE	,MR	:TYPE R	
1318	003360	000177	175630	5\$:	JMP	@RETURN	:START TESTING	


```

1375 003604 032777 040000 175370      BIT      #BIT14,@SWR      ;'LOOP ON THIS TEST'?
1376 003612 001407      TTST:  BEQ      1$              ;BR IF NO. (IF LOCK SW01=1; THIS LOC =240)
1377 003614 000437      BR       3$              ;GOTO 3$ (IF LOCK SW01=1; THIS LOC =240)
1378 003616 005737 003734      TST      DONE          ;WAS TKCSR DONE SET?
1379 003622 001434      BEQ      3$              ;BR IF NO (LOCKED ON TEST)
1380 003624 005037 003734      CLR      DONE          ;YES, CLEAR FLAG
1381 003630 000415      BR       2$              ;GO TO NEXT TEST
1382 003632 032777 004000 175342 1$:  BIT      #SW11,@SWR      ;DELETE ITERATION? (QUICK PASS)
1383 003640 001011      BNE      2$              ;BR IF YES
1384 003642 105737 001327      TSTB     QV.FLG          ;HAVE PASSES BEECOMPLETED?
1385 003646 001406      BEQ      2$              ;BR IF QUICK PASS.
1386 003650 005237 001224      INC      LPCNT          ;UPDATE ITERATION COUNTER
1387 003654 023737 001224 001222      CMP      LPCNT,ICOUNT      ;ARE ALL ITERATIONS DONE??
1388 003662 101414      BLOS     3$              ;BR IF NOT YET
1389 003664 105037 001325      CLRB     ERRFLG          ;PREPARE FOR NEW TEST
1390 003670 005037 001224      CLR      LPCNT          ;START ICOUNTER AT 0
1391 003674 005037 001220      CLR      LOCK          ;
1392 003700 012737 000020 001222      MOV      #20,ICOUNT      ;RESET ITERATIONS
1393 003706 013737 001216 001214      MOV      NEXT,RETURN      ;GET NEXT TEST
1394 003714 011600      3$:  MOV      (SP),R0          ;POP R0 OFF OF THE STACK
1395 003716 022626      POP2SP          ;FAKE AN 'RTI'
1396 003720 013701 001404      MOV      DMCSR,R1          ;R1 CONTAINS BASE DMC ADDRESS
1397 003724 000177 175264      JMP      @RETURN          ;GO DO THE TEST
1398 003730 001407      BRW:    1407
1399 003732 000437      BRX:    437
1400 003734 000000      DONE:   0
1401
1402
1403      ;CHECK FOR FREEZE ON CURRENT DATA
1404      ;-----
1405 003736 004737 007606      .SCOPI: JSR      PC,CKSWR          ;CHECK FOR SOFT SWR
1406 003742 032777 001000 175232      BIT      #SW09,@SWR      ;IS SW09=1(SET)?
1407 003750 001405      BEQ      1$              ;BR IF NOT SET.
1408 003752 005737 001220      TST      LOCK          ;
1409 003756 001402      BEQ      1$              ;
1410 003760 013716 001220      MOV      LOCK,(SP)          ;GOTO THE ADDRESS IN LOCK.
1411 003764 000002      1$:  RTI              ;GO BACK.
1412
1413      ;TELETYPE OUTPUT ROUTINE
1414      ;-----
1415
1416 003766 010546      .TYPE:  MOV      R5,-(SP)          ;SAVE R5 ON THE STACK.
1417 003770 017605      MOV      @2(SP),R5          ;GET ADDRESS OF MESSAGE.
1418 003774 062766 000002 000002      ADD      #2,2(SP)          ;POP OVER ADDRESS.
1419 004002 005737 010016      4$:  TST      SWFLG          ;SOFT SWR MESSAGE?
1420 004006 001004      BNE      1$              ;IF YES TYPE IT OUT REGARDLFSS OF SW12
1421 004010 032777 010000 175164      BIT      #SW12,@SWR      ;INHIBIT ALL PRINT OUT??
1422 004016 001012      BNE      3$              ;BR IF NO PRINT OUT WANTED (SW12=1)
1423 004020 105715      1$:  TSTB     (R5)          ;IS NUMBER MINUS? (MSB=1(BIT7))
1424 004022 100002      BPL      2$              ;BR IF NUMBER IS PLUS
1425 004024 104402 005672      TYPE     ,MCRLF          ;TYPE A CR/LF!
1426 004030 105777 175154      2$:  TSTB     @TPCSR          ;TTY READY?
1427 004034 100375      BPL      2$              ;BR IF NO.
1428 004036 112577 175150      MOV      (R5)+,@TPDBR      ;PRINT CURRENT CHAR.
1429 004042 001357      BNE      4$              ;IF NOT ZERO KEEP PRINTING.
1430 004044 012605      3$:  MOV      (SP)+,R5          ;END OF OUTPUT. RESTORE R5

```

```

1431 004046 000002          RTI          ;GO HOME
1432                      ;-----
1433
1434 004050 010346          .INSTR: MOV     R3,-(SP)          ;SAVE R3 ON STACK
1435 004052 010446          MOV     R4,-(SP)          ;SAVE R4 ON STACK
1436 004054 017637 000004 004072  MOV     @4(SP),.MSG
1437 004062 062766 000002 000004  ADD     #2,4(SP)
1438 004070 104402          .INST1: TYPE
1439 004072 000000          .MSG: 0
1440 004074 012704 007502          MOV     #INBUF,R4
1441 004100 012703 000007          MOV     #7,R3
1442 004104 105777 175074          1$:  TSTB   @TKCSR
1443 004110 100375          BPL     1$
1444 004112 117714 175070          MOVB   @TKDBR,(R4)
1445 004116 142714 000200          BICB   #200,(R4)
1446 004122 122427 000015          CMPB   (R4)+,#15
1447 004126 001417          BEQ     INSTR2
1448 004130 105777 175054          2$:  TSTB   @TPCSR
1449 004134 100375          BPL     2$
1450 004136 017777 175044 175046  MOV     @TKDBR,@TPDBR
1451 004144 005303          DEC     R3
1452 004146 001356          BNE     1$
1453 004150 012604          MOV     (SP)+,R4
1454 004152 012603          MOV     (SP)+,R3
1455 004154 104402 005666          .INSTE: TYPE  ,MQM
1456 004160 010346          MOV     R3,-(SP)
1457 004162 010446          MOV     R4,-(SP)
1458 004164 000741          BR      .INST1
1459 004166 012604          INSTR2: MOV    (SP)+,R4          ;RESTORE R4
1460 004170 012603          MOV     (SP)+,R3          ;RESTORE R3
1461 004172 000002          RTI
1462
1463                      ;CONVERT ASCII STRING TO OCTAL
1464                      ;-----
1465
1466 004174 010546          .PARAM: MOV     R5,-(SP)
1467 004176 010446          MOV     R4,-(SP)
1468 004200 016605 000004          MOV     4(SP),R5
1469 004204 012537 004364          MOV     (R5)+,LOLIM
1470 004210 012537 004366          MOV     (R5)+,HILIM
1471 004214 012537 004370          MOV     (R5)+,DEVADR
1472 004220 112537 004372          MOVB   (R5)+,LOBITS
1473 004224 112537 004373          MOVB   (R5)+,ADRCNT
1474 004230 010566 000004          MOV     R5,4(SP)
1475 004234 005005          PARAM1: CLR     R5
1476 004236 012704 007502          MOV     #INBUF,R4
1477 004242 122714 000015          CMPB   #15,(R4)
1478 004246 001420          BEQ     PARERR
1479 004250 121427 000060          1$:  CMPB   (R4),#60
1480 004254 002415          BLT     PARERR
1481 004256 121427 000067          CMPB   (R4),#67
1482 004262 003012          BGT     PARERR
1483 004264 142714 000060          BICB   #60,(R4)
1484 004270 152405          BISB   (R4)+,R5
1485 004272 122714 000015          CMPB   #15,(R4)
1486 004276 001406          BEQ     LIMITS

```

```
1487 004300 006305          ASL      R5
1488 004302 006305          ASL      R5
1489 004304 006305          ASL      R5
1490 004306 000760          BR        1$
1491 004310 104404          PARERR: INSTER
1492 004312 000750          BR        PARAM1
1493
1494          ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
1495          ;-----
1496
1497 004314 020537 004366    LIMITS: CMP      R5,HILIM
1498 004320 101373          BHI      PARERR
1499 004322 020537 004364    CMP      R5,LOLIM
1500 004326 103770          BLO      PARERR
1501 004330 133705 004372    BITB     LOBITS,R5
1502 004334 001365          BNE      PARERR
1503
1504          ;STORE NUMBER AT SPECIFIED ADDRESS
1505
1506 004336 013704 004370    1$:      MOV      DEVADR,R4
1507 004342 010524          MOV      R5,(R4)+
1508 004344 062705 000002    ADD      #2,R5
1509 004350 105337 004373    DECB     ADRCNT
1510 004354 001372          BNE      1$
1511 004356 012604          MOV      (SP)+,R4
1512 004360 012605          MOV      (SP)+,R5
1513 004362 000002          RTI
1514 004364 000000          LOLIM: 0
1515 004366 000000          HILIM: 0
1516 004370 000000          DEVADR: 0
1517 004372 000000          LOBITS: 0
1518          ADRCNT=LOBITS+1
1519
1520          ;SAVE PC OF TEST THAT FAILED AND R0-R5
1521          ;-----
1522
1523 004374 016637 000004 001276 .SAV05: MOV      4(SP),SAVPC      ;SAVE R7 (PC)
1524
1525          ;SAVE R0-R5
1526
1527 004402 010537 001272    SV05:  MOV      R5,SAVR5      ;SAVE R5
1528 004406 010437 001270    MOV      R4,SAVR4      ;SAVE R4
1529 004412 010337 001266    MOV      R3,SAVR3      ;SAVE R3
1530 004416 010237 001264    MOV      R2,SAVR2      ;SAVE R2
1531 004422 010137 001262    MOV      R1,SAVR1      ;SAVE R1
1532 004426 010037 001260    MOV      R0,SAVR0      ;SAVE R0
1533 004432 000002          RTI      ;LEAVE.
1534
1535          ;RESTORE R0-R5
1536
1537 004434 013700 001260    .RES05: MOV      SAVR0,R0      ;RESTORE R0
1538 004440 013701 001262    MOV      SAVR1,R1      ;RESTORE R1
1539 004444 013702 001264    MOV      SAVR2,R2      ;RESTORE R2
1540 004450 013703 001266    MOV      SAVR3,R3      ;RESTORE R3
1541 004454 013704 001270    MOV      SAVR4,R4      ;RESTORE R4
1542 004460 013705 001272    MOV      SAVR5,R5      ;RESTORE R5
```

```

1543 004464 000002 RTI ;LEAVE
1544
1545 ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
1546 -----
1547
1548 004466 104402 005672 .CONVR: TYPE ,MCRLF
1549 004472 010046 .CNVRT: MOV R0,-(SP)
1550 004474 010146 MOV R1,-(SP)
1551 004476 010346 MOV R3,-(SP)
1552 004500 010446 MOV R4,-(SP)
1553 004502 010546 MOV R5,-(SP)
1554 004504 017601 000012 MOV @12(SP),R1
1555 004510 062766 000002 000012 ADD #2,12(SP)
1556 004516 012137 004710 MOV (R1)+,WRDCNT
1557 004522 112137 004712 1$: MOV (R1)+,CHRCNT
1558 004526 112137 004713 MOV (R1)+,SPACNT
1559 004532 013137 004714 MOV @ (R1)+,BINWRD
1560 004536 122737 000003 004712 CMPB #3,CHRCNT
1561 004544 001003 BNE 2$
1562 004546 042737 177400 004714 BIC #177400,BINWRD
1563 004554 013704 004714 2$: MOV BINWRD,R4
1564 004560 113705 004712 MOVB CHRCNT,R5
1565 004564 012700 001416 MOV #TEMP,R0
1566 004570 010403 3$: MOV R4,R3
1567 004572 042703 177770 BIC #177770,R3
1568 004576 062703 000060 ADD #060,R3
1569 004602 110320 MOVB R3,(R0)+
1570 004604 000241 CLC
1571 004606 006004 ROR R4
1572 004610 000241 CLC
1573 004612 006004 ROR R4
1574 004614 000241 CLC
1575 004616 006004 ROR R4
1576 004620 005305 DEC R5
1577 004622 001362 BNE 3$
1578 004624 012703 007544 MOV #MDATA,R3
1579 004630 114023 4$: MOVB -(R0),(R3)+
1580 004632 105337 004712 DECB CHRCNT
1581 004636 001374 BNE 4$
1582 004640 105737 004713 TSTB SPACNT
1583 004644 001405 BEQ 6$
1584 004646 112723 000040 5$: MOVB #040,(R3)+
1585 004652 105337 004713 DECB SPACNT
1586 004656 001373 BNE 5$
1587 004660 105013 6$: CLRB (R3)
1588 004662 104402 007544 TYPE ,MDATA
1589 004666 005337 004710 DEC WRDCNT
1590 004672 001313 BNE 1$
1591 004674 012605 MOV (SP)+,R5
1592 004676 012604 MOV (SP)+,R4
1593 004700 012603 MOV (SP)+,R3
1594 004702 012601 MOV (SP)+,R1
1595 004704 012600 MOV (SP)+,R0
1596 004706 000002 RTI
1597 004710 000000 WRDCNT: 0
1598 004712 000000 CHRCNT: 0

```

1599		004713				SPACNT=CHRCNT+1	
1600	004714	000000				BINWRD: 0	
1601							
1602							
1603						:TRAP DISPATCH SERVICE	
1604						:ARGUMENT OF TRAP IS EXTRACTED	
1605						:AND USED AS OFFSET TO OBTAIN POINTER	
1606						:TO SELECTED SUBROUTINE	
1607							
1608	004716	011646				.TRPSR: MOV (SP),-(SP)	:GET PC OF RETURN
1609	004720	162716	000002			SUB #2,(SP)	:PC OF TRAP
1610	004724	017616	000000			MOV @ (SP),(SP)	:GET TRP
1611	004730	006316				TRPOK: ASL (SP)	:MULTIPLY TRAP ARG BY 2
1612	004732	042716	177001			BIC #177001,(SP)	:CLEAR UNWANTED BITS
1613	004736	062716	001330			ADD #.TRPTAB,(SP)	:POINTER TO SUBROUTINE ADDRESS
1614	004742	017616	000000			MOV @ (SP),(SP)	:SUBROUTINE ADDRESS
1615	004746	000136				JMP @ (SP)+	:GO TO SUBROUTINE
1616							
1617						:ERROR HANDLER	
1618						-----	
1619							
1620	004750	004737	007606			.HLT: JSR PC,CKSWR	:CHECK FOR SOFT SWR
1621	004754	032777	010000	174220		BIT #SW12,@SWR	:BELL ON ERROR?
1622	004762	001406				BEQ XBK	:BR IF NO BELL
1623	004764	105777	174220			TSTB @TPCSR	:TTY READY.
1624	004770	100003				BPL XBK	:DON'T WAIT IF TTY NOT READY.
1625	004772	112777	000207	174212		MOV #207,@TPDBR	:PUSH A BELL AT THE TTY.
1626	005000	032777	020000	174174		XBK: BIT #SW13,@SWR	:DELETE ERROR PRINT OUT?
1627	005006	001105				BNE HALTS	:BR IF NO PRINT OUT WANTED.
1628	005010	021637	001234			CMP (SP),LSTERR	:WAS THIS ERROR FOUND LAST TIME?
1629	005014	001404				BEQ 1\$	:BR IF YES
1630	005016	011637	001234			MOV (SP),LSTERR	:RECORD BEING HERE
1631	005022	105037	001325			CLRB ERRFLG	:PREPARE HEADER
1632	005026	104406				1\$: SAVO5	:SAVE ALL PROC REGISTERS
1633	005030	011605				MOV (SP),R5	:GET THE PC OF ERROR
1634	005032	162705	000002			SUB #2,R5	:GET ADDRESS OF TRAP CALL
1635	005036	011504				MOV (R5),R4	:GET HLT INSTRUCTION
1636	005040	006304				ASL R4	:MULT BY TWO
1637	005042	061504				ADD (R5),R4	:DOUBLE IT
1638	005044	006304				ASL R4	:MULT AGAIN
1639	005046	042704	177001			BIC #177001,R4	:CLEAR JUNK
1640	005052	062704	031704			ADD #.ERRTAB,R4	:GET POINTER
1641	005056	012437	005172			MOV (R4)+,ERRMSG	:GET ERROR MESSAGE
1642	005062	012437	005204			MOV (R4)+,DATAHD	:GET DATA HEADRER
1643	005066	011437	005216			MOV (R4),DATABP	:GET DATA TABLE
1644	005072	105737	001325			TSTB ERRFLG	:TYPE HEADREER
1645	005076	001403				BEQ TYPMSG	:BR IF YES
1646	005100	005737	005216			TST DATABP	:DOES DATA TABLE EXIST?
1647	005104	001040				BNE TYFDAT	:BR IF YES.
1648	005106	104402	005672			TYPMSG: TYPE ,MCRLF	
1649	005112	104402	005672			TYPE ,MCRLF	
1650	005116	005737	001220			TST LOCK	
1651	005122	001402				BEQ 1\$	
1652	005124	104402	006142			TYPE ,MASTEK	
1653	005130	104402	006130			1\$: TYPE ,TTSTN	
1654	005134	104411	005330			CNVRT ,XTSTN	:SHOW IT

```

1655 005140 104402 006217      TYPE      ,MEPRPC      ;TYPE PC.
1656 005144 104411 005322      CNVRT      ,ERTAB0      ;SHOW IT
1657 005150 104402 005672      TYPE      ,MCRLF      ;GIVE A CR/LF
1658 005154 112737 177777 001325  MOV      #-1,ERRFLG  ;NO MORE HEADER UNLESS NO DATA TABLE.
1659 005162 005737 005172      TST      ERRMSG      ;IS THERE AN ERROR MESSAGE?
1660 005166 001402      BEQ      WRKO.FM      ;BR IF NO.
1661 005170 104402      TYPE      ;TYPE
1662 005172 000000      ERRMSG: 0      ;ERROR MESSAGE
1663 005174      WRKO.FM:      ;
1664 005174 005737 005204      TST      DATAHD      ;DATA HEADER?
1665 005200 001402      BEQ      TYPDAT      ;BR IF NO
1666 005202 104402      TYPE      ;TYPE
1667 005204 000000      DATAHD: 0      ;DATA HEADER
1668 005206 005737 005216      TYPDAT: TST      DATABP      ;DATA TABLE?
1669 005212 001402      BEQ      RESREG      ;BR IF NO.
1670 005214 104410      CONVRT      ;SHOW
1671 005216 000000      DATABP: 0      ;DATA TABLE
1672 005220 104407      RESREG: RES05      ;RESTORE PROC REGISTERS
1673 005222 022737 003522 000042  HALTS:  CMP      #SENDAD,@#42 ;IF ACT-11 AUTOMATIC MODE, HALT.!
1674 005230 001403      BEQ      1$      ;
1675 005232 005777 173744      TST      @SWR      ;HALT ON ERROR?
1676 005236 100005      BPL      EXITER      ;BR IF NO HALT ON ERROR
1677 005240 010046      1$:  PUSHRO      ;SAVE RO
1678 005242 016600 000002      MOV      2(SP),RO      ;SHOW ERROR PC IN DATA LIGHTS
1679 005246 000000      HALT      ;HALT
1680 005250 012600      POPRO      ;GET RO
1681 005252 005237 001232      EXITER: INC      ERRCNT      ;UPDATE ERROR COUNT
1682 005256 032777 000400 173716  BIT      #SW08,@SWR      ;GOTO TOP OF TEST?
1683 005264 001007      BNE      1$      ;BR IF YES
1684 005266 032777 002000 173706  BIT      #SW10,@SWR      ;GOTO NEXT TEST?
1685 005274 001411      BEQ      2$      ;BR IF NO
1686 005276 013737 001216 001214  MOV      NEXT,RETURN ;SET FOR NEXT TEST
1687 005304 012706 001200      1$:  MOV      #STACK,SP      ;RESET SP
1688 005310 013701 001404      MOV      DMCSR,R1      ;SET UP R1
1689 005314 000177 173674      JMP      @RETURN      ;GOTO SPECIFIED TEST
1690 005320 000002      2$:  RTI      ;RETURN
1691 005322 000001      ERTAB0: 1      ;
1692 005324      006      002      .BYTE      6,2
1693 005326 001276      SAVPC
1694 005330 000001      XTSTN: 1
1695 005332      003      002      .BYTE      3,2
1696 005334 001226      TSTNO
1697      ;ENTER HERE ON POWER FAILURE
1698      ;-----
1699
1700
1701 005336      .PFAIL:
1702 005336 012737 005350 000024  MOV      #RESTART,24      ;SET UP FOR POWER UP TRAP
1703 005344 000000      HALT      ;HALT ON POWER DOWN NORMAL
1704 005346 000777      BR
1705
1706      ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
1707
1708 005350      RESTAR:
1709 005350 012737 005336 000024  MOV      #.PFAIL,24      ;SET UP FOR POWER FAILURE
1710 005356 012706 001200      MOV      #STACK,SP      ;RESET THE STACK POINTER

```

```
1711 005362 013701 001404      MOV      DMCSR,R1      ;RESTORE R1
1712 005366 005037 001416      CLR      TEMP      ;READY FOR TIMER
1713 005372 005237 001416      INC      TEMP      ;PLUS ONE TO THE TIMER!
1714 005376 001375      BNE      -4      ;BR IF MORE TO GO
1715 005400 104402 005675      TYPE     ,MPFAIL      ;TYPE THE MESSAGE
1716 005404 104411 005430      CNVRT    ,PFTAB      ;TELL WHAT TEST TO RETURN TO.
1717 005410 105037 001325      CLRB     ERRFLG      ;START CLEAN
1718 005414 005037 001234      CLR      LSTERR      ;.....
1719 005420 005011      CLR      (R1)      ;CLEAR MAINT BITS
1720 005422 104412      MSTCLR      ;START CLEAN UP OF DEVICE
1721 005424 000177 173564      JMP      @RETURN      ;START DOING THAT TEST AGAIN.
1722 005430 000001      PFTAB: 1
1723 005432 003      .BYTE 3,2
1724 005434 001226      TSTNO
1725
1726 005436      .DELAY:
1727 005436 012777 000020 173746      MOV      #20,@DMP04
1728 005444 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1729 005446 121111      121111      ;POKE CLOCK DELAY BIT
1730 005450
1731 005450 104414      1$: ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1732 005452 121224      121224      ;PORT4 IBUS*11
1733 005454 032777 000020 173730      BIT      #BIT4,@DMP04      ;IS CLOCK BIT SET?
1734 005462 001772      BEQ      1$      ;BR IF NO
1735 005464 000002      RTI
1736
1737 005466      .MSTCLR:
1738 005466 152777 000100 173712      BISB     #BIT6,@DMCSRH      ;SET MASTER CLEAR
1739 005474 142777 000300 173704      BICB     #BIT6!BIT7,@DMCSRH ;CLEAR MASTER CLEAR AND RUN
1740 005502 000002      RTI      ;RETURN
1741
1742 005504      .ROMCLK:
1743 005504 152777 000002 173674      BISB     #BIT1,@DMCSRH      ;SET ROMI
1744 005512 013677 173676      MOV      @ (SP)+,@DMP06      ;LOAD INSTRUCTION IN SEL6
1745 005516 062746 000002      ADD      #2,-(SP)      ;ADJUST STACK
1746 005522 032777 000100 173452      BIT      #SW06,@SWR      ;HALT IF SW06 =1
1747 005530 001401      BEQ      1$      ;BR IF SW06 =0
1748 005532 000000      HALT      ;HALT BEFORE CLOCKING INSTRUCTION
1749 005534 152777 000003 173644      1$: BISB     #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
1750 005542 142777 000007 173636      BICB     #BIT2!BIT1!BIT0,@DMCSRH ;CLEAR ROMO, ROMI, STEP
1751 005550 000002      RTI
1752
1753 005552      .DATACLK:
1754 005552 013637 001416      MOV      @ (SP)+,TEMP      ;PUT TICK COUNT IN TEMP
1755 005556 062746 000002      ADD      #2,-(SP)      ;ADJUST STACK
1756 005562 152777 000020 173616      1$: BISB     #BIT4,@DMCSRH      ;SET STEP LU
1757 005570 027777 173610 173606      CMP      @DMCSR,@DMCSR      ;WASTE TIME
1758 005576 142777 000020 173602      BICB     #BIT4,@DMCSRH      ;CLEAR STEP LU
1759 005604 005337 001416      DEC      TEMP      ;DEC TICK COUNT
1760 005610 001364      BNE      1$      ;BR IF NOT DONE
1761 005612 000002      RTI      ;RETURN
1762 005614 000001      3$: .BLKW 1
1763
1764 005616      .TIMER:
1765 005616 013637 001416      MOV      @ (SP)+,TEMP      ;MOVE COUNT TO TEMP
1766 005622 062746 000002      ADD      #2,-(SP)      ;ADJUST STACK
```

```

1767 005626 104414 1$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1768 005626 021364 021364 ;PORT4 IBUS* REG11
1769 005630 032777 000002 173552 BIT #2, @DMP04 ;IS PGM CLOCK BIT CLEAR?
1770 005632 001772 BEQ 1$ ;BR IF YES
1771 005640 104414 2$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1772 005642 021364 021364 ;PORT4 IBUS* REG11
1773 005642 032777 000002 173536 BIT #2, @DMP04 ;IS PGM CLOCK BIT SET?
1774 005644 001372 BNE 2$ ;BR IF YES
1775 005646 005337 001416 DEC TEMP ;DEC COUNT
1776 005654 001361 BNE 1$ ;BR IF NOT DONE
1777 005662 000002 RTI ;RETURN
1778 005664
1779
1780
1781 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 CZDME /
(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 MASTEK: .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: .ASCII <212>/
(2) 006263 377 020040 020040 .ASCII <377>/
(2) 006322 020212 050040 020103 .ASCII <212>/ PC CSR STAT1 STAT2 STAT3/
(2) 006374 026777 026455 026455 .ASCII <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 TYP
(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><1>>/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 DMC1: .ASCIIZ / (DMC) /
(2) 007330 024040 046513 024503 KMC1: .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) XSTATQ: 5
1782 007454 000005 .BYTE 6, 3
1783 007456 006 003 TEMP1
1783 007460 001246
  
```

```

1784 007462 006 003 .BYTE 6,3
1785 007464 001250 TEMP2
1786 007466 006 003 .BYTE 6,3
1787 007470 001252 TEMP3
1788 007472 006 003 .BYTE 6,3
1789 007474 001254 TEMP4
1790 007476 006 002 .BYTE 6,2
1791 007500 001256 TEMP5
1792 .EVEN
1793
1794 ;BUFFERS FOR INPUT-OUTPUT
1795
1796 007502 000000 INBUF: 0
1797 007544 .-.+40
1798 007544 000000 MDATA: 0
1799 007606 .-.+40
1800
1801
1802 ;ROUTINE USED TO CHANGE SOFTWARE SWITCH
1803 ;REGISTER USING THE CONSOLE TERMINAL
1804 -----
1805
1806 007606 022737 000176 001202 CKSWR: CMP #SWREG,SWR ;IS THE SOFT SWR BEING USED?
1807 007614 001077 BNE CKSWR5 ;BR IF NO
1808 007616 105777 171362 TSTB @TKCSR ;IS DONE SET?
1809 007622 100003 BPL 2$ ;GO ON IF NOT SET
1810 007624 012737 177777 003734 MOV #-1,DONE ;IF DONE SET, SET FLAG
1811 007632 022777 000007 171346 2$: CMP #7,@TKDBR ;WAS CTRL G TYPED? (7 BIT ASCII)
1812 007640 001404 BEQ 1$ ;BR IF YES
1813 007642 022777 000207 171336 CMP #207,@TKDBR ;WAS CTRL G TYPED? (8 BIT ASCII)
1814 007650 001061 BNE CKSWR5 ;BR IF NO
1815 007652 010246 1$: MOV R2,-(SP) ;STORE R2
1816 007654 010346 MOV R3,-(SP) ;STORE R3
1817 007656 010446 MOV R4,-(SP) ;STORE R4
1818 007660 012737 177777 010016 MOV #-1,SWFLG ;SET SOFT TYPE OUT FLAG
1819 007666 005002 CKSWR1: CLR R2 ;CLEAR NEW SWR CONTENTS
1820 007670 012704 177777 MOV #-1,R4 ;SET FLAG TO ALL ONES
1821 007674 104402 007205 TYPE ,SWMES ;TYPE 'SWR='
1822 007700 104411 CKSWR2: CNVRT ;TYPE OUT PRESENT CONTENTS
1823 007702 010052 SOFTSW ;OF SOFT SWITCH REGISTER
1824 007704 104402 007215 CKSWR3: TYPE ,SWMES1 ;TYPE 'NEW?'
1825 007710 004737 010020 CKSWR4: JSR PC,INCHAR ;GET RESPONSE
1826 007714 022703 000015 CMP #15,R3 ;WAS IT A CR?
1827 007720 001424 BEQ 5$ ;BR IF YES
1828 007722 022703 000012 CMP #12,R3 ;WAS IT A LF?
1829 007726 001416 BEQ 4$ ;BR IF YES
1830 007730 022703 000025 CMP #25,R3 ;WAS IT CTRL U?
1831 007734 001754 BEQ CKSWR1 ;BR IF YES(START OVER)
1832 007736 022703 000007 CMP #7,R3 ;IF CNTL G GET NEXT CHAR
1833 007742 001762 BEQ CKSWR4
1834 007744 005004 CLR R4 ;IT MUST BE A DIGIT SO CLR FLAG
1835 007746 042703 177770 BIC #177770,R3 ;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
1836 007752 006302 ASL R2 ;SHIFT R2 3 TIMES
1837 007754 006302 ASL R2
1838 007756 006302 ASL R2
1839 007760 050302 BIS R3,R2 ;ADD LAST DIGIT

```

```

1840 007762 000752          BR      CKSWR4      ;GET NEXT CHARACTER
1841 007764 012766 002002 000006 4$:  MOV      #.START,6(SP) ;LF WAS TYPED SO GO TO START
1842 007772 005704          5$:  TST      R4      ;IS FLAG CLEAR?
1843 007774 001002          BNE      6$      ;IF NOT DON'T CHANGE SOFT SWR
1844 007776 010277 171200    MOV      R2,@SWR    ;IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
1845 010002 005037 010016    6$:  CLR      SWFLG   ;CLEAR TYPEOUT FLAG
1846 010006 012604          MOV      (SP)+,R4   ;RESTORE R4
1847 010010 012603          MOV      (SP)+,R3   ;RESTORE R3
1848 010012 012602          MOV      (SP)+,R2   ;RESTORE R2
1849 010014 000207          CKSWR5: RTS      PC   ;RETURN
1850
1851 010016 000000          SWFLG: 0
1852
1853 010020 105777 171160    INCHAR: TSTB     @TKCSR
1854 010024 100375          BPL      .-4
1855 010026 017703 171154    MOV      @TKDDBR,R3
1856 010032 105777 171152    TSTB     @TPCSR
1857 010036 100375          BPL      .-4
1858 010040 010377 171146    MOV      R3,@TPDDBR
1859 010044 042703 000200    BIC      #BIT7,R3
1860 010050 000207          RTS      PC
1861
1862 010052 000001          SOFTSW: 1
1863 010054 006          .BYTE 6,2
1864 010056 000176          SWREG

```

```

1865
1866
1867      ;ROUTINE USED TO 'CYCLE' THROUGH UP TO 16 DMC11'S
1868      ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
1869      ;AND RUNS THE SPECIFIED DMC11'S.  THIS ROUTINE *MUST*
1870      ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
1871      ;SETUP NECESSARY.
1872
1873
1874 010060 005737 001306      CYCLE:  TST      DMACTV      ;ARE ANY DMC11'S TO BE TESTED?
1875 010064 001004              BNE      1$          ;BR IF OK.
1876 010066 104402 007154      TYPE      ,NOACT      ;NO DMC11'S SELECTED!!
1877 010072 000000              HALT              ;STOP THE SHOW.
1878 010074 000776              BR          -2        ;DISQUALIFY CONT. SW.
1879 010076 000241              1$:  CLC          ;CLEAR PROC. CARRY BIT.
1880 010100 006137 001316      ROL          ;UPDATE POINTER
1881 010104 005537 001316      RUN          ;CATCH CARRY FROM RUN
1882 010110 062737 000004 001322      ADD      #4,MILK      ;UPDATE POINTER
1883 010116 062737 000010 001320      ADD      #10,CREAM     ;UPDATE ADDRESS POINTER.
1884 010124 022737 001700 001320      CMP      #DM.MAP+200,CREAM
1885 010132 001006              BNE      2$          ;KEEP GOING; NOT ALL TESTED FOR.
1886 010134 012737 001500 001320      MOV      #DM.MAP,CREAM ;RESET ADDRESS POINTER.
1887 010142 012737 001702 001322      MOV      #CNT.MAP,MILK ;RESET PASS COUNT POINTER
1888 010150 033737 001316 001306      2$:  BIT      RUN,DMACTV ;IS THIS ONE ACTIVE?
1889 010156 001747              BEQ      1$          ;BR IF NO
1890 010160 013700 001320      MOV      CREAM,R0        ;GET ADDRESS POINTER
1891 010164 013702 001322      MOV      MILK,R2        ;GET PASS COUNT POINTER
1892 010170 012037 001404      MOV      (R0)+,DMCSR     ;LOAD SYSTEM CTRL. REG
1893 010174 011037 001374      MOV      (R0),DMRVEC     ;LOAD VECTOR
1894 010200 042737 177000 001374      BIC      #177000,DMRVEC ;CLEAR UNWANTED BITS
1895 010206 012037 001366      MOV      (R0)+,STAT1     ;LOAD STAT1
1896 010212 012037 001370      MOV      (R0)+,STAT2     ;LOAD STAT2
1897 010216 012037 001372      MOV      (R0)+,STAT3     ;LOAD STAT3
1898 010222 012237 001230      MOV      (R2)+,PASCNT     ;LOAD PASS COUNT
1899 010226 012237 001232      MOV      (R2)+,ERRCNT     ;LOAD ERROR COUNT
1900 010232 012700 000002      MOV      #2,R0          ;SAVE CORE THIS WAY!
1901 010236 013737 001404 001406      MOV      DMCSR,DMCSRH
1902 010244 005237 001406      INC      DMCSRH
1903 010250 013737 001406 001410      MOV      DMCSRH,DMCTL
1904 010256 005237 001410      INC      DMCTL
1905 010262 013737 001410 001412      MOV      DMCTL,DMP04
1906 010270 060037 001412      ADD      R0,DMP04
1907 010274 013737 001412 001414      MOV      DMP04,DMP06
1908 010302 060037 001414      ADD      R0,DMP06
1909
1910 010306 013737 001374 001376      MOV      DMRVEC,DMRLVL ;PTY LVL
1911 010314 060037 001376      ADD      R0,DMRLVL
1912 010320 013737 001376 001400      MOV      DMRLVL,DMTVEC ;TX VEC
1913 010326 060037 001400      ADD      R0,DMTVEC
1914 010332 013737 001400 001402      MOV      DMTVEC,DMTLVL ;TX LVL
1915 010340 060037 001402      ADD      R0,DMTLVL
1916
1917 010344 032737 000002 001236      BIT      #SW01,STRTSW ;IS TEST NO. SELECTED
1918 010352 001450              BEQ      7$          ;BR IF NO
1919 010354
1920 010354 005737 000042      4$:  TST      @#42      ;RUNNING IN AUTO MODE?
  
```

```

1921 010360 001045      BNE 7$          ;BR IF YES
1922 010362 104402 005672 TYPE ,MCRLF
1923 010366 104403      INSTR          ;GET TEST NO.
1924 010370 006130      MTSTN
1925 010372 104405      PARAM
1926 010374 000001      1
1927 010376 001000      1000
1928 010400 001226      TSTNO
1929 010402 000        .BYTE 0
1930 010403 001        .BYTE 1
1931 010404 012700 012320 MOV #TST1,R0
1932 010410 022710 5$: CMP (PC)+,(R0) ;CMP FIRST WORD TO 12737
1933 010412 012737      MOV (PC)+,@(PC)+
1934 010414 001020      BNE 6$          ;BR IF NOT SAME
1935 010416 023760 001226 000002 CMP TSTNO,2(R0) ;DOES TSTNO MATCH?
1936 010424 001014      BNE 6$          ;BR IF NO
1937 010426 022760 001226 000004 CMP #TSTNO,4(R0) ;IS LAST WORD OK?
1938 010434 001010      BNE 6$          ;BR IF NO
1939 010436 010037 001214 MOV R0,RETURN ;IT IS A LEGAL TEST SO DO IT
1940 010442 104402 005755 TYPE ,MR
1941 010446 042737 000002 001236 BIC #SW01,STRTSW
1942 010454 000412      BR 8$
1943 010456 005720 6$: TST (R0)+ ;POP R0
1944 010460 020027 026034 CMP R0,#TLAST+10 ;AT END YET?
1945 010464 001351      BNE 5$          ;BR IF NO
1946 010466 104402 005666 TYPE ,MQM ;YES ILLEGAL TEST NO.
1947 010472 000730      BR 4$          ;TRY AGAIN
1948
1949 010474 012737 012320 001214 7$: MOV #TST1,RETURN ;PREPARE RETURN ADDRESS
1950 010502 013701 001404 8$: MOV DMC11,R1 ;R1 = BASE DMC11 ADDRESS
1951 010506 000177 170502 JMP @RETURN ;GO START TESTING.
1952
1953
1954 ;ROUTINE USED TO 'AUTO SIZE' THE DMC11
1955 ;CSR AND VECTOR.
1956 ;NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
1957 ;ADDRESS RANGE (160000:164000)
1958 ;AND THE VECTOR MAY BE ANY WHERE IN THE
1959 ;FLOATING VECTOR RANGE (300:770)
1960 ;
1961 ;
1962 AUTO.SIZE:
1963 010512 000005      RESET
1964 010514 012702 001500 CSRMAP: MOV #DM.MAP,R2 ;INSURE A BUS INIT.
1965 010520 005022 1$: CLR (R2)+ ;LOAD MAP POINTER.
1966 010522 022702 001700 CMP #DM.END,R2 ;ZERO ENTIRE MAP
1967 010526 001374      BNE 1$          ;ALL DONE?
1968 010530 005037 001310 CLR DMNUM ;BR IF NO
1969 010534 012702 001500 MOV #DI1.MAP,R2 ;SET OCTAL NUMBER OF DMC11'S TO 0
1970 010540 005037 001306 CLR DMACTV ;R2 POINTS TO DMC MAP
1971 010544 032737 000001 001236 BIT #SW00,STRTSW ;CLEAR ACTIVE
1972 010552 001002      BNE ,+6 ;QUESTIONS?
1973 010554 000137 011252 JMP 7$ ;BR IF YES
1974 010560 012737 000001 001256 MOV #1,TEMP5 ;IF NO SKIP QUESTIONS
1975 010566 104403      INSTR ;START WITH 1
1976 010570 006450      NUM

```

CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 41

CZDMEC.P11 11-SEP-78 12:38

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

SEQ 0041

1977	010572	104405			PARAM			
1978	010574	000001			1			
1979	010576	000020			16.			
1980	010600	001252			TEMP3			
1981	010602	000			.BYTE	0		
1982	010603	001			.BYTE	1		
1983	010604	013737	001252	001310	MOV	TEMP3,DMNUM		;DMNUM = HOW MANY
1984	010612	104402	005672		TYPE	.MCRLF		
1985	010616	104410		12\$:	CONVRT			;TYPE WHICH DMC IS BEING DONE
1986	010620	012002			WHICH			;TEMP5 IS WHICH DMC
1987	010622	005237	001256		INC	TEMP5		
1988	010626	104403			INSTR			
1989	010630	006510			CSR			
1990	010632	104405			PARAM			
1991	010634	160000			160000			
1992	010636	164000			164000			
1993	010640	001254			TEMP4			
1994	010642	000			.BYTE	0		
1995	010643	001			.BYTE	1		
1996	010644	013722	001254		MOV	TEMP4,(R2)+		;STORE CSR IN MAP
1997	010650	104403			INSTR			
1998	010652	006526			VEC			
1999	010654	104405			PARAM			
2000	010656	000000			0			
2001	010660	000776			776			
2002	010662	001254			TEMP4			
2003	010664	000			.BYTE	0		
2004	010665	001			.BYTE	1		
2005	010666	013712	001254		MOV	TEMP4,(R2)		;STORE VECTOR IN MAP
2006	010672	104402		10\$:	TYPE			
2007	010674	006547			PRI0			;ASK WHAT BR LEVEL
2008	010676	004737	012266		JSR	PC,INTTY		;GET RESPONSE
2009	010702	022703	000024		CMP	#24,R3		
2010	010706	101014			BHI	50\$		;BR IF LESS THAN 4
2011	010710	022703	000027		CMP	#27,R3		
2012	010714	103411			BLO	50\$		;BR IF GREATER THAN 7
2013	010716	012704	000011		MOV	#11,R4		;R4 = NUMBER OF SHIFTS
2014	010722	006303			ASL	R3		;SHIFT R3 LEFT
2015	010724	005304			DEC	R4		;DEC SHIFT COUNT
2016	010726	001375			BNE	-4		;BR IF NOT DONE
2017	010730	042703	170777		BIC	#170777,R3		;BIC UNWANTED BITS
2018	010734	050312			BIS	R3,(R2)		;PUT BR LEVEL IN STATUS MAP
2019	010736	000403			BR	8\$		;CONTINUE
2020	010740	104402		50\$:	TYPE			
2021	010742	005666			MQM			;RESPONSE IS OUT OF LIMITS
2022	010744	000752			BR	10\$		;TRY AGAIN
2023	010746	104402		8\$:	TYPE			
2024	010750	006606			CRAM			;DOES DMC HAVE CRAM?
2025	010752	004737	012266		JSR	PC,INTTY		;GET REPLY
2026	010756	022703	000131		CMP	#131,R3		
2027	010762	001427			BEQ	9\$		;YES
2028	010764	022703	000116		CMP	#116,R3		;NO
2029	010770	001403			BEQ	40\$		;NOT A Y OR N
2030	010772	104402			TYPE			
2031	010774	005666			MQM			;TYPE '?'
2032	010776	000763			BR	8\$		;ASK AGAIN

CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 42

CZDMEC.P11 11-SEP-78 12:38

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

SEQ 0042

2033	011000	104402		40\$:	TYPE		
2034	011002	007340			SPEED		
2035	011004	004737	012266		JSR	PC,INTTY	:DMC11-AR OR DMC11-AL?
2036	011010	022703	000122		CMP	#122,R3	:GET RESPONSE
2037	011014	001414			BEQ	16\$	:IS IT R
2038	011016	022703	000114		CMP	#114,R3	:BR IF REMOTE
2039	011022	001403			BEQ	41\$	:IS IT L
2040	011024	104402			TYPE		:BR IF LOCAL
2041	011026	005666			MQM		
2042	011030	000763			BR	40\$	:TRY AGAIN
2043	011032	052762	000002	000004	41\$:	BIS	#BIT1,4(R2)
2044	011040	000402			BR	16\$	:SET BIT1 IN STAT3
2045	011042	052712	100000		9\$:	BIS	#BIT15,(R2)
2046	011046	104402			16\$:	TYPE	:CONTINUE
2047	011050	006704			MODU		:SET BIT 15 IF CRAM
2048	011052	004737	012266		JSR	PC,INTTY	:ASK WHICH LINE UNIT
2049	011056	022703	000021		CMP	#21,R3	:GET REPLY
2050	011062	001417			BEQ	30\$	: '1'
2051	011064	022703	000022		CMP	#22,R3	: '2'
2052	011070	001412			BEQ	31\$	: 'N'
2053	011072	022703	000116		CMP	#116,R3	
2054	011076	001403			BEQ	32\$	
2055	011100	104402			TYPE		
2056	011102	005666			MQM		:IF NOT A 1,2 OR N TYPE '?'
2057	011104	000760			BR	16\$	:TRY AGAIN
2058	011106	052722	010000		32\$:	BIS	#BIT12,(R2)+
2059	011112	022222			CMP	(R2)+,(R2)+	:SET BIT 12 IN STAT2 IF NO LU
2060	011114	000447			BR	33\$	:POP OVER STAT2 AND STAT3
2061	011116	052712	020000		31\$:	BIS	#BIT13,(R2)
2062	011122	104402			30\$:	TYPE	:SET BIT 13 IN STAT2 IF M8202
2063	011124	007114			CONN		
2064	011126	004737	012266		JSR	PC,INTTY	:ASK IF LOOP-BACK IS ON
2065	011132	022703	000131		CMP	#131,R3	:GET REPLY
2066	011136	001406			BEQ	17\$	:Y
2067	011140	022703	000116		CMP	#116,R3	:N
2068	011144	001406			BEQ	18\$	
2069	011146	104402			TYPE		
2070	011150	005666			MQM		:IF NOT Y OR N TYPE '?'
2071	011152	000763			BR	30\$	:TRY AGAIN
2072	011154	052722	040000		17\$:	BIS	#BIT14,(R2)+
2073	011160	000402			BR	19\$	:TURNAROUND IS CONNECTED
2074	011162	042722	040000		18\$:	BIC	#BIT14,(R2)+
2075	011166				19\$:		:NO TURNAROUND
2076	011166	104403			INSTR		
2077	011170	007016			LINE		
2078	011172	104405			PARAM		
2079	011174	000000			0		
2080	011176	000377			377		
2081	011200	001254			TEMP4		
2082	011202	000			.BYTE	0	
2083	011203	001			.BYTE	1	
2084	011204	113722	001254		MOVB	TEMP4,(R2)+	:STORE SWITCH PAC IN MAP
2085	011210	104403			INSTR		
2086	011212	007054			BM		
2087	011214	104405			PARAM		
2088	011216	000000			0		

2089	011220	000377			377		
2090	011222	001254			TEMP4		
2091	011224	000			.BYTE	0	
2092	011225	001			.BYTE	1	
2093	011226	113722	001254		MOVB	TEMP4,(R2)+	;STORE SWITCH PAC IN MAP
2094	011232	005722			TST	(R2)+	;POP OVER STAT3
2095	011234	005337	001252	33\$:	DEC	TEMP3	;DEC DMC COUNT
2096	011240	001402			BEQ	34\$	;BR IF DONE
2097	011242	000137	010612		JMP	12\$	;JUMP IF NOT
2098	011246	000137	011702	34\$:	JMP	13\$	;CONTINUE
2099	011252	012701	160000	7\$:	MOV	#160000,R1	;SET FOR FIRST ADDRESS TO BE TESTED
2100	011256	012737	011774	000004	MOV	#6\$,244	;SET FOR NON-EXISTANT DEVICE TIME OUT
2101	011264	005011		2\$:	CLR	(R1)	;CLEAR SEL0
2102	011266	005711			TST	(R1)	;IF DMC11 DMC SR S/B 0
2103	011270	001172			BNE	3\$	;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO DMC11
2104	011272	005061	000006		CLR	6(R1)	;CLEAR SEL6
2105	011276	005761	000006		TST	6(R1)	;IF DMC11 THEN DMRIC S/B 0.
2106	011302	001165			BNE	3\$	;BR IF NOT DMC11
2107	011304	012711	002000		MOV	#BIT10,(R1)	;SET ROM0
2108	011310	005061	000004		CLR	4(R1)	;CLEAR SEL4
2109	011314	012761	125252	000006	MOV	#125252,6(R1)	;WRITE THIS TO SEL6
2110	011322	052711	020000		BIS	#BIT13,(R1)	;WRITE IT!
2111	011326	022761	125252	000004	CMP	#125252,4(R1)	;WAS IT WRITTEN?
2112	011334	001004			BNE	21\$	;IF NO IT IS NOT CROM
2113	011336	052762	100000	000002	BIS	#BIT15,2(R2)	;SET BIT15 IF CROM
2114	011344	000431			BR	22\$	
2115	011346	012711	001000	21\$:	MOV	#BIT9,(R1)	;SET ROM1
2116	011352	012761	100430	000006	MOV	#100430,6(R1)	;PUT INSTRUCTION IN SEL6
2117	011360	012711	001400		MOV	#BIT9!BIT8,(R1)	;CLOCK INSTRUCTION (MICRO PROC PC TO 0)
2118	011364	012711	002000		MOV	#BIT10,(R1)	;SET ROM0
2119	011370	022761	016472	000006	CMP	#016472,6(R1)	;IS IT LOCAL CROM?
2120	011376	001411			BEQ	23\$	;BR IF YES
2121	011400	022761	016461	000006	CMP	#016461,6(R1)	;IS IT REMOTE CROM?
2122	011406	001410			BEQ	22\$	;BR IF YES
2123	011410	022761	177777	000006	CMP	#-1,6(R1)	;NO CROM?
2124	011416	001404			BEQ	22\$	;BR IF YES
2125	011420	000516			BR	3\$	;NOT A DMC
2126	011422	052762	000002	000006	23\$:	BIS	#BIT1,6(R2)
2127					22\$:	MOV	R1,(R2)+
2128	011430	010122			15\$:	MOV	#BIT9,(R1)
2129	011432	012711	001000		CLR	4(R1)	
2130	011436	005061	000004		MOV	#122113,6(R1)	
2131	011442	012761	122113	000006	BIS	#BIT8,(R1)	
2132	011450	052711	000400		MOV	#021264,6(R1)	
2133	011454	012761	021264	000006	BIS	#BIT8,(R1)	
2134	011462	052711	000400		CMPB	#377,4(R1)	
2135	011466	122761	000377	000004	BNE	10	
2136	011474	001003			BIS	#BIT12,(R2)	
2137	011476	052712	010000		BR	20\$	
2138	011502	000436			BIT	#BIT1,4(R1)	
2139	011504	032761	000002	000004	BEQ	10	
2140	011512	001403			BIS	#BIT13,BIT14,(R2)	
2141	011514	052712	060000		BR	20\$	
2142	011520	000427			BIT	#BIT13,4(R1)	
2143	011522	032761	000010	000004	BNE	20\$	
2144	011530	001023					

CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 44

CZDMEC.P11 11-SEP-78 12:38

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

SEQ 0044

2145	011532	012761	000100	000004		MOV	#BIT6,4(R1)	:LOAD PORT4
2146	011540	012761	122113	000006		MOV	#122113,6(R1)	:LOAD INSTRUCTION
2147	011546	052711	000400			BIS	#BIT8,(R1)	:CLOCK INSTRUCTION(SET DTR)
2148	011552	012761	021264	000006		MOV	#021264,6(R1)	:LOAD INSTRUCTION
2149	011560	052711	000400			BIS	#BIT8,(R1)	:CLOCK INSTRUCTION(READ MODEM REG)
2150	011564	032761	000010	000004		BIT	#BIT3,4(R1)	:IS MRDY SET NOW?
2151	011572	001402				BEQ	20\$	:BR IF NO CONNECTOR
2152	011574	052712	040000			BIS	#BIT14,(R2)	:SET STATUS BIT FOR CONNECTOR
2153	011600	005722			20\$:	TST	(R2)+	:POP POINTER
2154	011602	012761	021324	000006		MOV	#021324,6(R1)	:PUT INSTRUCTION IN PORT6
2155	011610	012711	001400			MOV	#BIT9!BIT8,(R1)	:PORT4 LU 15
2156	011614	156122	000004			BISB	4(R1),(R2)+	:STORE DDCMP LINE # IN TABLE
2157	011620	012761	021344	000006		MOV	#021344,6(R1)	:PORT6 INSTRUCTION
2158	011626	012711	001400			MOV	#BIT8!BIT9,(R1)	:CLOCK INSTR.
2159	011632	156122	000004			BISB	4(R1),(R2)+	:STORE BM873 ADD IN TABLE
2160	011636	005722				TST	(R2)+	:POP OVER STAT3
2161	011640	005011				CLR	(R1)	:CLEAR ROMI
2162	011642	005237	001310			INC	DMNUM	:UPDATE DEVICE COUNTER
2163	011646	022737	000020	001310		CMP	#20,DMNUM	:ARE MAX. NO. OF DEV FOUND?
2164	011654	001412				BEQ	13\$	:YES DON'T LOOK FOR ANY MORE.
2165	011656	005011			3\$:	CLR	(R1)	:CLEAR BIT 10
2166	011660	005061	000006			CLR	6(R1)	:CLEAR SEL 6
2167	011664	062701	000010		14\$:	ADD	#1),R1	:UPDATE CSR POINTER ADDRESS
2168	011670	022701	164000			CMP	#164000,R1	
2169	011674	001402				BEQ	13\$	:BR IF DONE
2170	011676	000137	011264			JMP	2\$	:JUMP IF NOT
2171	011702	005037	001306		13\$:	CLR	DMACTV	
2172	011706	005737	001310			TST	DMNUM	:WERE ANY DMC11'S FOUND AT ALL?
2173	011712	001423				BEQ	5\$	:ERROR AUTO SIZER FOUND NO DMC11'S IN THIS SYS.
2174	011714	013701	001310			MOV	DMNUM,R1	
2175	011720	010137	001314			MOV	R1,SAVNUM	:SAVE NUMBER OF DEVICES
2176	011724	000241			4\$:	CLC		
2177	011726	006137	001306			ROL	DMACTV	:GENERATE ACTIVE REGISTER OF DEVICES.
2178	011732	005237	001306			INC	DMACTV	:SET THE BIT
2179	011736	005301				DEC	R1	
2180	011740	001371				BNE	4\$	:BR IF MORE TO GENERATE
2181	011742	012737	000006	000004		MOV	#6,@#4	:RESTORE TRAP VECTOR
2182	011750	013737	001306	001322		MOV	DMACTV,SAVACT	:SAVE ACTIVE REGISTER
2183	011756	000137	012010			JMP	VECMAP	:GO FIND THE VECTOR NOW.
2184	011762	104402	005760		5\$:	TYPE	,MERR2	:NOTIFY OPR THAT NO DMC11'S FOUND.
2185	011766	005000				CLR	R0	:MAKE DATA LIGHTS ZERO
2186	011770	000000				HALT		:STOP THE SHOW
2187	011772	000776				BR	.-2	:DISABLE CONT. SW.
2188	011774	012716	011664		6\$:	MOV	#14\$,(SP)	:ENTERED BY NON-EXISTANT TIME-OUT.
2189	012000	000002				RTI		:RETURN TO MAINSTREAM
2190								
2191	012002	000001			WHICH:	1		
2192	012004	002	002			.BYTE	2,2	
2193	012006	001256				TEMP5		
2194								
2195	012010	032737	000001	001236	VECMAP:	BIT	#SW00,STRISW	
2196	012016	001114				BNE	5\$	
2197	012020	012737	000340	000022		MOV	#340,@#22	:SET IOT TRAP PRIO TO 7
2198	012026	012737	012202	000020		MOV	#4\$,@#20	:SET IOT TRAP VECTOR
2199	012034	012702	001500			MOV	#DM.MAP,R2	:SET SOFTWARE POINTER
2200	012040	012700	000300			MOV	#300,R0	:FLOATING VECTORS START HERE.

```

2201 012044 012701 000302      MOV      #302,R1      ;PC OF IOT INSTR.
2202 012050 010120      1$: MOV      R1,(R0)+      ;START FILLING VECTOR AREA
2203 012052 012721 000004      MOV      #4,(R1)+      ;WITH .+2; IOT
2204 012056 022021      CMP      (R0)+,(R1)+      ;ADD 2 TO R0 +R1
2205 012060 020127 001000      CMP      R1,#1000
2206 012064 101771      BLOS     1$      ;BR IF MORE TO FILL
2207 012066 013737 001306 001246      MOV      DMACTV,TEMP1      ;STORE TEMPORALLY
2208 012074 006037 001246      2$: ROR      TEMP1      ;BRING OUT A BIT
2209 012100 103063      BCC      5$      ;BR IF ALL DONE
2210 012102 012704 000012      MOV      #12,R4      ;R4 IS INDEX REGISTER
2211 012106 016437 012252 177776      MOV      BRLVL(R4),PS      ;SET PS TO 7
2212 012114 011201      MOV      (R2),R1
2213 012116 012761 000200 000004      MCV      #200,4(R1)
2214 012124 012711 001000      MOV      #BIT9,(R1)      ;SET ROMI
2215 012130 012761 121111 000006      MOV      #121111,6(R1)      ;PUT INSTRUCTION IN PORT6
2216 012136 012711 001400      MOV      #BIT9!BIT8,(R1)      ;FORCE AN INTERRUPT
2217 012142 105200      '5$: INCB     R0      ;STALL
2218 012144 001376      BNE      .-2      ;FOR TIME TO INTERUPT
2219 012146 162704 000002      SUB      #2,R4      ;GET NEXT LOWEST PS LEVEL
2220 012152 001404      BEQ      6$      ;BR IF R4 = 0
2221 012154 016437 012252 177776      MOV      BRLVL(R4),PS      ;MOVE NEXT LOWER LEVEL IN PS
2222 012162 000761      BR       7$      ;BR TO DELAY
2223 012164 052762 005300 000002 6$: BIS      #5300,2(R2)      ;NO INTERRUPT ASSUME 300 AT LEVEL 5 AND FIX DMC11 LATER
2224 012172 005011      3$: CLR      (R1)      ;CLEAR ROMI
2225 012174 062702 000010      ADD      #10,R2      ;POP SOFTWARE POINTER
2226 012200 000735      BR       2$      ;KEEP GOING
2227 012202 051662 000002 4$: BIS      (SP),2(R2)      ;GET VECTOR ADDRESS
2228 012206 042762 000007 000002      BIC      #7,2(R2)      ;CLEAR JUNK
2229 012214 016405 012254      MOV      BRLVL+2(R4),R5      ;GET BR LEVEL OF DMC11
2230 012220 006305      ASL      R5      ;SHIFT LEVEL 4 PLACES
2231 012222 006305      ASL      R5      ;TO THE LEFT FOR THE
2232 012224 006305      ASL      R5      ;STATUS TABLE
2233 012226 006305      ASL      R5
2234 012230 042705 170777      BIC      #170777,R5      ;CLEAR UNWANTED BITS
2235 012234 050562 000002      BIS      R5,2(R2)      ;PUT BR LEVEL IN STATUS TABLE
2236 012240 022626      CMP      (SP)+,(SP)+      ;POP IOT JUNK OFF STACK
2237 012242 012716 012172      MOV      #3$, (SP)      ;SET FOR RETURN
2238 012246 000002      RTI
2239 012250 000207      5$: RTS      PC      ;ALL DONE WITH 'AUTO SIZING'
2240
2241 012252 000000      BRLVL: 0      ;LEVEL 0
2242 012254 000000      0      ;LEVEL 0
2243 012256 000200      200      ;LEVEL 4
2244 012260 000240      240      ;LEVEL 5
2245 012262 000300      300      ;LEVEL 6
2246 012264 000340      340      ;LEVEL 7
2247
2248
2249 012266 105777 166712      INTTY: TSTB     @TKCSR      ;WAIT FOR DONE
2250 012272 100375      BPL      .-4
2251 012274 017703 166706      MOV      @TKDBR,R3      ;PUT CHAR IN R3
2252 012300 105777 166704      TSTB     @TPCSR      ;WAIT UNTIL PRINTER IS READY
2253 012304 100375      BPL      .-4
2254 012306 010377 166700      MOV      R3,@TPDBR      ;ECHO CHAR
2255 012312 042703 000240      BIC      #0'17.BITS,R3      ;MASK OFF LOWER CASE
2256 012316 000207      RTS      PC      ;RETURN

```

2257
 2258
 2259
 2260
 2261
 2262
 2263
 2264
 2265
 2266
 2267
 2268
 2269
 2270
 2271
 2272
 2273
 2274
 2275
 2276
 2277
 2278
 2279
 2280
 2281
 2282
 2283
 2284
 2285
 2286
 2287
 2288
 2289
 2290
 2291
 2292
 2293
 2294
 2295
 2296
 2297
 2298
 2299
 2300
 2301
 2302
 2303
 2304
 2305
 2306
 2307
 2308
 2309
 2310
 2311
 2312

```

***** TEST 1 *****
;*OUT CONTROL REGISTER READ/ONLY TEST
;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
;*BITS ARE IN THE CORRECT STATE
*****

: TEST 1
-----
TST1: MOV #1,TSTNO
      MOV #TST2,NEXT

      CLR @DMCSR           ;R1 CONTAINS BASE DMC11 ADDRESS
      MOV #11,R2          ;CLEAR SELO
      ROMCLK              ;SAVE R2 FOR TYPEOUT
      021004!<20*11>      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
      MOV 4(R1),R4         ;PORT4 LINE UNIT REG 11
      BIC #54,R4           ;PUT 'FOUND' IN R4
      MOV #20,R5           ;CLEAR UNKNOWN BITS
      CMPB R5,R4           ;PUT 'EXPECTED' IN R5
      BEQ 1$              ;IS OUT READY SET?
      HLT 2                ;BR IF YES
      SCOPE                ;ERROR IN LU 11
                           ;SCOPE THIS TEST

***** TEST 2 *****
;*IN CONTROL REGISTER READ/ONLY TEST
;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
;*BITS ARE IN THE CORRECT STATE
*****

: TEST 2
-----
TST2: MOV #2,TSTNO
      MOV #TST3,NEXT

      MOV #12,R2          ;R1 CONTAINS BASE DMC11 ADDRESS
      ROMCLK              ;SAVE R2 FOR TYPEOUT
      021004!<20*12>      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
      MOV 4(R1),R4         ;PORT4 LINE UNIT REG 12
      BIC #17,R4           ;PUT 'FOUND' IN R4
      CLR R5              ;CLEAR UNKNOWN BITS
      CMPB R5,R4           ;PUT 'EXPECTED' IN R5
      BEQ 1$              ;ARE ALL BITS CLEARED?
      HLT 2                ;BR IF YES
      SCOPE                ;ERROR IN LU 12
                           ;SCOPE THIS TEST

***** TEST 3 *****
;*MODEM CONTROL REGISTER READ/ONLY TEST
;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
;*BITS ARE IN THE CORRECT STATE
*****
  
```

```

2313
2314
2315      : TEST 3
2316      :-----
2316 012442 012737 000003 001226 TST3: MOV #3,TSTNO
2317 012450 012737 012514 001216 MOV #TST4,NEXT
2318
2319 012456 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
2320 012460 012702 000013 MOV #13,R2 ;MASTER CLEAR DMC11
2321 012464 104414 ROMCLK ;SAVE R2 FOR TYPEOUT
2322 012466 021264 021004!<20*13> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2323 012470 016104 000004 MOV 4(R1),R4 ;PORT4 LINE UNIT REG 13
2324 012474 042704 000213 BIC #213,R4 ;PUT 'FOUND' IN R4
2325 012500 012705 000100 MOV #100,R5 ;CLEAR UNKNOWN BITS
2326 012504 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
2327 012506 001401 BEQ 1$ ;ARE RING, DTR, AND MODEM READY SET?
2328 012510 104002 HLT 2 ;BR IF YES
2329 012512 104400 $: SCOPE ;ERROR IN LU 13
2330 ;SCOPE THIS TEST
2331
2332      :***** TEST 4 *****
2333      :*MAINTENANCE REGISTER READ/ONLY TEST
2334      :*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
2335      :*BITS ARE IN THE CORRECT STATE
2336      :*****
2337
2338      : TEST 4
2339      :-----
2340 012514 012737 000004 001226 TST4: MOV #4,TSTNO
2341 012522 012737 012616 001216 MOV #TST5,NEXT
2342
2343 012530 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
2344 012532 012702 000017 MOV #17,R2 ;MASTER CLEAR DMC11
2345 012536 104414 ROMCLK ;SAVE R2 FOR TYPEOUT
2346 012540 021364 021004!<20*17> ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2347 012542 016104 000004 MOV 4(R1),R4 ;PORT4 LINE UNIT REG 17
2348 012546 042704 000206 BIC #206,R4 ;PUT 'FOUND' IN R4
2349 012552 012705 000051 MOV #51,R5 ;CLEAR UNKNOWN BITS
2350 012556 032737 020000 001366 BIT #BIT13,STAT1 ;PUT 'EXPECTED' IN R5
2351 012564 001004 BNE 2$ ;IS LU AN M8202 OR M8201?
2352 012566 032737 040000 001366 BIT #BIT14,STAT1 ;BR IF M8202
2353 012574 001004 BNE 3$ ;CONNECTOR???
2354 012576 042704 000040 2$: BIC #40,R4 ;BR IF M8201 WITH CONNECTOR
2355 012602 042705 000040 BIC #BIT5,R5 ;MASK OFF SI BIT IF M8202 OR M8201, NO CONNECTOR
2356 012606 120504 3$: CMPB R5,R4 ;SI BIT IS UNKNOWN
2357 012610 001401 BEQ 1$ ;ARE SI AND ICIR SET?
2358 012612 104002 HLT 2 ;BR IF YES
2359 012614 104400 1$: SCOPE ;ERROR IN LU 17
2360 ;SCOPE THIS TEST
2361
2362      :***** TEST 5 *****
2363      :*LINE UNIT REGISTER WRITE/READ TEST
2364      :*SET BITS IN LU REGISTER 12, VERIFY IT IS SET
2365      :*CLEAR BITS IN LU REGISTER 12, VERIFY IT IS CLEAR
2366      :*****
2367
2368

```

```

2369      : TEST 5
2370      :-----
2371 012616 012737 000005 001226 TST5: MOV #5,TSTNO
2372 012624 012737 012756 001216 MOV #TST6,NEXT
2373 012632 012737 012646 001220 MOV #1$,LOCK
2374      :
2375 012640 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
2376 012642 012702 000012 MOV #12,R2 ;MASTER CLEAR DMC11
2377 012646 012761 000040 000004 1$: MOV #40,4(R1) ;SAVE REGISTER ADDRESS FOR TYPEOUT
2378 012654 104414 ROMCLK ;LOAD PORT4
2379 012656 122112 122112 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2380 012660 104414 ROMCLK ;SET BIT5 IN LU-12
2381 012662 021245 021245 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2382 012664 012705 000040 MOV #40,R5 ;READ LU-12
2383 012670 116104 000005 MOV 5(R1),R4 ;PUT 'EXPECTED' IN R5
2384 012674 042704 000337 BIC #337,R4 ;PUT 'FOUND' IN R4
2385 012700 120504 CMPB R5,R4 ;CLEAR UNWANTED BITS
2386 012702 001401 BEQ 2$ ;IS BIT5 SET?
2387 012704 104003 HLT 3 ;BR IF YES
2388 012706 104401 2$: SCOP1 ;ERROR, BIT 5 IS NOT SET
2389 012710 012737 012716 001220 3$: MOV #3$,LOCK ;SCOPE SUBTEST (SW09=1)
2390 012716 005061 000004 CLR 4(R1) ;NEW SCOP1
2391 012722 104414 ROMCLK ;LOAD PORT4
2392 012724 122112 122112 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2393 012726 104414 ROMCLK ;CLEAR BIT 5 IN LU-12
2394 012730 021245 021245 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2395 012732 005005 CLR R5 ;READ LU-12
2396 012734 116104 000005 MOV 5(R1),R4 ;PUT 'EXPECTED' IN R5
2397 012740 042704 000337 BIC #337,R4 ;PUT 'FOUND' IN R4
2398 012744 120504 CMPB R5,R4 ;CLEAR UNWANTED BITS
2399 012746 001401 BEQ 4$ ;IS BIT5 CLEAR?
2400 012750 104003 HLT 3 ;BR IF YES
2401 012752 104401 4$: SCOP1 ;ERROR, BIT5 IS NOT CLEAR
2402 012754 104400 SCOPE ;SCOPE SUBTEST (SW09=1)
2403      :
2404      :
2405      :***** TEST 6 *****
2406      :*LINE UNIT REGISTER WRITE/READ TEST
2407      :*SET BIT1 IN LU REGISTER 17, VERIFY IT IS SET
2408      :*CLEAR BIT1 IN LU REGISTER 17, VERIFY IT IS CLEAR
2409      :*****
2410      :
2411      : TEST 6
2412      :-----
2413 012756 012737 000006 001226 TST6: MOV #6,TSTNO
2414 012764 012737 013116 001216 MOV #TST7,NEXT
2415 012772 012737 013006 001220 MOV #1$,LOCK
2416      :
2417 013000 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
2418 013002 012702 000017 MOV #17,R2 ;MASTER CLEAR DMC11
2419 013006 012761 000001 000004 1$: MOV #1,4(R1) ;SAVE REGISTER ADDRESS FOR TYPEOUT
2420 013014 104414 ROMCLK ;LOAD PORT4
2421 013016 122117 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2422 013020 104414 ROMCLK ;SET BIT1 IN LU-17
2423 013022 021365 021365 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2424 013024 012705 000001 MOV #1,R5 ;READ LU-17
      :PUT 'EXPECTED' IN R5

```

2425	013030	116104	000005			MOVB	5(R1),R4	:PUT 'FOUND' IN R4
2426	013034	042704	000376			BIC	#376,R4	:CLEAR UNWANTED BITS
2427	013040	120504				CMPB	R5,R4	:IS BIT1 SET?
2428	013042	001401				BEQ	2\$	:BR IF YES
2429	013044	104003				HLT	3	:ERROR, BIT 1 IS NOT SET
2430	013046	104401			2\$:	SCOP1		:SCOPE SUBTEST (SW09=1)
2431	013050	012737	013056	001220		MOV	#3\$,LOCK	:NEW SCOP1
2432	013056	005061	000004		3\$:	CLR	4(R1)	:LOAD PORT4
2433	013062	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2434	013064	122117				122117		:CLEAR BIT 1 IN LU-17
2435	013066	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2436	013070	021365				021365		:READ LU-17
2437	013072	005005				CLR	R5	:PUT 'EXPECTED' IN R5
2438	013074	116104	000005			MOVB	5(R1),R4	:PUT 'FOUND' IN R4
2439	013100	042704	000376			BIC	#376,R4	:CLEAR UNWANTED BITS
2440	013104	120504				CMPB	R5,R4	:IS BIT1 CLEAR?
2441	013106	001401				BEQ	4\$	:BR IF YES
2442	013110	104003				HLT	3	:ERROR, BIT1 IS NOT CLEAR
2443	013112	104401			4\$:	SCOP1		:SCOPE SUBTEST (SW09=1)
2444	013114	104400				SCOPE		:SCOPE THIS TEST
2445								
2446								
2447								
2448								
2449								
2450								
2451								
2452								
2453								
2454								
2455	013116	012737	000007	001226	TST7:	MOV	#7,TSTNO	
2456	013124	012737	013326	001216		MOV	#TST10,NEXT	
2457	013132	012737	013152	001220		MOV	#64\$,LOCK	
2458								
2459	013140	104412				MSTCLR		:R1 CONTAINS BASE DMC11 ADDRESS
2460	013142	012702	000013			MOV	#13,R2	:MASTER CLEAR DMC11
2461	013146	012700	000001			MOV	#1,R0	:SAVE REGISTER ADDRESS FOR TYPEOUT
2462	013152				64\$:			:START WITH BIT 0
2463	013152	010061	000004			MOV	R0,4(R1)	:PUT PATTERN INTO PORT4
2464	013156	042761	000257	000004		BIC	#257,4(R1)	:CLEAR UNWANTED BITS
2465	013164	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2466	013166	122113				122100!13		:MOV DATA TO IBUS REGISTER 13
2467	013170	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2468	013172	021265				21005!<13*20>		:READ FROM IBUS REGISTER 13
2469	013174	010005				MOV	R0,R5	:PUT EXPECTED IN R5
2470	013176	042705	000257			BIC	#257,R5	:CLEAR UNWANTED BITS
2471	013202	116104	000005			MOVB	5(R1),R4	:PUT 'FOUND' INTO R4
2472	013206	042704	000257			BIC	#257,R4	:CLEAR UNWANTED BITS
2473	013212	120504				CMPB	R5,R4	:DATA CORRECT?
2474	013214	001401				BEQ	65\$	:BR IF YES
2475	013216	104003				HLT	3	:ERROR
2476	013220	104401			65\$:	SCOP1		:SW09=1?
2477	013222	000241				CLC		:CLEAR CARRY
2478	013224	106100				ROLB	R0	:SHIFT BIT IN R0
2479	013226	001351				BNE	64\$	:IF R0=0 THEN DONE
2480	013230	012737	013244	001220		MOV	#67\$,LOCK	:NEW SCOP1

***** TEST 7 *****
 :*LINE UNIT REGISTER WRITE/READ TEST
 :*FLOAT A 1 THROUGH LINE UNIT REGISTER 13
 :*FLOAT A 0 THROUGH LINE UNIT REGISTER 13
 :*****

: TEST 7

2481	013236	012700	000001		MOV	#1,R0	:START WITH BIT 0
2482	013242	005100		69\$:	COM	R0	:CHANGE TO FLOATING ZERO
2483	013244			67\$:			
2484	013244	010061	000004		MOV	R0,4(R1)	:PUT PATTERN INTO PORT4
2485	013250	042761	000257	000004	BIC	#257,4(R1)	:CLEAR UNWANTED BITS
2486	013256	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2487	013260	122113			122100:13		:MOV DATA TO IBUS REGISTER 13
2488	013262	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2489	013264	021265			21005!<13*20>		:READ FROM IBUS REGISTER 13
2490	013266	010005			MOV	R0,R5	:PUT EXPECTED IN R5
2491	013270	042705	000257		BIC	#257,R5	:CLEAR UNWANTED BITS
2492	013274	116104	000005		MOVB	5(R1),R4	:PUT 'FOUND' INTO R4
2493	013300	042704	000257		BIC	#257,R4	:CLEAR UNWANTED BITS
2494	013304	120504			CMPB	R5,R4	:DATA CORRECT?
2495	013306	001401			BEQ	68\$	:BR IF YES
2496	013310	104003			HLT	3	:ERROR
2497	013312	104401		68\$:	SCOP1		:SW09=1?
2498	013314	005100			COM	R0	:CHANGE TO FLOATING 1
2499	013316	000241			CLC		:CLEAR CARRY
2500	013320	106100			ROLB	R0	:SHIFT BIT IN R0
2501	013322	001347			BNE	69\$	:IF R0=0 THEN DONE
2502	013324	104400			SCOPE		:SCOPE THIS TEST

2503
 2504
 2505
 2506
 2507
 2508
 2509
 2510
 2511
 2512
 2513
 2514
 2515
 2516
 2517
 2518
 2519
 2520
 2521
 2522
 2523
 2524
 2525
 2526
 2527
 2528
 2529
 2530
 2531
 2532
 2533
 2534
 2535
 2536

:***** TEST 10 *****
 :*LINE UNIT REGISTER WRITE/READ TEST
 :*FLOAT A 1 THROUGH LINE UNIT REGISTER 14
 :*FLOAT A 0 THROUGH LINE UNIT REGISTER 14
 :*****

: TEST 10

2513	013326	012737	000010	001226	TST10:	MOV	#10,TSTNO	
2514	013334	012737	013502	001216		MOV	#TST11,NEXT	
2515	013342	012737	013362	001220		MOV	#64\$,LOCK	
2516								:R1 CONTAINS BASE DMC11 ADDRESS
2517	013350	104412				MSTCLR		:MASTER CLEAR DMC11
2518	013352	012702	000014			MOV	#14,R2	:SAVE REGISTER ADDRESS FOR TYPEOUT
2519	013356	012700	000001			MOV	#1,R0	:START WITH BIT 0
2520	013362				64\$:			
2521	013362	010061	000004			MOV	R0,4(R1)	:PUT PATTERN INTO PORT4
2522	013366	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2523	013370	122114				122100:14		:MOV DATA TO IBUS REGISTER 14
2524	013372	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2525	013374	021305				21005!<14*20>		:READ FROM IBUS REGISTER 14
2526	013376	010005				MOV	R0,R5	:PUT EXPECTED IN R5
2527	013400	116104	000005			MOVB	5(R1),R4	:PUT 'FOUND' INTO R4
2528	013404	120504				CMPB	R5,R4	:DATA CORRECT?
2529	013406	001401				BEQ	65\$	:BR IF YES
2530	013410	104003				HLT	3	:ERROR
2531	013412	104401			65\$:	SCOP1		:SW09=1?
2532	013414	000241				CLC		:CLEAR CARRY
2533	013416	106100				ROLB	R0	:SHIFT BIT IN R0
2534	013420	001360				BNE	64\$	:IF R0=0 THEN DONE
2535	013422	012737	013436	001220		MOV	#64\$,LOCK	:NEW SCOP1
2536	013430	012700	000001			MOV	#1,R0	:START WITH BIT 0

```

2537 013434 005100          69$: COM      R0          ;CHANGE TO FLOATING ZERO
2538 013436          67$:          ;
2539 013436 010061 000004    MOV      R0,4(R1)      ;PUT PATTERN INTO PORT4
2540 013442 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2541 013444 122114          122100.14          ;MOV DATA TO IBUS REGISTER 14
2542 013446 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2543 013450 021305          21005!<14*20>      ;READ FROM IBUS REGISTER 14
2544 013452 010005          MOV      R0,R5          ;PUT EXPECTED IN R5
2545 013454 116104 000005    MOV      5(R1),R4      ;PUT 'FOUND' INTO R4
2546 013460 120504          CMP      R5,R4          ;DATA CORRECT?
2547 013462 001401          BEQ      68$          ;BR IF YES
2548 013464 104003          HLT      3              ;ERROR
2549 013466 104401          68$: SCOP1          ;SW09=1?
2550 013470 005100          COM      R0          ;CHANGE TO FLOATING 1
2551 013472 000241          CLC              ;CLEAR CARRY
2552 013474 106100          ROL      R0          ;SHIFT BIT IN R0
2553 013476 001356          BNE      69$          ;IF R0=0 THEN DONE
2554 013500 104400          SCOPE          ;SCOPE THIS TEST
2555
2556
2557          ;***** TEST 11 *****
2558          ;*SWITCH PAC TEST
2559          ;*THIS TEST READS SWITCH PAC#1
2560          ;*THIS SWITCH PAC CONTAINS THE DDCMP LINE #
2561          ;*****
2562
2563          ; TEST 11
2564          ;-----
2565 013502 012737 000011 001226 TST11: MOV      #11,TSTNO
2566 013510 012737 013544 001216    MOV      #TST12,NEXT
2567
2568 013516 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
2569 013520 104414          ROMCLK          ;MASTER CLEAR DMC11
2570 013522 021324          021324          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2571 013524 016104 000004    MOV      4(R1),R4      ;PORT4_LU15
2572 013530 113705 001370    MOV      STAT2,R5      ;PUT 'FOUND' IN R4
2573 013534 120504          CMP      R5,R4          ;PUT 'EXPECTED' IN R5
2574 013536 001401          BEQ      1$          ;SW OK?
2575 013540 104031          HLT      31          ;BR IF YES
2576 013542 104400          1$: SCOPE          ;ERROR, SWITCH PAC READ ERROR
2577          ;SCOPE THIS TEST
2578
2579          ;***** TEST 12 *****
2580          ;*SWITCH PAC TEST
2581          ;*THIS TEST READS SWITCH PAC#2
2582          ;*THIS SWITCH PAC CONTAINS THE BM873 BOOT ADD
2583          ;*****
2584
2585          ; TEST 12
2586          ;-----
2587 013544 012737 000012 001226 TST12: MOV      #12,TSTNO
2588 013552 012737 013606 001216    MOV      #TST13,NEXT
2589
2590 013560 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
2591 013562 104414          ROMCLK          ;MASTER CLEAR DMC11
2592 013564 021344          021344          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
          ;PORT4_LU16

```

LINE UNIT WRITE/READ TESTS

[illegible]

CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 53
 CZDMEC.P11 11-SEP-78 12:38 BASIC TRANSMITTER TESTS

SEQ 0053

```

2649 013736 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2650 013740 122111 122111 ;SET SOM
2651 013742 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2652 013744 122110 122110 ;LOAD OUT DATA SILO
2653 013746 104416 000002 TIMER, 2 ;WAIT FOR OCOR
2654 013752 012702 000017 MOV #17,R2 ;SAVE ADDRESS FOR TYPEOUT
2655 013756 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2656 013760 021364 021364 ;PORT4 LU 17
2657 013762 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
2658 013766 042704 000357 BIC #357,R4 ;CLEAR UNWANTED BITS
2659 013772 012705 000020 MOV #20,R5 ;PUT 'EXPECTED' IN R5
2660 013776 120504 CMPB R5,R4 ;IS OCOR SET?
2661 014000 001401 BEQ 1$ ;BR IF YES
2662 014002 104005 HLT 5
2663 014004
2664 014004 104400 1$: SCOPE ;SCOPE THIS TEST
2665
2666
2667 ***** TEST 15 *****
2668 ;*DDCMP TEST OF RTS AND OUT ACTIVE
2669 ;*SET SOM AND LOAD OUT DATA SILO
2670 ;*SINGLE STEP 2 DATA CLOCKS, VERIFY
2671 ;*THAT RTS AND ACTIVE ARE SET
2672 ;*****
2673
2674 : TEST 15
2675 :-----
2676 014006 012737 000015 001226 TST15: MOV #15,TSTNO
2677 014014 012737 014144 001216 MOV #TST16,NEXT
2678
2679 014022 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
2680 014024 012711 004000 MOV #BIT11,(R1) ;MASTER CLEAR DMC11
2681 014030 012761 000001 000004 MOV #1,4(R1) ;SET LINE UNIT LOOP
2682 014036 104414 ROMCLK ;LOAD PORT4 WITH BIT0
2683 014040 122111 122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2684 014042 104414 ROMCLK ;SET SOM
2685 014044 122110 122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2686 014046 004737 026374 JSR PC,OCOR ;LOAD OUT DATA SILO
2687 014052 104415 000002 DATACLK, 2 ;WAIT FOR OCOR
2688 014056 012702 000011 MOV #11,R2 ;CLOCK DATA FOUR TIMES
2689 014062 104414 ROMCLK ;SAVE ADDRESS FOR TYPEOUT
2690 014064 021224 021224 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2691 014066 016104 000004 MOV 4(R1),R4 ;PORT4 LU 11
2692 014072 042704 000257 BIC #257,R4 ;PUT 'FOUND' IN R4
2693 014076 012705 000120 MOV #120,R5 ;CLEAR UNWANTED BITS
2694 014102 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
2695 014104 001401 BEQ 1$ ;IS ACTIVE SET?
2696 014106 104005 HLT 5 ;BR IF YES
2697 014110
2698 014110 012702 000013 1$: MOV #13,R2 ;SAVE ADDRESS FOR TYPEOUT
2699 014114 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2700 014116 021264 021264 ;PORT4 LU 13
2701 014120 016104 000004 MOV 4(R1),R4 ;PUT EXPECTED IN R4
2702 014124 042704 000337 BIC #337,R4 ;CLEAR UNWANTED BITS
2703 014130 012705 000040 MOV #215,R5 ;PUT 'EXPECTED' IN R5, RTS SHOULD BE SET
2704 014134 120504 CMPB R5,R4 ;IS RTS OK?

```

```

2705 014136 001401      BEQ      2$      ;BR IF YES
2706 014140 104005      HLT      5      ;RTS ERROR
2707 014142      2$:
2708 014142 104400      SCOPE      ;SCOPE THIS TEST
2709
2710
2711      ;***** TEST 16 *****
2712      ;*TEST OF OUT CLEAR
2713      ;*SET SOM AND LOAD OUT DATA SILO
2714      ;*SINGLE STEP DATA CLOCK, SET OUT CLEAR
2715      ;*VERIFY THAT OCOR,RTS, AND ACTIVE ARE CLEARED
2716      ;*****
2717
2718      ; TEST 16
2719      ;-----
2720 014144 012737 000016 001226 TST16: MOV      #16,TSTNO
2721 014152 012737 014342 001216      MOV      #TST17,NEXT
2722
2723      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
2724 014162 012711 004000      MOV      #BIT11,(R1) ;MASTER CLEAR DMC11
2725 014166 012761 000001 000004      MOV      #1,4(R1) ;SET LINE UNIT LOOP
2726 014174 104414      ROMCLK      ;LOAD PORT4 WITH BIT0
2727 014176 122111      122111      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2728 014200 104414      ROMCLK      ;SET SOM
2729 014202 122110      122110      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2730 014204 004737 026374      JSR      PC,OCOR ;LOAD OUT DATA SILO
2731 014210 104415 000002      DATACLK, 2 ;WAIT FOR OCOR
2732 014214 012761 000200 000004      MOV      #BIT7,4(R1) ;CLOCK DATA FOUR TIMES
2733 014222 104414      ROMCLK      ;SET BIT7 IN PORT4
2734 014224 122111      122111      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2735 014226 104415 000001      DATACLK, 1 ;SET OUT CLEAR
2736 014232 012702 000017      MOV      #17,R2 ;GIVE A TICK TO CLEAR RTS
2737 014236 104414      ROMCLK      ;SAVE ADDRESS FOR TYPEOUT
2738 014240 021364      021364      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2739 014242 016104 000004      MOV      4(R1),R4 ;PORT4 LU 17
2740 014246 042704 000357      BIC      #357,R4 ;PUT 'FOUND' IN R4
2741 014252 005005      CLR      R5 ;CLEAR UNWANTED BITS
2742 014254 120504      CMPB      R5,R4 ;PUT 'EXPECTED' IN R5
2743 014256 001401      BEQ      1$ ;IS OCOR CLEARED?
2744 014260 104005      HLT      5 ;BR IF YES
2745 014262
2746 014262 012702 000013      1$: MOV      #13,R2 ;SAVE ADDRESS FOR TYPEOUT
2747 014266 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2748 014270 021264      021264      ;PORT4 LU 13
2749 014272 016104 000004      MOV      4(R1),R4 ;PUT EXPECTED IN R4
2750 014276 042704 000337      BIC      #337,R4 ;CLEAR UNWANTED BITS
2751 014302 005005      CLR      R5 ;PUT 'EXPECTED' IN R5, RTS SHOULD BE CLEARED
2752 014304 120504      CMPB      R5,R4 ;IS RTS OK?
2753 014306 001401      BEQ      2$ ;BR IF YES
2754 014310 104005      HLT      5 ;RTS ERROR
2755 014312
2756 014312 012702 000011      2$: MOV      #11,R2 ;SAVE ADDRESS FOR TYPEOUT
2757 014316 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2758 014320 021224      021224      ;PORT4 LU11
2759 014322 016104 000004      MOV      4(R1),R4 ;PUT 'FOUND' IN R4
2760 014326 012705 000020      MOV      #BIT4,R5 ;ONLY OUT READY SHOULD BE SET
  
```

```

2761 014332 120504      (MPB    R5,R4      ;IS ACTIVE CLEAR?
2762 014334 001401      BEQ      3$      ;BR IF YES
2763 014336 104005      HLT      5      ;ERROR ACTIVE NOT CLEARED
2764 014340      3$:      SCOPE      ;SCOPE THIS TEST
2765 014340 104400
2766
2767
2768      ;***** TEST 17 *****
2769      ;DDCMP TRANSMITTER TEST
2770      ;SINGLE CLOCK THE CHARACTER 0
2771      ;VERIFY EACH BIT POSITION AS IT
2772      ;PASSES THE BIT WINDOW (SI BIT)
2773      ;ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2774      ;*****
2775
2776      ; TEST 17
2777      ;-----
2778 014342 012737 000017 001226 TST17: MOV      #17,TSTNO
2779 014350 012737 014524 001216      MOV      #TST20,NEXT
2780
2781 014356 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
2782 014360 012711 004000      MOV      #BIT11,(R1) ;MASTER CLEAR DMC11
2783 014364 004737 026526      JSR      PC,OUTRDY ;SET LINE UNIT LOOP
2784 014370 012761 000001 000004      MOV      #1,4(R1) ;WAIT FOR OUT-READY
2785 014376 104414      ROMCLK      ;SET BIT0 IN PORT4
2786 014400 122111      122111      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2787 014402 012705 000000      MOV      #0,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
2788 014406 004737 026526      JSR      PC,OUTRDY ;SET SOM!
2789 014412 010561 000004      MOV      R5,4(R1) ;WAIT FOR OUT-READY
2790 014416 104414      ROMCLK      ;LOAD PORT4 WITH CHARACTER
2791 014420 122110      122110      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2792 014422 004737 026374      JSR      PC,OCOR ;LOAD OUT DATA
2793 014426 005003      CLR      R3 ;WAIT FOR OCOR TO SET
2794 014430 010502      MOV      R5,R2 ;CLEAR BIT COUNTER
2795 014432 104415 000002      DATACLK, 2 ;LOAD CHARACTER IN R2
2796 014436 104415 000001 1$: DATACLK, 1 ;2 TICKS TO SET UP TRANSMITTER
2797 014442 106002      RORB      R2 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2798 014444 103005      BCC      2$ ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2799 014446 004737 026342      JSR      PC,GETSI ;BR IF CARRY CLEAR
2800 014452 103406      BCS      3$ ;GET THE WINDOW
2801 014454 104006      HLT      6 ;BR IF BIT IS A MARK
2802 014456 000404      BR      3$ ;ERROR BIT WAS A SPACE
2803 014460 004737 026342 2$: JSR      PC,GETSI ;CONTINUE WITH TEST
2804 014464 103001      BCC      3$ ;GET THE WINDOW
2805 014466 104006      HLT      6 ;BR IF BIT IS A SPACE
2806 014470      3$:      ;ERROR BIT WAS A MARK
2807 014470 005203      INC      R3 ;NEXT BIT
2808 014472 022703 000010      CMP      #10,R3 ;DONE YET?
2809 014476 001357      BNE      1$ ;BR IF NO
2810 014500 104415 000014      DATACLK, 14 ;CLOCK TRANSMITTER 14 MORE TICKS
2811 014504 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2812 014506 021264      021264      ;PORT4 LU-13
2813 014510 032761 000040 000004      BJT      #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
2814 014516 001401      BEQ      4$ ;BR IF YES
2815 014520 104034      HLT      3$ ;ERROR, RTS NOT CLEAR
2816 014522 104400      4$:      SCOPE ;SCOPE THIS TEST

```

2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872

```

***** TEST 20 *****
*DDCMP TRANSMITTER TEST
*SINGLE CLOCK THE CHARACTER 125
*VERIFY EACH BIT POSITION AS IT
*PASSES THE BIT WINDOW (SI BIT)
*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
*****

: TEST 20
-----
TST20: MOV #20,TSTNO
MOV #TST21,NEXT

MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
MOV #BIT11,(R1) ;MASTER CLEAR DMC11
JSR PC,OUTRDY ;SET LINE UNIT LOOP
MOV #1,4(R1) ;WAIT FOR OUT-READY
ROMCLK ;SET BIT0 IN PORT4
122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET SOM!
MOV #125,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
JSR PC,OUTRDY ;WAIT FOR OUT-READY
MOV R5,4(R1) ;LOAD PORT4 WITH CHARACTER
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110 ;LOAD OUT DATA
JSR PC,OCOR ;WAIT FOR OCOR TO SET
CLR R3 ;CLEAR BIT COUNTER
MOV R5,R2 ;LOAD CHARACTER IN R2
DATACLK, 2 ;2 TICKS TO SET UP TRANSMITTER
DATACLK, 1 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
RORB R2 ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
BCC 2$ ;BR IF CARRY CLEAR
JSR PC,GETSI ;GET THE WINDOW
BCS 3$ ;BR IF BIT IS A MARK
HLT 6 ;ERROR BIT WAS A SPACE
BR 3$ ;CONTINUE WITH TEST
JSR PC,GETSI ;GET THE WINDOW
BCC 3$ ;BR IF BIT IS A SPACE
HLT 6 ;ERROR BIT WAS A MARK

3$: INC R3 ;NEXT BIT
CMP #10,R3 ;DONE YET?
BNE 1$ ;BR IF NO
DATACLK, 14 ;CLOCK TRANSMITTER 14 MORE TICKS
ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021264 ;PORT4 LU-13
BIT #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
BEQ 4$ ;BR IF YES
HLT 34 ;ERROR, RTS NOT CLEAR
4$: SCOPE ;SCOPE THIS TEST

***** TEST 21 *****
*DDCMP TRANSMITTER TEST
*SINGLE CLOCK THE CHARACTER 252

```

CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 57
 CZDMEC.P11 11-SEP-78 12:38 BASIC TRANSMITTER TESTS

SEQ 0057

```

2873      ;*VERIFY EACH BIT POSITION AS IT
2874      ;*PASSES THE BIT WINDOW (SI BIT)
2875      ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2876      ;*****
2877
2878      ; TEST 21
2879      ;-----
2880 014706 012737 000021 001226 TST21: MOV #21,TSTNO
2881 014714 012737 015070 001216 MOV #TST22,NEXT
2882
2883 014722 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
2884 014724 012711 004000 MOV #BIT11,(R1) ;MASTER CLEAR DMC11
2885 014730 004737 026526 JSR PC,OUTRDY ;SET LINE UNIT LOOP
2886 014734 012761 000001 000004 MOV #1,4(R1) ;WAIT FOR OUT-READY
2887 014742 104414 ROMCLK ;SET BIT0 IN PORT4
2888 014744 122111 122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2889 014746 012705 000252 MOV #252,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
2890 014752 004737 026526 JSR PC,OUTRDY ;WAIT FOR OUT-READY
2891 014756 010561 000004 MOV R5,4(R1) ;LOAD PORT4 WITH CHARACTER
2892 014762 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2893 014764 122110 122110 ;LOAD OUT DATA
2894 014766 004737 026374 JSR PC,OCOR ;WAIT FOR OCOR TO SET
2895 014772 005003 CLR R3 ;CLEAR BIT COUNTER
2896 014774 010502 MOV R5,R2 ;LOAD CHARACTER IN R2
2897 014776 104415 000002 DATACLK, 2 ;2 TICKS TO SET UP TRANSMITTER
2898 015002 104415 000001 1$: DATACLK, 1 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2899 015006 106002 RORB R2 ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2900 015010 103005 BCC 2$ ;BR IF CARRY CLEAR
2901 015012 004737 026342 JSR PC,GETSI ;GET THE WINDOW
2902 015016 103406 BCS 3$ ;BR IF BIT IS A MARK
2903 015020 104006 HLT 6 ;ERROR BIT WAS A SPACE
2904 015022 000404 BR 3$ ;CONTINUE WITH TEST
2905 015024 004737 026342 2$: JSR PC,GETSI ;GET THE WINDOW
2906 015030 103001 BCC 3$ ;BR IF BIT IS A SPACE
2907 015032 104006 HLT 6 ;ERROR BIT WAS A MARK
2908 015034 3$:
2909 015034 005203 INC R3 ;NEXT BIT
2910 015036 022703 000010 CMP #10,R3 ;DONE YET?
2911 015042 001357 BNE 1$ ;BR IF NO
2912 015044 104415 000014 DATACLK, 14 ;CLOCK TRANSMITTER 14 MORE TICKS
2913 015050 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2914 015052 021264 021264 ;PORT4 LU-13
2915 015054 032761 000040 000004 BIT #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
2916 015062 001401 BEQ 4$ ;BR IF YES
2917 015064 104034 HLT 34 ;ERROR, RTS NOT CLEAR
2918 015066 104400 4$: SCOPE ;SCOPE THIS TEST
2919
2920
2921      ;*****~***** TEST 22 *****
2922      ;*DDCMP TRANSMITTER TEST
2923      ;*SINGLE CLOCK THE CHARACTER 377
2924      ;*VERIFY EACH BIT POSITION AS IT
2925      ;*PASSES THE BIT WINDOW (SI BIT)
2926      ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2927      ;*****
2928

```

```

2929      : TEST 22
2930      :-----
2931 015070 012737 000022 001226 TST22: MOV #22,TSTNO
2932 015076 012737 015252 001216      MOV #TST23,NEXT
2933
2934 015104 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
2935 015106 012711 004000      MOV #BIT11,(R1) ;MASTER CLEAR DMC11
2936 015112 004737 026526      JSR PC,OUTRDY ;SET LINE UNIT LOOP
2937 015116 012761 000001 000004      MOV #1,4(R1) ;WAIT FOR OUT-READY
2938 015124 104414      ROMCLK      ;SET BIT0 IN PORT4
2939 015126 122111      122111      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2940 015130 012705 000377      MOV #377,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
2941 015134 004737 026526      JSR PC,OUTRDY ;SET SOM!
2942 015140 010561 000004      MOV R5,4(R1) ;WAIT FOR OUT-READY
2943 015144 104414      ROMCLK      ;LOAD PORT4 WITH CHARACTER
2944 015146 122110      122110      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2945 015150 004737 026374      JSR PC,OCOR ;LOAD OUT DATA
2946 015154 005003      CLR R3 ;WAIT FOR OCOR TO SET
2947 015156 010502      MOV R5,R2 ;CLEAR BIT COUNTER
2948 015160 104415 000002      DATACLK, 2 ;LOAD CHARACTER IN R2
2949 015164 104415 000001      DATACLK, 1 ;2 TICKS TO SET UP TRANSMITTER
2950 015170 106002      RORB R2 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2951 015172 103005      BCC 2$ ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2952 015174 004737 026342      JSR PC,GETSI ;BR IF CARRY CLEAR
2953 015200 103406      BCS 3$ ;GET THE WINDOW
2954 015202 104006      HLT 6 ;BR IF BIT IS A MARK
2955 015204 000404      BR 3$ ;ERROR BIT WAS A SPACE
2956 015206 004737 026342      JSR PC,GETSI ;CONTINUE WITH TEST
2957 015212 103001      BCC 3$ ;GET THE WINDOW
2958 015214 104006      HLT 6 ;BR IF BIT IS A SPACE
2959 015216      3$: ;ERROR BIT WAS A MARK
2960 015216 005203      INC R3 ;NEXT BIT
2961 015220 022703 000010      CMP #10,R3 ;DONE YET?
2962 015224 001357      BNE 1$ ;BR IF NO
2963 015226 104415 000014      DATACLK, 14 ;CLOCK TRANSMITTER 14 MORE TICKS
2964 015232 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2965 015234 021264      021264      ;PORT4 LU-13
2966 015236 032761 000040 000004      BIT #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
2967 015244 001401      BEQ 4$ ;BR IF YES
2968 015246 104034      HLT 34 ;ERROR, RTS NOT CLEAR
2969 015250      4$: SCOPE ;SCOPE THIS TEST

```

```

2970
2971
2972      :***** TEST 23 *****
2973      :*DDCMP TRANSMITTER TEST
2974      :*SINGLE CLOCK A BINARY COUNT PATTERN
2975      :*VERIFY EACH BIT POSITION AS IT
2976      :*PASSES THE BIT WINDOW (SI BIT)
2977      :*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2978      :*AND R5 CONTAINS THE CHARACTER THAT FAILED
2979      :*****
2980
2981      : TEST 23
2982      :-----
2983 015252 012737 000023 001226 TST23: MOV #23,TSTNO
2984 015260 012737 015460 001216      MOV #TST24,NEXT

```


CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 60
 CZDMEC.P11 11-SEP-78 12:38 BASIC RECEIVER TESTS

SEQ 0060

```

3041 015460 012737 000024 001226 TST24: MOV #24,TSTNO
3042 015466 012737 015546 001216 MOV #TST25,NEXT
3043
3044 015474 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3045 015476 012711 004000 MOV #BIT11,(R1) ;MASTER CLEAR DMC11
3046 015502 012702 000012 MOV #12,R2 ;SET LU LOOP
3047 015506 004737 026412 JSR PC,SYNC ;SAVE LU REG FOR TYPEOUT
3048 015512 000005 5 ;SINGLE CLOCK 5 SYNC CHARACTERS
3049 015514 104415 000054 DATACLK, 54
3050 015520 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3051 015522 021244 021244 ;PORT4 LU12
3052 015524 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3053 015530 042704 000277 BIC #277,R4 ;CLEAR UNWANTED BITS
3054 015534 005005 CLR R5 ;PUT 'EXPECTED' IN R5
3055 015536 120504 CMPB R5,R4 ;IS ACTIVE CLEAR?
3056 015540 001401 BEQ 1$ ;BR IF YES
3057 015542 104040 HLT 40 ;ERROR ACTIVE IS NOT CLEAR
3058 015544 104400 1$: SCOPE ;SCOPE THIS TEST
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068

```

```

:***** TEST 25 *****
:*DDCMP IN ACTIVE TEST
:*SET LU LOOP, SINGLE STEP 5 SYNC AND A NON-SYNC (301)
:*VERIFY THAT IN ACTIVE IS SET
:*****

```

```

: TEST 25
:-----

```

```

3069 015546 012737 000025 001226 TST25: MOV #25,TSTNO
3070 015554 012737 015636 001216 MOV #TST26,NEXT
3071
3072 015562 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3073 015564 012711 004000 MOV #BIT11,(R1) ;MASTER CLEAR DMC11
3074 015570 012702 000012 MOV #12,R2 ;SET LU LOOP
3075 015574 004737 026412 JSR PC,SYNC ;SAVE LU REG FOR TYPEOUT
3076 015600 000005 5 ;SINGLE CLOCK 5 SYNC CHARACTERS
3077 015602 104415 000064 DATACLK, 64
3078 015606 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3079 015610 021244 021244 ;PORT4 LU12
3080 015612 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3081 015616 042704 000277 BIC #277,R4 ;CLEAR UNWANTED BITS
3082 015622 012705 000100 MOV #BIT6,R5 ;PUT 'EXPECTED' IN R5
3083 015626 120504 CMPB R5,R4 ;IS ACTIVE SET?
3084 015630 001401 BEQ 1$ ;BR IF YES
3085 015632 104040 HLT 40 ;ERROR ACTIVE IS NOT SET
3086 015634 104400 1$: SCOPE ;SCOPE THIS TEST
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096

```

```

:***** TEST 26 *****
:*DDCMP IN ACTIVE TEST
:*SET LU LOOP, SINGLE STEP 1 SYNC AND A NON-SYNC (301)
:*VERIFY THAT IN ACTIVE DOES NOT SET
:*****

```

```

: TEST 26
:-----

```

```

3097 015636 012737 000026 001226 TST26: MOV #26,TSTNO
3098 015644 012737 015724 001216 MOV #TST27,NEXT
3099
3100 015652 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3101 015654 012711 004000 MOV #BIT11,(R1) ;MASTER CLEAR DMC11
3102 015660 012702 000012 MOV #12,R2 ;SET LU LOOP
3103 015664 004737 026412 JSR PC,SYNC ;SAVE LU REG FOR TYPEOUT
3104 015670 000001 1 ;SINGLE CLOCK 1 SYNC CHARACTERS
3105 015672 104415 000024 DATACLK, 24
3106 015676 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3107 015700 021244 021244 ;PORT4 LU12
3108 015702 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3109 015706 042704 000277 BIC #277,R4 ;CLEAR UNWANTED BITS
3110 015712 005005 CLR R5 ;PUT 'EXPECTED' IN R5
3111 015714 120504 CMPB R5,R4 ;IS ACTIVE CLEAR?
3112 015716 001401 BEQ 1$ ;BR IF YES
3113 015720 104040 HLT 40 ;ERROR ACTIVE IS NOT CLEAR
3114 015722 104400 1$: SCOPE ;SCOPE THIS TEST
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124

```

***** TEST 27 *****
 ;*DDCMP IN ACTIVE TEST
 ;*SET LU LOOP, SINGLE STEP 2 SYNCs AND A NON-SYNC (301)
 ;*VERIFY THAT IN ACTIVE IS SET
 ;*****

; TEST 27

```

3125 015724 012737 000027 001226 TST27: MOV #27,TSTNO
3126 015732 012737 016014 001216 MOV #TST30,NEXT
3127
3128 015740 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3129 015742 012711 004000 MOV #BIT11,(R1) ;MASTER CLEAR DMC11
3130 015746 012702 000012 MOV #12,R2 ;SET LU LOOP
3131 015752 004737 026412 JSR PC,SYNC ;SAVE LU REG FOR TYPEOUT
3132 015756 000002 2 ;SINGLE CLOCK 2 SYNC CHARACTERS
3133 015760 104415 000034 DATACLK, 34
3134 015764 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3135 015766 021244 021244 ;PORT4 LU12
3136 015770 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
3137 015774 042704 000277 BIC #277,R4 ;CLEAR UNWANTED BITS
3138 016000 012705 000100 MOV #BIT6,R5 ;PUT 'EXPECTED' IN R5
3139 016004 120504 CMPB R5,R4 ;IS ACTIVE SET?
3140 016006 001401 BEQ 1$ ;BR IF YES
3141 016010 104040 HLT 40 ;ERROR ACTIVE IS NOT SET
3142 016012 104400 1$: SCOPE ;SCOPE THIS TEST
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152

```

***** TEST 30 *****
 ;*IN CLEAR TEST
 ;*SYNC UP RECEIVER AND TRANSMIT A CHARACTER
 ;*WAIT FOR IN RDY, THEN SET IN CLEAR
 ;*VERIFY THAT IN ACTIVE AND IN RDY ARE CLEARED
 ;*****

; TEST 30

Address	Hex Address	Hex Address	Hex Address	Hex Address	Label	Assembly	Comments
3153						-----	
3154	016014	012737	000030	001226	TST30:	MOV #30,TSTNO	
3155	016022	012737	016166	001216		MOV #TST31,NEXT	
3156							:R1 CONTAINS BASE DMC11 ADDRESS
3157	016030	104412				MSTCLR	:MASTER CLEAR DMC11
3158	016032	012702	000012			MOV #12,R2	:SAVE REG ADDRESS IN R2 FOR TYPEOUT
3159	016036	012711	004000			MOV #BIT11,(R1)	:SET LINE UNIT LOOP
3160	016042	004737	026560			JSR PC,CHAR	:LOAD SILO WITH 3 SYNCs
3161	016046	000301				301	:AND A NON-SYNC (301)
3162	016050	104415	000053			DATACLK, 53	:SINGLE CLOCK THE DATA
3163	016054	104416	000002			TIMER, 2	:WAIT FOR INRDY
3164	016060	104414				ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3165	016062	021244				021244	:PORT4 LU 12
3166	016064	016104	000004			MOV 4(R1),R4	:PUT 'FOUND' IN R4
3167	016070	042704	000357			BIC #357,R4	:CLEAR UNWANTED BITS
3168	016074	012705	000020			MOV #BIT4,R5	:PUT 'EXPECTED' IN R5
3169	016100	120504				CMPB R5,R4	:IS INRDY SET?
3170	016102	001401				BEQ 1\$	
3171	016104	104040				HLT 40	:ERROR, INRDY IS NOT SET
3172	016106				1\$:		
3173	016106	012761	000200	000004		MOV #BIT7,4(R1)	:LOAD PORT4
3174	016114	104414				ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3175	016116	122112				122112	:SET IN CLEAR
3176	016120	104414				ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3177	016122	021244				021244	:PORT4 LU 12
3178	016124	016104	000004			MOV 4(R1),R4	:PUT 'FOUND' IN R4
3179	016130	042704	000277			BIC #277,R4	:CLEAR UNWANTED BITS
3180	016134	005005				CLR R5	:PUT 'EXPECTED' IN R5
3181	016136	120504				CMPB R5,R4	:IS IN ACTIVE CLEAR?
3182	016140	001401				BEQ 2\$	
3183	016142	104040				HLT 40	:ERROR, IN ACTIVE IS NOT CLEAR
3184	016144				2\$:		
3185	016144	016104	000004			MOV 4(R1),R4	:PUT 'FOUND' IN R4
3186	016150	042704	000357			BIC #357,R4	:CLEAR UNWANTED BITS
3187	016154	005005				CLR R5	:PUT 'EXPECTED' IN R5
3188	016156	120504				CMPB R5,R4	:IS INRDY CLEARED?
3189	016160	001401				BEQ 3\$	
3190	016162	104040				HLT 40	:ERROR, INRDY IS NOT CLEARED
3191	016164	104400			3\$:	SCOPE	:SCOPE THIS TEST
3192							
3193							
3194						***** TEST 31 *****	
3195						:*DDCMP BASIC RECEICER TEST	
3196						:*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 0	
3197						:*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED	
3198						:*****	
3199							
3200						: TEST 31	
3201						-----	
3202	016166	012737	000031	001226	TST31:	MOV #31,TSTNO	
3203	016174	012737	016302	001216		MOV #TST32,NEXT	
3204							:R1 CONTAINS BASE DMC11 ADDRESS
3205	016202	104412					

```

3209 016220 000000 0
3210 016222 104415 000053 DATACLK, 53
3211 016226 104416 000002 TIMER, 2
3212 016232 104414 ROMCLK
3213 016234 021244 021244
3214 016236 016104 000004 MOV 4(R1),R4
3215 016242 042704 000357 BIC #357,R4
3216 016246 012705 000020 MOV #BIT4,R5
3217 016252 120504 CMPB R5,R4
3218 016254 001401 BEQ 1$
3219 016256 104040 HLT 40
3220 016260 1$:
3221 016260 104414 ROMCLK
3222 016262 021204 021204
3223 016264 016104 000004 MOV 4(R1),R4
3224 016270 005005 CLR R5
3225 016272 120504 CMPB R5,R4
3226 016274 001401 BEQ 2$
3227 016276 104010 HLT 10
3228 016300 104400 2$: SCOPE
3229
3230
3231 ;***** TEST 32 *****
3232 ;*DDCMP BASIC RECEICER TEST
3233 ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 125
3234 ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3235 ;*****
3236
3237 ; TEST 32
3238 ;-----
3239 016302 012737 000032 001226 TST32: MOV #32,TSTNO
3240 016310 012737 016420 001216 MOV #TST33,NEXT
3241
3242 016316 104412 MSTCLR
3243 016320 012702 000012 MOV #12,R2
3244 016324 012711 004000 MOV #BIT11,(R1)
3245 016330 004737 026560 JSR PC,CHAR
3246 016334 000125 125
3247 016336 104415 000053 DATACLK, 53
3248 016342 104416 000002 TIMER, 2
3249 016346 104414 ROMCLK
3250 016350 021244 021244
3251 016352 016104 000004 MOV 4(R1),R4
3252 016356 042704 000357 BIC #357,R4
3253 016362 012705 000020 MOV #BIT4,R5
3254 016366 120504 CMPB R5,R4
3255 016370 001401 BEQ 1$
3256 016372 104040 HLT 40
3257 016374 1$:
3258 016374 104414 ROMCLK
3259 016376 021204 021204
3260 016400 016104 000004 MOV 4(R1),R4
3261 016404 012705 000125 MOV #125,R5
3262 016410 120504 CMPB R5,R4
3263 016412 001401 BEQ 2$
3264 016414 104010 HLT 10

```

```

;AND THE CHARACTER 0
;SINGLE CLOCK THE DATA
;WAIT FOR INRDY
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4 LU 12
;PUT 'FOUND' IN R4
;CLEAR UNWANTED BITS
;PUT 'EXPECTED' IN R5
;IS INRDY SET?
;ERROR, INRDY IS NOT SET
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4 IN DATA
;PUT 'FOUND' IN R4
;PUT 'EXPECTED' IN R5
;WAS A 0 RECEIVED?
;ERROR, RECEIVED DATA IS WRONG
;SCOPE THIS TEST

```

```

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SAVE REG ADDRESS IN R2 FOR TYPEOUT
;SET LINE UNIT LOOP
;LOAD SILO WITH 3 SYNCs
;AND THE CHARACTER 125
;SINGLE CLOCK THE DATA
;WAIT FOR INRDY
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4 LU 12
;PUT 'FOUND' IN R4
;CLEAR UNWANTED BITS
;PUT 'EXPECTED' IN R5
;IS INRDY SET?
;ERROR, INRDY IS NOT SET
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4 IN DATA
;PUT 'FOUND' IN R4
;PUT 'EXPECTED' IN R5
;WAS A 125 RECEIVED?
;ERROR, RECEIVED DATA IS WRONG

```

```

3265 016416 104400          2$: SCOPE                      ;SCOPE THIS TEST
3266
3267
3268          :***** TEST 33 *****
3269          :DDCMP BASIC RECEICER TEST
3270          :*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 252
3271          :*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3272          :*****
3273
3274          : TEST 33
3275          :-----
3276 016420 012737 000033 001226 TST33: MOV #33,TSTNO
3277 016426 012737 016536 001216      MOV #TST34,NEXT
3278
3279 016434 104412          MSTCLR                      ;R1 CONTAINS BASE DMC11 ADDRESS
3280 016436 012702 000012      MOV #12,R2              ;MASTER CLEAR DMC11
3281 016442 012711 004000      MOV #BIT11,(R1)          ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3282 016446 004737 026560      JSR PC,CHAR              ;SET LINE UNIT LOOP
3283 016452 000252          252                        ;LOAD SILO WITH 3 SYNCs
3284 016454 104415 000053      DATACLK, 53            ;AND THE CHARACTER 252
3285 016460 104416 000002      TIMER, 2                ;SINGLE CLOCK THE DATA
3286 016464 104414          ROMCLK                      ;WAIT FOR INRDY
3287 016466 021244          021244                    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3288 016470 016104 000004      MOV 4(R1),R4              ;PORT4 LU 12
3289 016474 042704 000357      BIC #357,R4              ;PUT 'FOUND' IN R4
3290 016500 012705 000020      MOV #BIT4,R5              ;CLEAR UNWANTED BITS
3291 016504 120504          CMPB R5,R4                  ;PUT 'EXPECTED' IN R5
3292 016506 001401          BEQ 1$                      ;IS INRDY SET?
3293 016510 104040          HLT 40                      ;ERROR, INRDY IS NOT SET
3294 016512          1$:
3295 016512 104414          ROMCLK                      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3296 016514 021204          021204                    ;PORT4 IN DATA
3297 016516 016104 000004      MOV 4(R1),R4              ;PUT 'FOUND' IN R4
3298 016522 012705 000252      MOV #252,R5              ;PUT 'EXPECTED' IN R5
3299 016526 120504          CMPB R5,R4                  ;WAS A 252 RECEIVED?
3300 016530 001401          BEQ 2$                      ;
3301 016532 104010          HLT 10                      ;ERROR, RECEIVED DATA IS WRONG
3302 016534 104400          2$: SCOPE                      ;SCOPE THIS TEST
3303
3304
3305          :***** TEST 34 *****
3306          :DDCMP BASIC RECEICER TEST
3307          :*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 377
3308          :*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3309          :*****
3310
3311          : TEST 34
3312          :-----
3313 016536 012737 000034 001226 TST34: MOV #34,TSTNO
3314 016544 012737 016654 001216      MOV #TST35,NEXT
3315
3316 016552 104412          MSTCLR                      ;R1 CONTAINS BASE DMC11 ADDRESS
3317 016554 012702 000012      MOV #12,R2              ;MASTER CLEAR DMC11
3318 016560 012711 004000      MOV #BIT11,(R1)          ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3319 016564 004737 026560      JSR PC,CHAR              ;SET LINE UNIT LOOP
3320 016570 000377          377                        ;LOAD SILO WITH 3 SYNCs
          ;AND THE CHARACTER 377

```

```

3321 016572 104415 000053          DATACLK, 53          ;SINGLE CLOCK THE DATA
3322 016576 104416 000002          TIMER, 2          ;WAIT FOR INRDY
3323 016602 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3324 016604 021244          021244          ;PORT4 LU 12
3325 016606 016104 000004          MOV 4(R1),R4          ;PUT 'FOUND' IN R4
3326 016612 042704 000357          BIC #357,R4          ;CLEAR UNWANTED BITS
3327 016616 012705 000020          MOV #BIT4,R5          ;PUT 'EXPECTED' IN R5
3328 016622 120504          CMPB R5,R4          ;IS INRDY SET?
3329 016624 001401          BEQ 1$
3330 016626 104040          HLT 40          ;ERROR, INRDY IS NOT SET
3331 016630          1$:
3332 016630 104414          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3333 016632 021204          021204          ;PORT4 IN DATA
3334 016634 016104 000004          MOV 4(R1),R4          ;PUT 'FOUND' IN R4
3335 016640 012705 000377          MOV #377,R5          ;PUT 'EXPECTED' IN R5
3336 016644 120504          CMPB R5,R4          ;WAS A 377 RECEIVED?
3337 016646 001401          BEQ 2$
3338 016650 104010          HLT 10          ;ERROR, RECEIVED DATA IS WRONG
3339 016652 104400          2$: SCOPE          ;SCOPE THIS TEST
3340
3341
3342          ;***** TEST 35 *****
3343          ;*DDCMP DATA TEST
3344          ;*THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
3345          ;*CHECKING EACH CHARACTER AS IT IS RECEIVED
3346          ;*****
3347
3348          : TEST 35
3349          :-----
3350 016654 012737 000035 001226 TST35: MOV #35,TSTNO
3351 016662 012737 017004 001216          MOV #TST36,NEXT
3352
3353          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
3354 016672 005037 027176          CLR SCHAR          ;MASTER CLEAR DMC11
3355 016676 005037 027200          CLR STUFLG          ;START BINARY COUNT AT ZERO
3356 016702 005002          CLR R2          ;CLEAR BITSTUFF FLAG
3357 016704 012703 000073          MOV #73,R3          ;R2 IS 'EXPECTED' DATA
3358 016710 012711 004000          MOV #BIT11,(R1)          ;R3 IS CHARACTER COUNT
3359 016714 004737 026736          JSR PC,SILOLD          ;SET LINE UNIT LOOP
3360 016720 104415 000043          DATACLK, 43          ;LOAD SILO WITH COUNT PATTERN
3361 016724 104415 000730          1$: DATACLK, 730          ;SYNC RECEIVER AND GET IT ACTIVE
3362 016730 004737 027202          4$: JSR PC,INRDY          ;CLOCK IN 73 CHARACTERS
3363 016734 104414          ROMCLK          ;WAIT FOR INRDY
3364 016736 021204          021204          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3365 016740 016104 000004          MOV 4(R1),R4          ;PORT4 IN DATA
3366 016744 010205          MOV R2,R5          ;PUT 'FOUND' IN R4
3367 016746 120504          CMPB R5,R4          ;PUT 'EXPECTED' IN R5
3368 016750 001401          BEQ 2$          ;IS DATA CORRECT?
3369 016752 104010          HLT 10          ;BR IF YES
3370 016754 005202          2$: INC R2          ;DATA ERROR
3371 016756 022702 000400          CMP #400,R2          ;NEXT CHARACTER
3372 016762 001407          BEQ 3$          ;ALL DONE?
3373 016764 005303          DEC R3          ;BR IF YES
3374 016766 001360          BNE 4$          ;DECREMENT CHARACTER COUNT
3375 016770 004737 026736          JSR PC,SILOLD          ;BR IF SILO NOT EMPTY
3376 016774 012703 000073          MOV #73,R3          ;LOAD SILO WITH MORE OF COUNT PATTERN
          ;RELOAD CHARACTER COUNT

```

CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 66
 CZDMEC.P11 11-SEP-78 12:38 BASIC RECEIVER TESTS

SEQ 0066

```

3377 017000 000751          BR      1$          ;CONTINUE
3378 017002 104400          3$:      SCOPE          ;SCOPE THIS TEST
3379
3380
3381          ;***** TEST 36 *****
3382          ;*DDCMP DATA TEST
3383          ;*THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
3384          ;*CHECKING EACH CHARACTER AS IT IS RECEIVED
3385          ;*THIS TEST IS EXACTLY THE SAME AS THE LAST TEST,
3386          ;*EXCEPT LINE UNIT LOOP IS SET IN LU REGISTER 12
3387          ;*****
3388
3389          ; TEST 36
3390          ;-----
3391 017004 012737 000036 001226 TST36: MOV      #36,TSTNO
3392 017012 012737 017144 001216      MOV      #TST37,NEXT
3393
3394 017020 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
3395 017022 005037 027176      CLR      SCHAR          ;MASTER CLEAR DMC11
3396 017026 005037 027200      CLR      STUFLG          ;START BINARY COUNT AT ZERO
3397 017032 005002          CLR      R2          ;CLEAR BITSTUFF FLAG
3398 017034 012703 000073      MOV      #73,R3          ;R2 IS 'EXPECTED' DATA
3399 017040 005011          CLR      (R1)          ;R3 IS CHARACTER COUNT
3400 017042 012761 000040 000004      MOV      #BIT5,4(R1) ;CLEAR LU LOOP IN MAINT REG
3401 017050 104414          ROMCLK          ;LOAD PORT4
3402 017052 122112          122112          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3403 017054 004737 026736      JSR      PC,SILOLD          ;SET LU LOOP IN LU REG 12
3404 017060 104415 000043      DATACLK, 43          ;LOAD SILO WITH COUNT PATTERN
3405 017064 104415 000730      DATACLK, 730          ;SYNC RECEIVER AND GET IT ACTIVE
3406 017070 004737 027202          1$:      JSR      PC,INRDY          ;CLOCK IN 73 CHARACTERS
3407 017074 104414          4$:          ;WAIT FOR INRDY
3408 017076 021204          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3409 017100 016104 000004      MOV      4(R1),R4          ;PORT4 IN DATA
3410 017104 010205          MOV      R2,R5          ;PUT 'FOUND' IN R4
3411 017106 120504          CMPB      R5,R4          ;PUT 'EXPECTED' IN R5
3412 017110 001401          BEQ      2$          ;IS DATA CORRECT?
3413 017112 104010          HLT      10          ;BR IF YES
3414 017114 005202          2$:      INC      R2          ;DATA ERROR
3415 017116 022702 000400      CMP      #400,R2          ;NEXT CHARACTER
3416 017122 001407          BEQ      3$          ;ALL DONE?
3417 017124 005303          DEC      R3          ;BR IF YES
3418 017126 001360          BNE      4$          ;DECREMENT CHARACTER COUNT
3419 017130 004737 026736      JSR      PC,SILOLD          ;BR IF SILO NOT EMPTY
3420 017134 012703 000073      MOV      #73,R3          ;LOAD SILO WITH MORE OF COUNT PATTERN
3421 017140 000751          BR      1$          ;RELOAD CHARACTER COUNT
3422 017142 104400          3$:      SCOPE          ;CONTINUE
3423          ;SCOPE THIS TEST
3424
3425          ;***** TEST 37 *****
3426          ;*TRANSMITTER MARK TEST
3427          ;*SINGLE CLOCK 3 SYNCs AND A 301 AND 20 EXTRA
3428          ;*CLOCK TICKS, VERIFY THAT A 301, A 377 AND A 377
3429          ;*WERE RECEIVED INDICATING THAT THE TRANSMITTER WENT
3430          ;*TO A MARK STATE FOR 16 BITS WHEN OUT SILO WAS EMPTY
3431          ;*****
3432

```

```

3433      : TEST 37
3434      :-----
3435 017144 012737 000037 001226 TST37: MOV #37,TSTNO
3436 017152 012737 017304 001216 MOV #TST40,NEXT
3437
3438 017160 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3439 017162 012711 004000 MOV #BIT11,(R1) ;MASTER CLEAR DMC11
3440 017166 004737 026560 JSR PC,CHAR ;SET LINE UNIT LOOP
3441 017172 000301 301 ;LOAD SILO WITH 3 SYNCs
3442 017174 104415 000073 DATACLK, 73 ;AND A 301
3443 017200 004737 027202 JSR PC,INRDY ;CLOCK THE 301 IN AND 20 EXTRA TICKS
3444 017204 104414 ROMCLK ;WAIT FOR INRDY
3445 017206 021204 021204 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3446 017210 016104 000004 MOV 4(R1),R4 ;PORT4 IN DATA
3447 017214 012705 000301 MOV #301,R5 ;PUT 'FOUND' IN R4
3448 017220 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
3449 017222 001401 BEQ 1$ ;WAS A 301 RECEIVED?
3450 017224 104010 HLT 10 ;
3451 017226 004737 027202 1$: JSR PC,INRDY ;ERROR FIRST CHARACTER INCORRECT
3452 017232 104414 ROMCLK ;WAIT FOR INRDY
3453 017234 021204 021204 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3454 017236 016104 000004 MOV 4(R1),R4 ;PORT4 IN DATA
3455 017242 012705 000377 MOV #377,R5 ;PUT 'FOUND' IN R4
3456 017246 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
3457 017250 001401 BEQ 2$ ;WAS A 377 RECEIVED?
3458 017252 104010 HLT 10 ;
3459 017254 004737 027202 2$: JSR PC,INRDY ;ERROR, 377 WAS NOT RECEIVED
3460 017260 104414 ROMCLK ;WAIT FOR INRDY
3461 017262 021204 021204 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3462 017264 016104 000004 MOV 4(R1),R4 ;PORT4 IN DATA
3463 017270 012705 000377 MOV #377,R5 ;PUT 'FOUND' IN R4
3464 017274 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
3465 017276 001401 BEQ 3$ ;WAS A 377 RECEIVED?
3466 017300 104010 HLT 10 ;
3467 017302 104400 3$: SCOPE ;ERROR, 377 WAS NOT RECEIVED
3468 ;SCOPE THIS TEST
3469
3470 ;***** TEST 40 *****
3471 ;*CABLE TURNAROUND TEST
3472 ;*CLEAR LINE UNIT LOOP, SET DTR
3473 ;*VERIFY THAT MODEM READY IS SET
3474 ;*CLEAR DTR, VERIFY THAT MRDY IS CLEARED
3475 ;*****
3476
3477 : TEST 40
3478 :-----
3479 017304 012737 000040 001226 TST40: MOV #40,TSTNO
3480 017312 012737 017466 001216 MOV #TST41,NEXT
3481
3482 017320 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3483 017322 032737 020000 001366 BIT #BIT13,STAT1 ;MASTER CLEAR DMC11
3484 017330 001004 BNE .+12 ;IS LINE UNIT M8202?
3485 017332 032737 040000 001366 BIT #BIT14,STAT1 ;BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
3486 017340 001451 BEQ 2$ ;IS TURNAROUND CONNECTOR ON?
3487 017342 005011 CLR (R1) ;SKIP TEST IF NO
3488 017344 012761 000100 000004 MOV #100,4(R1) ;CLEAR LINE UNIT LOOP
;LOAD PORT4

```

3489	017352	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3490	017354	122113			122113		:SET DTR
3491	017356	104416	000002		TIMER, 2		:WAIT
3492	017362	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3493	017364	021264			021264		:PORT4, LU13
3494	017366	016104	000004		MOV 4(R1),R4		:PUT 'FOUND' IN R4
3495	017372	042704	000223		BIC #223,R4		:CLEAR UNWANTED BITS
3496	017376	012705	000110		MOV #110,R5		:PUT 'EXPECTED' IN R5
3497	017402	120504			CMPB R5,R4		:IS MRDY SET?
3498	017404	001401			BEQ 1\$		
3499	017406	104011			HLT 11		:ERROR, MRDY NOT SET
3500	017410	005061	000004	1\$:	CLR 4(R1)		:CLEAR PORT4
3501	017414	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3502	017416	122113			122113		:CLEAR DTR
3503	017420	104416	000002		TIMER, 2		
3504	017424	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3505	017426	021264			021264		:PORT4, LU13
3506	017430	016104	000004		MOV 4(R1),R4		:PUT 'FOUND' IN R4
3507	017434	042704	000223		BIC #223,R4		:CLEAR UNWANTED BITS
3508	017440	005005			CLR R5		:PUT 'EXPECTED' IN R5
3509	017442	032737	020000	001366	BIT #BIT13,STAT1		:IS LINE UNIT M8202?
3510	017450	001402			BEQ .+6		:BR IF NO
3511	017452	052705	000010		BIS #BIT3,R5		:MRDY SET ON M8202
3512	017456	120504			CMPB R5,R4		:IS MRDY CLEAR?
3513	017460	001401			BEQ 2\$		
3514	017462	104011			HLT 11		:ERROR, MRDY NOT CLEAR
3515	017464	104400		2\$:	SCOPE		:SCOPE THIS TEST
3516							
3517							
3518							:***** TEST 41 *****
3519							:*CABLE TURNAROUND TEST
3520							:*CLEAR LINE UNIT LOOP, LOAD OUT DATA SILO
3521							:*VERIFY THAT ALL MODEM SIGNALS ARE SET
3522							:*****
3523							
3524							: TEST 41
3525							:-----
3526	017466	012737	000041	001226	TST41: MOV #41,TSTNO		
3527	017474	012737	017532	001216	MOV #TST42,NEXT		
3528							
3529	017502	104412			MSTCLR		:R1 CONTAINS BASE DMC11 ADDRESS
3530	017504	032737	020000	001366	BIT #BIT13,STAT1		:MASTER CLEAR DMC11
3531	017512	001004			BNE .+12		:IS LINE UNIT M8202?
3532	017514	032737	040000	001366	BIT #BIT14,STAT1		:BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
3533	017522	001442			BEQ 1\$		:IS TURNAROUND CONNECTOR ON?
3534	017524	012711	004000		MOV #BIT11,(R1)		:SKIP TEST IF NO
3535	017530	012761	000100	000004	MOV #100, 4(R1)		:SET LINE UNIT LOOP
3536	017536	104414			ROMCLK		:LOAD PORT4
3537	017540	122113			122113		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3538	017542	104416	000002		TIMER, 2		:CLEAR ALL MODEM SIGNALS, EXCEPT DTR
3539	017546	012761	000001	000004	MOV #1,4(R1)		:WAIT
3540	017554	104414			ROMCLK		:LOAD PORT4
3541	017556	122111			122111		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3542	017560	004537	027644		JSR R5,MESLD		:SET SOM
3543	017564	030126			MESDAT		:FILL OUT DATA SILO
3544	017566	000100			64.		:WITH 64 CHARACTERS

CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 69
 CZDMEC.P11 11-SEP-78 12:38 BASIC RECEIVER TESTS

SEQ 0069

```

3545 017570 012700 000050      MOV    #50,R0      ;PREPARE FOR DELAY
3546 017574 005011              CLR    (R1)        ;CLEAR LINE UNIT LOOP
3547 017576              2$: ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3548 017576 104414              021264    ;PORT4 LU13
3549 017600 021264              MOV    4(R1),R4    ;PUT 'FOUND' IN R4
3550 017602 016104 000004      BIC    #223,R4    ;CLEAR UNWANTED BITS
3551 017606 042704 000223      MOV    #154,R5    ;PUT 'EXPECTED' IN R5
3552 017612 012705 000154      CMPB   R5,R4    ;COMPARE EXPECTED AND FOUND
3553 017616 120504              BEQ     1$        ;BR IF OK
3554 017620 001403              DEC     R0        ;DEC DELAY COUNT
3555 017622 005300              BNE     2$        ;BR IF NOT ZERO
3556 017624 001364              HLT     11        ;ERROR, ALL SIGNALS ARE NOT SET
3557 017626 104011              1$: SCOPE      ;SCOPE THIS TEST
3558 017630 104400
3559
3560
3561      ;***** TEST 42 *****
3562      ;*TEST OF CRC OPERATION
3563      ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3564      ;*0, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3565      ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3566      ;*****
3567
3568      ; TEST 42
3569      ;-----
3570 017632 012737 000042 001226 TST42: MOV    #42,TSTNO
3571 017640 012737 020146 001216      MOV    #TST43,NEXT
3572 017646 012737 017662 001220      MOV    #64$,LOCK
3573
3574 017654 104412              MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
3575 017656 012711 004000      MOV    #BIT11,(R1) ;MASTER CLEAR DMC11
3576 017662 004737 027706      JSR     PC,CLR10    ;SET LU LOOP
3577 017666 005000              CLR     R0        ;CLEAR BCC REGISTERS
3578 017670 012737 120001 027342      MOV    #CRC16,XPOLY ;START SHIFT COUNTER AT ZERO
3579 017676 012737 000000 017736      MOV    #0,66$      ;LOAD POLYNOMIAL FOR SOFTWARE BCC
3580 017704 005037 017740      CLR     67$        ;LOAD CHAR FOR SOFTWARE BCC
3581 017710 004737 027346      JSR     PC,BCCLD    ;CLEAR OLD SOFTWARE BCC
3582 017714 000000              0          ;LOAD OUT SILO WITH 2 SYNCs
3583 017716 104415 000021      DATACL, 21        ;AND THE CHARACTER 0
3584 017722 104415 000001      DATACL, 1          ;GET TRANSMITTER ACTIVE
3585 017726 005200              INC     R0        ;SHIFT BCC ONCE
3586 017730 004537 027236      JSR     R5,SIMBCC   ;BUMP SHIFT COUNT
3587 017734 000001              1          ;CALCULATE SOFTWARE BCC LSB
3588 017736 000000              66$: 0        ;ONE SHIFT
3589 017740 000000              67$: 0        ;DATA CHARACTER
3590 017742 103405              BCS     68$      ;OLD BCC
3591 017744 004737 027460      JSR     PC,GETQ0    ;BR IF SOFT BCC LSB IS SET
3592 017750 103006              BCC     69$      ;GET HARDWARE TRANSMITTER BCC LSB
3593 017752 104012              HLT     12        ;BR IF HARD BCC LSB IS CLEAR
3594 017754 000404              BR      69$      ;ERROR, BCC LSB IS SET
3595 017756 004737 027460      68$: JSR     PC,GETQ0    ;CONTINUE
3596 017762 103401              BCS     69$      ;GET HARDWARE TRANSMITTER BCC LSB
3597 017764 104016              HLT     16        ;BR IF HARD BCC LSB IS SET
3598 017766              69$:          ;ERROR, HARD BCC LSB IS CLEAR
3599 017766 006037 017736      ROR     60$      ;SHIFT SOFT DATA
3600 017772 013737 027344 017740      MOV    CALBCC,67$ ;LOAD OLD SOFT BCC

```

3601	020000	022700	000010			CMP	#10,R0	:DONE YET?
3602	020004	001346				BNE	65\$	:BR IF NOT DONE
3603	020006	104401				SCOPI		:SCOPE SUBTEST (SW09=1)
3604	020010	012737	020016	001220		MOV	#71\$,LOCK	:NEW SCOPE1
3605	020016	004737	027706		71\$:	JSR	PC,CLRIO	:CLEAR BCC REGISTERS
3606	020022	005000				CLR	R0	:START SHIFT COUNTER AT ZERO
3607	020024	012737	120001	027342		MOV	#CRC16,XPOLY	:LOAD POLYNOMIAL FOR SOFTWARE BCC
3608	020032	012737	000000	020072		MOV	#0,73\$	:LOAD CHAR FOR SOFTWARE BCC
3609	020040	005037	020074			CLR	74\$	:CLEAR OLD SOFTWARE BCC
3610	020044	004737	027346			JSR	PC,BCCLD	:LOAD OUT SILO WITH 2 SYNCs
3611	020050	000000				0		:AND THE CHARACTER 0
3612	020052	104415	000032			DATACLK,	32	:GET RECEIVER ACTIVE
3613	020056	104415	000001		72\$:	DATACLK,	1	:SHIFT BCC ONCE
3614	020062	005200				INC	R0	:BUMP SHIFT COUNT
3615	020064	004537	027236			JSR	R5,SIMBCC	:CALCULATE SOFTWARE BCC LSB
3616	020070	000001				1		:ONE SHIFT
3617	020072	000000			73\$:	0		:DATA CHARACTER
3618	020074	000000			74\$:	0		:OLD BCC
3619	020076	103405				BCS	75\$	:BR IF SOFT BCC LSB IS SET
3620	020100	004737	027472			JSR	PC,GETQI	:GET HARDWARE RECEIVER BCC LSB
3621	020104	103006				BCC	76\$	:BR IF HARD BCC LSB IS CLEAR
3622	020106	104013				HLT	13	:ERROR, BCC LSB IS SET
3623	020110	000404				BR	76\$	:CONTINUE
3624	020112	004737	027472		75\$:	JSR	PC,GETQI	:GET HARDWARE RECEIVER BCC LSB
3625	020116	103401				BCS	76\$	:BR IF HARD BCC LSB IS SET
3626	020120	104017				HLT	17	:ERROR, BCC LSB IS CLEAR
3627	020122				76\$:			
3628	020122	006037	020072			ROR	73\$	:SHIFT SOFT DATA
3629	020126	013737	027344	020074		MOV	CALBCC,74\$	:LOAD OLD SOFT BCC
3630	020134	022700	000010			CMP	#10,R0	:DONE YET?
3631	020140	001346				BNE	72\$	:BR IF NOT DONE
3632	020142	104401				SCOPI		:SCOPE SUBTEST (SW09=1)
3633	020144	104400			77\$:	SCOPE		:SCOPE THIS TEST

3634
 3635
 3636
 3637
 3638
 3639
 3640
 3641
 3642
 3643
 3644
 3645
 3646
 3647
 3648
 3649
 3650
 3651
 3652
 3653
 3654
 3655
 3656

***** TEST 43 *****
 :*TEST OF CRC OPERATION
 :*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
 :*377, VERIFY THE LSB OF THE BCC ON EACH SHIFT
 :*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
 :*****

: TEST 43

3645	020146	012737	000043	001226	TST43:	MOV	#43,TSTNO	
3646	020154	012737	020462	001216		MOV	#TST44,NEXT	
3647	020162	012737	020176	001220		MOV	#64\$,LOCK	
3648								:R1 CONTAINS BASE DMC11 ADDRESS
3649	020170	104412				MSTCLR		:MASTER CLEAR DMC11
3650	020172	012711	004000			MOV	#BIT11,(R1)	:SET LJ LOOP
3651	020176	004737	027706		64\$:	JSR	PC,CLRIO	:CLEAR BCC REGISTERS
3652	020202	005000				CLR	R0	:START SHIFT COUNTER AT ZERO
3653	020204	012737	120001	027342		MOV	#CRC16,XPOLY	:LOAD POLYNOMIAL FOR SOFTWARE BCC
3654	020212	012737	000377	020252		MOV	#377,66\$;	:LOAD CHAR FOR SOFTWARE BCC
3655	020220	005037	020254			CLR	67\$	:CLEAR OLD SOFTWARE BCC
3656	020224	004737	027346			JSR	PC,BCCLD	:LOAD OUT SILO WITH 2 SYNCs


```

3713      : *USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3714      : *125, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3715      : *TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3716      : *****
3717
3718      : TEST 44
3719      : -----
3720 020462 012737 000044 001226 TST44: MOV #44,TSTNO
3721 020470 012737 020776 001216 MOV #TST45,NEXT
3722 020476 012737 020512 001220 MOV #64$,LOCK
3723
3724 020504 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3725 020506 012711 004000 ;MASTER CLEAR DMC11
3726 020512 004737 027706 64$: MOV #BIT11,(R1) ;SET LU LOOP
3727 020516 005000 CLR PC,CLRIO ;CLEAR BCC REGISTERS
3728 020520 012737 120001 027342 CLR R0 ;START SHIFT COUNTER AT ZERO
3729 020526 012737 000125 020566 MOV #CRC16,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
3730 020534 005037 020570 CLR #125,66$; ;LOAD CHAR FOR SOFTWARE BCC
3731 020540 004737 027346 JSR PC,BCCLD ;CLEAR OLD SOFTWARE BCC
3732 020544 000125 125 ;LOAD OUT SILO WITH 2 SYNCs
3733 020546 104415 000021 DATACLK, 2' ;AND THE CHARACTER 125
3734 020552 104415 000001 65$: DATACLK, 1 ;GET TRANSMITTER ACTIVE
3735 020556 005200 INC R0 ;SHIFT BCC ONCE
3736 020560 004537 027236 JSR R5,SIMBCC ;BUMP SHIFT COUNT
3737 020564 000001 1 ;CALCULATE SOFTWARE BCC LSB
3738 020566 000000 66$: 0 ;ONE SHIFT
3739 020570 000000 67$: 0 ;DATA CHARACTER
3740 020572 103405 BCS 68$ ;OLD BCC
3741 020574 004737 027460 JSR PC,GETQO ;BR IF SOFT BCC LSB IS SET
3742 020600 103006 BCC 69$ ;GET HARDWARE TRANSMITTER BCC LSB
3743 020602 104012 HLT 12 ;BR IF HARD BCC LSB IS CLEAR
3744 020604 000404 BR 69$ ;ERROR, BCC LSB IS SET
3745 020606 004737 027460 68$: JSR PC,GETQO ;CONTINUE
3746 020612 103401 BCS 69$ ;GET HARDWARE TRANSMITTER BCC LSB
3747 020614 104016 HLT 16 ;BR IF HARD BCC LSB IS SET
3748 020616 006037 020566 69$: ;ERROR, HARD BCC LSB IS CLEAR
3749 020622 013737 027344 020570 ROR 66$ ;SHIFT SOFT DATA
3750 020630 022700 000010 MOV CALBCC,67$ ;LOAD OLD SOFT BCC
3751 020634 001346 CMP #10,R0 ;DONE YET?
3752 020636 104401 BNE 65$ ;BR IF NOT DONE
3753 020640 012737 020646 001220 SCOP1 ;SCOPE SUBTEST (SW09=1)
3754 020646 004737 027706 71$: MOV #71$,LOCK ;NEW SCOPE1
3755 020652 005000 CLR PC,CLRIO ;CLEAR BCC REGISTERS
3756 020654 012737 120001 027342 CLR R0 ;START SHIFT COUNTER AT ZERO
3757 020662 012737 000125 020722 MOV #CRC16,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
3758 020670 005037 020724 CLR #125,73$; ;LOAD CHAR FOR SOFTWARE BCC
3759 020674 004737 027346 JSR PC,BCCLD ;CLEAR OLD SOFTWARE BCC
3760 020700 000125 125 ;LOAD OUT SILO WITH 2 SYNCs
3761 020702 104415 000032 DATACLK, 32 ;AND THE CHARACTER 125
3762 020706 104415 000001 72$: DATACLK, 1 ;GET RECEIVER ACTIVE
3763 020712 005200 INC R0 ;SHIFT BCC ONCE
3764 020714 004537 027236 JSR R5,SIMBCC ;BUMP SHIFT COUNT
3765 020720 000001 1 ;CALCULATE SOFTWARE BCC LSB
3766 020722 000000 73$: 0 ;ONE SHIFT
3767 020724 000000 74$: 0 ;DATA CHARACTER
3768 ;OLD BCC

```

3769	020726	103405				BCC	75\$	;BR IF SOFT BCC LSB IS SET
3770	020730	004737	027472			JSR	PC,GETQI	;GET HARDWARE RECEIVER BCC LSB
3771	020734	103006				BCC	76\$	;BR IF HARD BCC LSB IS CLEAR
3772	020736	104013				HLT	13	;ERROR, BCC LSB IS SET
3773	020740	000404				BR	76\$	;CONTINUE
3774	020742	004737	027472		75\$:	JSR	PC,GETQI	;GET HARDWARE RECEIVER BCC LSB
3775	020746	103401				BCC	76\$	;BR IF HARD BCC LSB IS SET
3776	020750	104017				HLT	17	;ERROR, BCC LSB IS CLEAR
3777	020752				76\$:			
3778	020752	006037	020722			ROR	73\$	;SHIFT SOFT DATA
3779	020756	013737	027344	020724		MOV	CALBCC,74\$	;LOAD OLD SOFT BCC
3780	020764	022700	000010			CMP	#10,R0	;DONE YET?
3781	020770	001346				BNE	72\$	;BR IF NOT DONE
3782	020772	104401				SCOPI		;SCOPE SUBTEST (SW09=1)
3783	020774	104400			77\$:	SCOPE		;SCOPE THIS TEST
3784								
3785								
3786								:***** TEST 45 *****
3787								:*TEST OF CRC OPERATION
3788								:*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3789								:*252, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3790								:*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3791								:*****
3792								
3793								: TEST 45
3794								:-----
3795	020776	012737	000045	001226	TST45:	MOV	#45,TSTNO	
3796	021004	012737	021312	001216		MOV	#TST46,NEXT	
3797	021012	012737	021026	001220		MOV	#64\$,LOCK	
3798								
3799	021020	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
3800	021022	012711	004000			MOV	#BIT11,(R1)	;MASTER CLEAR DMC11
3801	021026	004737	027706		64\$:	JSR	PC,CLRIO	;SET LU LOOP
3802	021032	005000				CLR	R0	;CLEAR BCC REGISTERS
3803	021034	012737	120001	027342		MOV	#CRC16,XPOLY	;START SHIFT COUNTER AT ZERO
3804	021042	012737	000252	021102		MOV	#252,66\$;	;LOAD POLYNOMIAL FOR SOFTWARE BCC
3805	021050	005037	021104			CLR	67\$	;LOAD CHAR FOR SOFTWARE BCC
3806	021054	004737	027346			JSR	PC,BCCLD	;CLEAR OLD SOFTWARE BCC
3807	021060	000252				252		;LOAD OUT SILO WITH 2 SYNC'S
3808	021062	104415	000021			DATACLK,	21	;AND THE CHARACTER 252
3809	021066	104415	000001		65\$:	DATACLK,	1	;GET TRANSMITTER ACTIVE
3810	021072	005200				INC	R0	;SHIFT BCC ONCE
3811	021074	004537	027236			JSR	R5,SIMBCC	;BUMP SHIFT COUNT
3812	021100	000001				1		;CALCULATE SOFTWARE BCC LSB
3813	021102	000000			66\$:	0		;ONE SHIFT
3814	021104	000000			67\$:	0		;DATA CHARACTER
3815	021106	103405				BCC	68\$	;OLD BCC
3816	021110	004737	027460			JSR	PC,GETQO	;BR IF SOFT BCC LSB IS SET
3817	021114	103006				BCC	69\$	;GET HARDWARE TRANSMITTER BCC LSB
3818	021116	104012				HLT	12	;BR IF HARD BCC LSB IS CLEAR
3819	021120	000404				BR	69\$	;ERROR, BCC LSB IS SET
3820	021122	004737	027460		68\$:	JSR	PC,GETQO	;CONTINUE
3821	021126	103401				BCC	69\$	;GET HARDWARE TRANSMITTER BCC LSB
3822	021130	104016				HLT	16	;BR IF HARD BCC LSB IS SET
3823	021132				69\$:			;ERROR, HARD BCC LSB IS CLEAR
3824	021132	006037	021102			ROR	66\$	;SHIFT SOFT DATA

```

3825 021136 013737 027344 021104      MOV    CALBCC,67$      ;LOAD OLD SOFT BCC
3826 021144 022700 000010              CMP    #10,R0        ;DONE YET?
3827 021150 001346                    BNE    65$           ;BR IF NOT DONE
3828 021152 104401                    SCOP1          ;SCOPE SUBTEST (SW09=1)
3829 021154 012737 021162 001220      MOV    #71$,LOCK      ;NEW SCOPE1
3830 021162 004737 027706 71$:        JSR    PC,CLR10       ;CLEAR BCC REGISTERS
3831 021166 005000                    CLR    R0          ;START SHIFT COUNTER AT ZERO
3832 021170 012737 120001 027342      MOV    #CRC16,XPOLY   ;LOAD POLYNOMIAL FOR SOFTWARE BCC
3833 021176 012737 000252 021236      MOV    #252,73$;      ;LOAD CHAR FOR SOFTWARE BCC
3834 021204 005037 021240              CLR    74$          ;CLEAR OLD SOFTWARE BCC
3835 021210 004737 027346              JSR    PC,BCCLD      ;LOAD OUT SILO WITH 2 SYNCs
3836 021214 000252                    252          ;AND THE CHARACTER 252
3837 021216 104415 000032              DATACLK, 32        ;GET RECEIVER ACTIVE
3838 021222 104415 000001 72$:        DATACLK, 1          ;SHIFT BCC ONCE
3839 021226 005200                    INC    R0          ;BUMP SHIFT COUNT
3840 021230 004537 027236              JSR    R5,SIMBCC     ;CALCULATE SOFTWARE BCC LSB
3841 021234 000001                    1            ;ONE SHIFT
3842 021236 000000 73$:              0            ;DATA CHARACTER
3843 021240 000000 74$:              0            ;OLD BCC
3844 021242 103405                    BCS    75$          ;BR IF SOFT BCC LSB IS SET
3845 021244 004737 027472              JSR    PC,GETQ1      ;GET HARDWARE RECEIVER BCC LSB
3846 021250 103006                    BCC    76$          ;BR IF HARD BCC LSB IS CLEAR
3847 021252 104013                    HLT    13           ;ERROR, BCC LSB IS SET
3848 021254 000404                    BR     76$          ;CONTINUE
3849 021256 004737 027472 75$:        JSR    PC,GETQ1      ;GET HARDWARE RECEIVER BCC LSB
3850 021262 103401                    BCS    76$          ;BR IF HARD BCC LSB IS SET
3851 021264 104017                    HLT    17           ;ERROR, BCC LSB IS CLEAR
3852 021266                    76$:              ;
3853 021266 006037 021236              ROR    73$          ;SHIFT SOFT DATA
3854 021272 013737 027344 021240      MOV    CALBCC,74$     ;LOAD OLD SOFT BCC
3855 021300 022700 000010              CMP    #10,R0        ;DONE YET?
3856 021304 001346                    BNE    72$           ;BR IF NOT DONE
3857 021306 104401                    SCOP1          ;SCOPE SUBTEST (SW09=1)
3858 021310 104400 77$:              SCOPE          ;SCOPE THIS TEST
3859
3860
3861 ;***** TEST 46 *****
3862 ;*TRANSMITTER CRC TEST
3863 ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK A BINARY
3864 ;*COUNT PATTERN, VERIFY THE LSB OF THE TRANSMITTER BCC ON EACH SHIFT
3865 ;*****
3866
3867 ; TEST 46
3868 ;-----
3869 021312 012737 000046 001226 TST46: MOV    #46,TSTNO
3870 021320 012737 021550 001216      MOV    #TST47,NEXT
3871
3872 021326 104412                    MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
3873 021330 012711 004000              MOV    #BIT11,(R1)   ;MASTER CLEAR DMC11
3874 021334 005003                    CLR    R3          ;SET LINE UNIT LOOP
3875 021336 005004                    CLR    R4          ;ZERO BIT COUNT
3876 021340 005005                    CLR    R5          ;R4 CONTAINS CHAR TO BE LOADED IN SILO
3877 021342 005037 021444              CLR    4$          ;R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
3878 021346 012737 120001 027342      MOV    #CRC16,XPOLY   ;CLEAR SOFT BCC
3879 021354 004737 027510              JSR    PC,SYNLD      ;LOAD POLYNOMIAL
3880 021360 010461 000004              MOV    R4,4(R1)     ;LOAD SILO WITH 2 SYNCs, SOM SET
                                ;PORT4_CHAR

```

3881	021364	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3882	021366	122110			122110		;LOAD OUT DATA
3883	021370	005204			INC	R4	;INCREMENT TO NEXT CHARACTER
3884	021372	010461	000004		MOV	R4,4(R1)	;PORT4 CHAR
3885	021376	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3886	021400	122110			122110		;LOAD OUT DATA
3887	021402	005204			INC	R4	;INCREMENT TO NEXT CHARACTER
3888	021404	010461	000004		MOV	R4,4(R1)	;PORT4 CHAR
3889	021410	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3890	021412	122110			122110		;LOAD OUT DATA
3891	021414	004737	026374		JSR	PC,OCOR	;WAIT FOR OCOR
3892	021420	104415	000021		DATACLK,21		;CLOCK DATA
3893	021424	010537	021442	1\$:	MOV	R5,3\$	;LOAD CHAR FOR SOFT CRC
3894	021430	104415	000001	2\$:	DATACLK,1		;SHIFT BCC ONCE
3895	021434	004537	027236		JSR	R5,SIMBCC	;CALCULATE SOFT BCC
3896	021440	000001			1		;SOFT SHIFT COUNT
3897	021442	000000		3\$:	0		;SOFT CHARACTER
3898	021444	000000		4\$:	0		;OLD SOFT BCC
3899	021446	103405			BCS	5\$	;BR IF SOFT BCC LSB IS SET
3900	021450	004737	027460		JSR	PC,GETQ0	;GET HARDWARE TRANSMITTER BCC LSB
3901	021454	103006			BCC	6\$	;BR IF OK (CLEARED)
3902	021456	104020			HLT	20	;ERROR, BCC LSB WAS SET
3903	021460	000404			BR	6\$	;CONTINUE WITH TEST
3904	021462	004737	027460	5\$:	JSR	PC,GETQ0	;GET HARDWARE TRANSMITTER BCC LSB
3905	021466	103401			BCS	6\$	;BR IF OK (SET)
3906	021470	104021			HLT	21	;ERROR, BCC LSB WAS CLEAR
3907							
3908	021472			6\$:			
3909	021472	006037	021442		ROR	3\$	;SHIFT SOFT DATA
3910	021476	013737	027344	021444	MOV	CALBCC,4\$	;LOAD OLD SOFT BCC
3911	021504	005203			INC	R3	;INCREMENT BIT COUNTER
3912	021506	022703	000010		CMP	#10,R3	;DONE A FULL CHARACTER YET?
3913	021512	001346			BNE	2\$	;BR IF NO
3914	021514	005003			CLR	R3	;RESTART BIT COUNTER
3915	021516	005204			INC	R4	;INCREMENT DATA FOR SILO
3916	021520	022704	000400		CMP	#400,R4	;DONE BINARY COUNT YET?
3917	021524	003404			BLE	9\$	;BR IF YES
3918	021526	010461	000004		MOV	R4,4(R1)	;PORT4 DATA
3919	021532	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3920	021534	122110			122110		;LOAD OUT DATA
3921	021536	005205		9\$:	INC	R5	;INCREMENT DATA
3922	021540	022705	000400		CMP	#400,R5	;DONE BINARY PATTERN YET?
3923	021544	001327			BNE	1\$	;BR IF NO
3924	021546	104400		7\$:	SCOPE		;SCOPE THIS TEST
3925							
3926							
3927							
3928							
3929							
3930							
3931							
3932							
3933							
3934							
3935	021550	012737	000047	001226	*ST47:	MOV	#47,TSTNO
3936	021556	012737	022006	001216		MOV	#TST50,NEXT

***** TEST 47 *****
 ;*RECEIVER CRC TEST
 ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK A BINARY
 ;*COUNT PATTERN, VERIFY THE LSB OF THE RECEIVER BCC ON EACH SHIFT
 ;*****
 ; TEST 47
 ;-----

BASIC RECEIVER TESTS

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

```

3993      ;***** TEST 50 *****
3994      ;*TRANSMITTER DDCMP CRC TEST
3995      ;*THIS TEST TRANSMITS A FOUR CHARACTER MESSAGE WITH CRC
3996      ;*BOTH DATA AND THE BCC ARE VERIFIED IN THE BIT
3997      ;*WINDOW. THE FOUR CHARACTERS ARE 0,125,252,377
3998      ;*THE TRANSMITTER IS CHECKED FOR GOING TO A MARK STATE AFTER THE BCC
3999      ;*****
4000
4001      : TEST 50
4002      :-----
4003      022006 012737 000050 001226 TST50: MOV #50,TSTNO
4004      022014 012737 022340 001216 MOV #TST51,NEXT
4005
4006      022022 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
4007      ;MASTER CLEAR DMC11
4008
4009      ;LOAD OUT DATA SILO
4010      022024 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
4011      022030 012704 030126 MOV #MESDAT,R4 ;LOAD POINTER TO DATA
4012      022034 005037 022130 CLR 10$ ;CLEAR SOFT BCC
4013      022040 012700 000004 MOV #4,R0 ;LOAD CHARACTER COUNT
4014      022044 004737 027510 JSR PC,SYNLD ;LOAD 2 SYNC IN OUT SILO
4015      022050 004737 026526 JSR PC,OUTRDY ;WAIT FOR OUTRDY
4016      022054 004537 027644 JSR R5,MESLD ;LOAD SILO WITH 4 CHAR MESS
4017      022060 030126 MESDAT ;ADDRESS OF MESSAGE
4018      022062 000004 4 ;NUMBER OF CHARACTERS
4019      022064 004737 027620 JSR PC,EOM ;LOAD GARBAGE CHARACTER, WITH EOM SET
4020      022070 004737 026374 JSR PC,OCOR ;WAIT FOR OCOR
4021      022074 005003 CLR R3 ;CLEAR BIT COUNTER
4022      022076 104415 000022 DATACLK,22 ;CLOCK DATA
4023      022102 112405 12$: MOVB (R4)+,R5 ;LOAD R5 WITH CHAR
4024      022104 010502 MOV R5,R2 ;LOAD R2 WITH CHAR
4025
4026      ;CHECK FIRST FOUR CHARACTER MESSAGE
4027      ;IN THE BIT WINDOW (0,125,252,377)
4028
4029      022106 012737 120001 027342 MOV #CRC16,XPOLY ;LOAD POLYNOMIAL
4030      022114 010537 022126 MOV R5,67$ ;LOAD SOFT CHAR FOR BCC
4031      022120 004537 027236 JSR R5,SIMBCC ;CALCULATE SOFT BCC
4032      022124 000010 10 ;SHIFT COUNT
4033      022126 000000 67$: 0 ;CHARACTER
4034      022130 000000 10$: 0 ;OLD BCC
4035      022132 013737 027344 022130 MOV CALBCC,10$ ;LOAD SOFT BCC FOR NEXT SHIFT
4036      022140 104415 000001 64$: DATACLK, 1 ;SHIFT DATA IN TO BIT WINDOW
4037      022144 106002 RORB R2 ;SHIFT SOFT DATA
4038      022146 103005 BCC 65$ ;BR IF A SPACE
4039      022150 004737 026342 JSR PC,GETSI ;LOOK AT BIT WINDOW
4040      022154 103406 BCS 66$ ;BR IF OK (MARK)
4041      022156 104006 HLT 6 ;ERROR, BIT WINDOW WAS A SPACE
4042      022160 000404 BR 66$ ;CONTINUE
4043      022162 004737 026342 65$: JSR PC,GETSI ;LOOK AT BIT WINDOW
4044      022166 103001 BCC 66$ ;BR IF OK (SPACE)
4045      022170 104006 HLT 6 ;ERROR, BIT WINDOW WAS A MARK
4046      022172 66$:
4047      022172 005203 INC R3 ;BUMP BIT COUNTER
4048      022174 022703 000010 CMP #10,R3 ;DONE FULL 8 BITS YET

```

```

4049 022200 001357      BNE      64$      ;BR IF NO
4050 022202 005003      CLR      R3      ;CLEAR BIT COUNTER
4051 022204 005300      DEC      R0      ;DEC CHARACTER COUNT
4052 022206 001335      BNE      12$      ;BR IF NOT DONE YET
4053
4054      ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4055
4056 022210 013700 027344 68$: MOV      CALBCC,R0      ;PUT BCC IN R0
4057 022214 104415 000001 DATACLK,1      ;SHIFT HARDWARE BCC
4058 022220 006000      ROR      R0      ;SHIFT SOFT BCC
4059 022222 103005      BCC      69$      ;BR IF CARRY CLEAR
4060 022224 004737 026342 JSR      PC,GETSI      ;LOOK AT BIT WINDOW
4061 022230 103406      BCS      70$      ;BR IF OK (MARK)
4062 022232 104014      HLT      14      ;ERROR, CRC WRONG (SPACE)
4063 022234 000404      BR      70$      ;CONTINUE
4064 022236 004737 026342 69$: JSR      PC,GETSI      ;LOOK AT BIT WINDOW
4065 022242 103001      BCC      70$      ;BR IF OK (SPACE)
4066 022244 104014      HLT      14      ;ERROR, CRC WRONG (MARK)
4067 022246
4068 022246 005203      70$: INC      R3      ;BUMP BIT COUNTER
4069 022250 022703 000020 CMP      #20,R3      ;FINISHED BCC YET?
4070 022254 001357      BNE      68$      ;BR IF NO
4071 022256 005003      CLR      R3      ;CLEAR BIT COUNTER
4072
4073      ;CHECK TO SEE IF TRANSMITTER IS MARKING
4074
4075 022260 104415 000001 2$: DATACLK,1      ;CLOCK TRANSMITTER
4076 022264 004737 026342 JSR      PC,GETSI      ;LOOK AT WINDOW
4077 022270 103401      BCS      3$      ;IT SHOULD BE MARKING
4078 022272 104024      HLT      24      ;ERROR, BIT WAS A SPACE
4079 022274 005203      3$: INC      R3      ;BUMP BIT COUNTER
4080 022276 022703 000007 CMP      #7,R3      ;DONE YET
4081 022302 001366      BNE      2$      ;BR IF NO
4082 022304 104415 000010 DATACLK,10      ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
4083 022310 005003      CLR      R3      ;CLEAR BIT COUNTER
4084 022312 104415 000001 4$: DATACLK,1      ;SHIFT OUT NEXT BIT
4085 022316 004737 026342 JSR      PC,GETSI      ;LOOK AT BIT WINDOW
4086 022322 103401      BCS      +4      ;BR IF IT IS A MARK
4087 022324 104024      HLT      24      ;ERROR, TRANSMITTER IS NOT MARKING
4088 022326 005203      INC      R3      ;INC BIT COUNT
4089 022330 022703 000020 CMP      #20,R3      ;DONE YET?
4090 022334 001366      BNE      4$      ;BR IF NO
4091 022336 104400      5$: SCOPE      ;SCOPE THIS TEST
4092
4093
4094      ;***** TEST 51 *****
4095      ;*RECEIVER DDCMP CRC TEST
4096      ;*THIS TEST CLOCKS A FOUR CHARACTER MESSAGE WITH BCC
4097      ;*AND VERIFYS CORRECT DATA RECEPTION AND BCC MATCH.
4098      ;*IN BCC MATCH IS CHECKED TO SEE THAT IT IS NOT ASSERTED
4099      ;*UNTIL THE LAST HALF OF THE CRC IS RECEIVED
4100      ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4101      ;*****
4102
4103      ; TEST 51:
4104      ;-----

```

```
4105 022340 012737 000051 001226 TST51: MOV #51,TSTNO
4106 022346 012737 022566 001216 MOV #TST52,NEXT
4107
4108 022354 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
4109 022356 012711 004000 MOV #BIT11,(R1) ;MASTER CLEAR DMC11
4110 022362 012702 030126 MOV #MESDAT,R2 ;SET LINE UNIT LOOP
4111 022366 012700 000004 MOV #4,R0 ;LOAD POINTER TO DATA
4112 022372 004737 027510 JSR PC,SYNLD ;LOAD CHARACTER COUNT
4113 022376 004737 026526 JSR PC,OUTRDY ;LOAD 2 SYNCs IN OUT SILO
4114 022402 004537 027644 JSR R5,MESLD ;WAIT FOR OUTRDY
4115 022406 030126 MESDAT ;LOAD SILO WITH 4 CHAR MESS
4116 022410 000004 4 ;ADDRESS OF MESSAGE
4117 022412 004737 027620 JSR PC,EOM ;NUMBER OF CHARACTERS
4118 022416 004737 026374 JSR PC,OCOR ;LOAD GARBAGE CHARACTER, WITH EOM SET
4119 022422 104415 000114 DATACLK,114 ;WAIT FOR OCOR
4120 022426 004737 027202 3$: JSR PC,INRDY ;CLOCK DATA
4121 022432 104414 ROMCLK ;WAIT FOR INRDY
4122 022434 021204 021204 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4123 022436 016104 000004 MOV 4(R1),R4 ;GET IN DATA
4124 022442 112205 MOVB (R2)+,R5 ;PUT 'FOUND' IN R4
4125 022444 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
4126 022446 001401 BEQ 1$ ;COMPARE RECEIVED DATA
4127 022450 104010 HLT 10 ;BR IF OK
4128 022452 1$: ;DATA ERROR
4129 022452 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4130 022454 021244 021244 MOV 4(R1),R4
4131 022456 016104 000004 BIC #376,R4 ;PUT 'FOUND' IN R4
4132 022462 042704 000376 CLR R5 ;CLEAR UNWANTED BITS
4133 022466 005005 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
4134 022470 120504 BEQ 4$ ;IS IN BCC MATCH STILL CLEAR?
4135 022472 001401 HLT 15 ;IN BCC MATCH ERROR
4136 022474 104015 4$: DEC R0 ;DEC CHARACTER COUNT
4137 022476 005300 BNE 3$ ;BR IF NOT DONE YET
4138 022500 001352
4139
4140 ;CHECK TO SEE THAT IN BCC MATCH IS SET
4141
4142 022502 004737 027202 JSR PC,INRDY ;WAIT FOR INRDY
4143 022506 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4144 022510 021204 021204 ;GET FIRST HALF OF CRC
4145 022512 116137 000004 001252 MOVB 4(R1),TEMP3 ;PUT IN TEMP3
4146 022520 042737 177400 001252 BIC #177400,TEMP3 ;CLEAR HI BYTE
4147 022526 004737 027202 JSR PC,INRDY ;WAIT FOR INRDY
4148 022532 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4149 022534 021244 021244 MOV 4(R1),R4
4150 022536 016104 000004 BIC #376,R4 ;PUT 'FOUND' IN R4
4151 022542 042704 000376 MOV #1,R5 ;CLEAR UNWANTED BITS
4152 022546 012705 000001 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
4153 022552 120504 BEQ 25$ ;IS IN BCC MATCH SET?
4154 022554 001401 HLT 15 ;IN BCC MATCH ERROR
4155 022556 104015 25$:
4156 022560 25$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4157 022560 104414 021204 ;GET LAST HALF
4158 022562 021204 2$: SCOPE ;SCOPE THIS TEST
4159 022564 104400
4160
```

```

4161
4162
4163
4164
4165
4166
4167
4168
4169
4170
4171
4172
4173
4174
4175 022566 012737 000052 001226 TST52:
4176 022574 012737 023666 001216
4177
4178 022602 104412
4179
4180
4181
4182 022604 012711 004000
4183 022610 012704 030126
4184 022614 005037 022724
4185 022620 012700 000004
4186 022624 004737 027510
4187 022630 004737 026526
4188 022634 004537 027644
4189 022640 030126
4190 022642 000004
4191 022644 004737 027620
4192 022650 004537 027644
4193 022654 030126
4194 022656 000004
4195 022660 004737 027620
4196 022664 004737 026374
4197 022670 005003
4198 022672 104415 000022
4199 022676 112405
4200 022700 010502
4201
4202
4203
4204
4205 022702 012737 120001 027342
4206 022710 010537 022722
4207 022714 004537 027236
4208 022720 000010
4209 022722 000000
4210 022724 000000
4211 022726 013737 027344 022724
4212 022734 104415 000001
4213 022740 106002
4214 022742 103005
4215 022744 004737 026342
4216 022750 103406

;***** TEST 52 *****
;DDCMP EOM FUNCTION TEST
;THIS TEST LOADS OUT SILO WITH: 2 SYNCs, 4 CHAR MESSAGE, EOM
;*4 CHARACTER MESS, EOM. THE DATA STREAM IS CHECKED TO BE
;*4 CHAR, BCC, 4 CHAR, BCC, MARKS. THIS TEST VERIFYS THAT
;*THE CHARCTERS LOADED WITH EOM SET ARE LOST
;*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
;*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
;*****

; TEST 52
;-----
MOV #52, TSTNO
MOV #TST53, NEXT
;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
MSTCLR
;LOAD OUT DATA SILO
MOV #BIT11, (R1) ;SET LINE UNIT LOOP
MOV #MESDAT, R4 ;LOAD POINTER TO DATA
CLR 10$ ;CLEAR SOFT BCC
MOV #4, R0 ;LOAD CHARACTER COUNT
JSR PC, SYNLD ;LOAD 2 SYNCs IN OUT SILO
JSR PC, OUTRDY ;WAIT FOR OUTRDY
JSR R5, MESLD ;LOAD SILO WITH 4 CHAR MESS
MESDAT ;ADDRESS OF MESSAGE
4 ;NUMBER OF CHARACTERS
JSR PC, EOM ;LOAD GARBAGE CHARACTER, WITH EOM SET
JSR R5, MESLD ;LOAD FOUR MORE CHARACTERS
MESDAT ;ADDRESS OF MESSAGE
4 ;NUMBER OF CHACTERS
JSR PC, EOM ;SET EOM
JSR PC, OCOR ;WAIT FOR OCOR
CLR R3 ;CLEAR BIT COUNTER
DATA CLK, 22 ;CLOCK DATA
12$: MOVB (R4)+, R5 ;LOAD R5 WITH CHAR
MOV R5, R2 ;LOAD R2 WITH CHAR

;CHECK FIRST FOUR CHARACTER MESSAGE
;IN THE BIT WINDOW (0,125,252,377)
MOV #CRC16, XPOLY ;LOAD POLYNOMIAL
MOV R5, 67$ ;LOAD SOFT CHAR FOR BCC
JSR R5, SIMBCC ;CALCULATE SOFT BCC
10 ;SHIFT COUNT
0 ;CHARACTER
10$: 0 ;OLD BCC
MOV CALBCC, 10$ ;LOAD SOFT BCC FOR NEXT SHIFT
DATA CLK, 1 ;SHIFT DATA IN TO BIT WINDOW
RORB R2 ;SHIFT SOFT DATA
BCC 65$ ;BR IF A SPACE
JSR PC, GETSI ;LOOK AT BIT WINDOW
BCS 66$ ;BR IF OK (MARK)

```

4217	022752	104006			HLT	6	;ERROR, BIT WINDOW WAS A SPACE
4218	022754	000404			BR	66\$	;CONTINUE
4219	022756	004737	026342	65\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
4220	022762	103001			BCC	66\$	;BR IF OK (SPACE)
4221	022764	104006			HLT	6	;ERROR, BIT WINDOW WAS A MARK
4222	022766			66\$:			
4223	022766	005203			INC	R5	;BUMP BIT COUNTER
4224	022770	022703	000010		CMP	#10,R3	;DONE FULL 8 BITS YET
4225	022774	001357			BNE	64\$	;BR IF NO
4226	022776	005003			CLR	R3	;CLEAR BIT COUNTER
4227	023000	005300			DEC	R0	;DEC CHARACTER COUNT
4228	023002	001335			BNE	12\$	;BR IF NOT DONE YET
4229							
4230							;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4231							
4232	023004	013700	027344		MOV	CALBCC,R0	;PUT BCC IN R0
4233	023010	104415	000001	68\$:	DATACLK,	1	;SHIFT HARDWARE BCC
4234	023014	006000			ROR	R0	;SHIFT SOFT BCC
4235	023016	103005			BCC	69\$	;BR IF CARRY CLEAR
4236	023020	004737	026342		JSR	PC,GETSI	;LOOK AT BIT WINDOW
4237	023024	103406			BCS	70\$	;BR IF OK (MARK)
4238	023026	104014			HLT	14	;ERROR, CRC WRONG (SPACE)
4239	023030	000404			BR	70\$	;CONTINUE
4240	023032	004737	026342	69\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
4241	023036	103001			BCC	70\$	;BR IF OK (SPACE)
4242	023040	104014			HLT	14	;ERROR, CRC WRONG (MARK)
4243	023042			70\$:			
4244	023042	005203			INC	R3	;BUMP BIT COUNTER
4245	023044	022703	000020		CMP	#20,R3	;FINISHED BCC YET?
4246	023050	001357			BNE	68\$	;BR IF NO
4247	023052	005003			CLR	R3	;CLEAR BIT COUNTER
4248	023054	012700	000004		MOV	#4,R0	;RESET CHARACTER COUNTER
4249	023060	012704	030126		MOV	#MESDAT,R4	;LOAD MESSAGE POINTER
4250	023064	005037	023116		CLR	11\$	;CLR SOFT BCC
4251	023070	112405		13\$:	MOVB	(R4)+,R5	;LOAD CHAR IN R5
4252	023072	010502			MOV	R5,R2	;LOAD CHAR IN R2
4253							
4254							;CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
4255							
4256	023074	012737	120001	027342	MOV	#CRC16,XPOLY	;LOAD POLYNOMIAL
4257	023102	010537	023114		MOV	R5,76\$	;LOAD SOFT CHAR FOR BCC
4258	023106	004537	027236		JSR	R5,SIMBCC	;CALCULATE SOFT BCC
4259	023112	000010			10		;SHIFT COUNT
4260	023114	000000		76\$:	0		;CHARACTER
4261	023116	000000		11\$:	0		;OLD BCC
4262	023120	013737	027344	023116	MOV	CALBCC,11\$	;LOAD SOFT BCC FOR NEXT SHIFT
4263	023126	104415	000001	73\$:	DATACLK,	1	;SHIFT DATA IN TO BIT WINDOW
4264	023132	106002			RORB	R2	;SHIFT SOFT DATA
4265	023134	103005			BCC	74\$	;BR IF A SPACE
4266	023136	004737	026342		JSR	PC,GETSI	;LOOK AT BIT WINDOW
4267	023142	103406			BCS	75\$	;BR IF OK (MARK)
4268	023144	104006			HLT	6	;ERROR, BIT WINDOW WAS A SPACE
4269	023146	000404			BR	75\$	;CONTINUE
4270	023150	004737	026342	74\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
4271	023154	103001			BCC	75\$	;BR IF OK (SPACE)
4272	023156	104006			HLT	6	;ERROR, BIT WINDOW WAS A MARK

CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 82
 CZDMEC.P11 11-SEP-78 12:38 BASIC RECEIVER TESTS

SEQ 0082

```

4273 023160          75$: INC R3          ;BUMP BIT COUNTER
4274 023160 005203    CMP #10,R3       ;DONE FULL 8 BITS YET
4275 023162 022703 000010 BNE 73$      ;BR IF NO
4276 023166 001357    CLR R3          ;CLEAR BIT COUNTER
4277 023170 005003    DEC R0          ;DEC CHARACTER COUNT
4278 023172 005300    BNE 13$         ;BR IF NOT DONE YET
4279 023174 001335
4280
4281          ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4282
4283 023176 013700 027344 MOV CALBCC,R0 ;PUT BCC IN R0
4284 023202 104415 000001 77$: DATACLK,1 ;SHIFT HARDWARE BCC
4285 023206 006000    ROR R0          ;SHIFT SOFT BCC
4286 023210 103005    BCC 78$        ;BR IF CARRY CLEAR
4287 023212 004737 026342 JSR PC,GETSI ;LOOK AT BIT WINDOW
4288 023216 103406    BCS 79$        ;BR IF OK (MARK)
4289 023220 104014    HLT 14         ;ERROR, CRC WRONG (SPACE)
4290 023222 000404    BR 79$         ;CONTINUE
4291 023224 004737 026342 78$: JSR PC,GETSI ;LOOK AT BIT WINDOW
4292 023230 103001    BCC 79$        ;BR IF OK (SPACE)
4293 023232 104014    HLT 14         ;ERROR, CRC WRONG (MARK)
4294 023234
4295 023234 005203    79$: INC R3          ;BUMP BIT COUNTER
4296 023236 022703 000020 CMP #20,R3 ;FINISHED BCC YET?
4297 023242 001357    BNE 77$        ;BR IF NO
4298 023244 005003    CLR R3          ;CLEAR BIT COUNTER
4299
4300          ;CHECK TO SEE IF TRANSMITTER IS MARKING
4301
4302 023246 104415 000001 2$: DATACLK,1 ;CLOCK TRANSMITTER
4303 023252 004737 026342 JSR PC,GETSI ;LOOK AT WINDOW
4304 023256 103401    BCS 3$         ;IT SHOULD BE MARKING
4305 023260 104024    HLT 24         ;ERROR, BIT WAS A SPACE
4306 023262 005203    3$: INC R3          ;BUMP BIT COUNTER
4307 023264 022703 000007 CMP #7,R3 ;DONE YET
4308 023270 001366    BNE 2$         ;BR IF NO
4309 023272 104415 000010 DATACLK,10 ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
4310 023276 005003    CIR R3          ;CLEAR BIT COUNTER
4311 023300 104415 000001 4$: DATACLK,1 ;SHIFT OUT NEXT BIT
4312 023304 004737 026342 JSR PC,GETSI ;LOOK AT BIT WINDOW
4313 023310 103401    BCS +4         ;BR IF IT IS A MARK
4314 023312 104024    HLT 24         ;ERROR, TRANSMITTER IS NOT MARKING
4315 023314 005203    INC R3          ;INC BIT COUNT
4316 023316 022703 000020 CMP #20,R3 ;DONE YET?
4317 023322 001366    BNE 4$         ;BR IF NO
4318
4319          ;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
4320          ;WAS RECEIVED CORRECTLY (0,125,252,377)
4321
4322 023324 104415 000001 DATACLK,1 ;GET LAST BIT IN RECEIVER
4323 023330 012703 000004 MOV #4,R3 ;R3=CHARACTER COUNT
4324 023334 012702 030126 MOV #MESDAT,R2 ;LOAD MESSAGE POINTER IN R2
4325 023340 004737 027202 JSR PC,INRDY ;WAIT FOR INRDY
4326 023344 104414    ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
4327 023346 021204
4328 023350 016104 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4

```

4329	023354	112205			MOVB	(R2)+,R5	;PUT 'EXPECTED' IN R5
4330	023356	120504			CMPS	R5,R4	;IS RECEIVED DATA CORRECT?
4331	023360	001401			BEQ	41\$	;BR IF YES
4332	023362	104010			HLT	10	;RECEIVE DATA ERROR
4333	023364	005303			DEC	R3	;DEC CHARACTER COUNT
4334	023366	001364			BNE	40\$	;BR IF NOT DONE YET
4335							
4336							;CHECK TO SEE THAT IN BCC MATCH IS SET
4337							;AND THAT THE BCC WAS RECEIVED CORRECTLY
4338							
4339	023370	004737	027202		JSR	PC,INRDY	;WAIT FOR INRDY
4340	023374	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4341	023376	021204			021204		;GET FIRST HALF OF CRC
4342	023400	116137	000004	001252	MOVB	4(R1),TEMP3	;PUT IN TEMP3
4343	023406	042737	177400	001252	BIC	#177400,TEMP3	;CLEAR HI BYTE
4344	023414	004737	027202		JSR	PC,INRDY	;WAIT FOR INRDY
4345	023420	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4346	023422	021244			021244		
4347	023424	016104	000004		MOV	4(R1),R4	;PUT 'FOUND' IN R4
4348	023430	042704	000376		BIC	#376,R4	;CLEAR UNWANTED BITS
4349	023434	012705	000001		MOV	#1,R5	;PUT 'EXPECTED' IN R5
4350	023440	120504			CMPS	R5,R4	;IS IN BCC MATCH SET?
4351	023442	001401			BEQ	50\$	
4352	023444	104015			HLT	15	;IN BCC MATCH ERROR
4353	023446						
4354	023446	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4355	023450	021204			021204		;GET LAST HALF
4356	023452	116137	000004	001251	MOVB	4(R1),TEMP2+1	;PUT IN TEMP2
4357	023460	042737	000377	001250	BIC	#377,TEMP2	;CLEAR LO BYTE
4358	023466	053737	001250	001252	BIS	TEMP2,TEMP3	;16 BIT BCC NOW IN TEMP3
4359	023474	023737	027344	001252	CMP	CALBCC,TEMP3	;IS IT CORRECT?
4360	023502	001401			BEQ	42\$	;BR IF OK
4361	023504	104027			HLT	27	
4362							
4363							;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
4364							;WAS RECEIVED CORRECTLY (0,125,252,377)
4365							
4366	023506	012703	000004		MOV	#4,R3	;R3=CHARACTER COUNT
4367	023512	012702	030126		MOV	#MESDAT,R2	;LOAD MESSAGE POINTER IN R2
4368	023516	004737	027202		JSR	PC,INRDY	;WAIT FOR INRDY
4369	023522	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4370	023524	021204			021204		
4371	023526	016104	000004		MOV	4(R1),R4	;PUT 'FOUND' IN R4
4372	023532	112205			MOVB	(R2)+,R5	;PUT 'EXPECTED' IN R5
4373	023534	120504			CMPS	R5,R4	;IS RECEIVED DATA CORRECT?
4374	023536	001401			BEQ	44\$	;BR IF YES
4375	023540	104010			HLT	10	;RECEIVE DATA ERROR
4376	023542	005303			DEC	R3	;DEC CHARACTER COUNT
4377	023544	001364			BNE	43\$	;BR IF NOT DONE YET
4378							
4379							;CHECK TO SEE THAT IN BCC MATCH IS SET
4380							;AND THAT THE BCC WAS RECEIVED CORRECTLY
4381							
4382	023546	004737	027202		JSR	PC,INRDY	;WAIT FOR INRDY
4383	023552	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4384	023554	021204			021204		;GET FIRST HALF OF CRC

CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 84
 CZDMEC.P11 11-SEP-78 12:38 BASIC RECEIVER TESTS

SEQ 0084

```

4385 023556 116137 000004 001252      MOVB 4(R1),TEMP3      ;PUT IN TEMP3
4386 023564 042737 177400 001252      BIC #177400,TEMP3    ;CLEAR HI BYTE
4387 023572 004737 027202              JSR PC,INRDY         ;WAIT FOR INRDY
4388 023576 104414              ROMCLK 021244          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4389 023600 021244              MOV 4(R1),R4           ;PUT 'FOUND' IN R4
4390 023602 016104 000004      BIC #376,R4          ;CLEAR UNWANTED BITS
4391 023606 042704 000376      MOV #1,R5             ;PUT 'EXPECTED' IN R5
4392 023612 012705 000001      CMPB R5,R4           ;IS IN BCC MATCH SET?
4393 023616 120504      BEQ 51$              ;IN BCC MATCH ERROR
4394 023620 001401      HLT 15
4395 023622 104015      51$: ROMCLK 021204          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4396 023624              021204          ;GET LAST HALF
4397 023624 104414      MOVB 4(R1),TEMP2+1        ;PUT IN TEMP2
4398 023626 021204      BIC #377,TEMP2          ;CLEAR LO BYTE
4399 023630 116137 000004 001251      BIS TEMP2,TEMP3      ;16 BIT BCC NOW IN TEMP3
4400 023636 042737 000377 001250      CMP CALBCC,TEMP3    ;IS IT CORRECT?
4401 023644 053737 001250 001252      BEQ 5$              ;BR IF OK
4402 023652 023737 027344 001252      HLT 27
4403 023660 001401      5$: SCOPE                  ;SCOPE THIS TEST
4404 023662 104027
4405 023664 104400
4406
4407
4408      ;***** TEST 53 *****
4409      ;*DDCMP EOM FUNCTION TEST
4410      ;*THIS TEST LOADS OUT SILO WITH: 2 SYNCs, 4 CHAR MESSAGE, EOM
4411      ;*SOM, 4 CHAR MESS, EOM. THE DATA STREAM IS CHECKED TO BE
4412      ;*4 CHAR, BCC, 4 CHAR, BCC, MARKS. THIS TEST VERIFYS THAT
4413      ;*THE CHARACTERS LOADED WITH EOM SET ARE LOST
4414      ;*ALSO THAT THE CHAR LOADED WITH SOM IS NOT IN THE BCC
4415      ;*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
4416      ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4417      ;*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
4418      ;*****
4419
4420      ; TEST 53
4421      ;-----
4422 023666 012737 000053 001226 TST53: MOV #53,TSTNO
4423 023674 012737 025066 001216      MOV #TST54,NEXT
4424
4425 023702 104412      MSTCLR              ;R1 CONTAINS BASE DMC11 ADDRESS
4426
4427      ;LOAD OUT DATA SILO
4428
4429 023704 012711 004000      MOV #BIT11,(R1)      ;SET LINE UNIT LOOP
4430 023710 012704 030126      MOV #MESDAT,R4      ;LOAD POINTER TO DATA
4431 023714 005037 024030      CLR 10$          ;CLEAR SOFT BCC
4432 023720 012700 000004      MOV #4,R0           ;LOAD CHARACTER COUNT
4433 023724 004737 027510      JSR PC,SYNLD        ;LOAD 2 SYNCs IN OUT SILO
4434 023730 004737 026526      JSR PC,OUTRDY       ;WAIT FOR OUTRDY
4435 023734 004537 027644      JSR R5,MESLD        ;LOAD SILO WITH 4 CHAR MESS
4436 023740 030126      MESDAT              ;ADDRESS OF MESSAGE
4437 023742 000004      4              ;NUMBER OF CHARACTERS
4438 023744 004737 027620      JSR PC,EOM          ;LOAD GARBAGE CHARACTER, WITH EOM SET
4439 023750 004737 027570      JSR PC,SOM          ;LOAD GARBAGE CHAR WITH SOM SET
4440 023754 004537 027644      JSR R5,MESLD        ;LOAD FOUR MORE CHARACTERS

```

4441	023760	030126			MESDAT	:ADDRESS OF MESSAGE
4442	023762	000004			4	:NUMBER OF CHACTERS
4443	023764	004737	027620		JSR PC,EOM	:SET EOM
4444	023770	004737	026374		JSR PC,OCOR	:WAIT FOR OCOR
4445	023774	005003			CLR R3	:CLEAR BIT COUNTER
4446	023776	104415	000022		DATACLK,22	:CLOCK DATA
4447	024002	112405		12\$:	MOVB (R4)+,R5	:LOAD R5 WITH CHAR
4448	024004	010502			MOV R5,R2	:LOAD R2 WITH CHAR
4449						
4450					:CHECK FIRST FOUR CHARACTER MESSAGE	
4451					:IN THE BIT WINDOW (0,125,252,377)	
4452						
4453	024006	012737	120001	027342	MOV #CRC16,XPOLY	:LOAD POLYNOMIAL
4454	024014	010537	024026		MOV R5,67\$	:LOAD SOFT CHAR FOR BCC
4455	024020	004537	027236		JSR R5,SIMBCC	:CALCULATE SOFT BCC
4456	024024	000010			10	:SHIFT COUNT
4457	024026	000000		67\$:	0	:CHARACTER
4458	024030	000000		10\$:	0	:OLD BCC
4459	024032	013737	027344	024030	MOV CALBCC,10\$	:LOAD SOFT BCC FOR NEXT SHIFT
4460	024040	104415	000001		DATACLK,1	:SHIFT DATA IN TO BIT WINDOW
4461	024044	106002		64\$:	ROR R2	:SHIFT SOFT DATA
4462	024046	103005			BCC 65\$	:BR IF A SPACE
4463	024050	004737	026342		JSR PC,GETSI	:LOOK AT BIT WINDOW
4464	024054	103406			BCC 66\$	:BR IF OK (MARK)
4465	024056	104006			HLT 6	:ERROR, BIT WINDOW WAS A SPACE
4466	024060	000404			BR 66\$	:CONTINUE
4467	024062	004737	026342		JSR PC,GETSI	:LOOK AT BIT WINDOW
4468	024066	103001		65\$:	BCC 66\$	:BR IF OK (SPACE)
4469	024070	104006			HLT 6	:ERROR, BIT WINDOW WAS A MARK
4470	024072			66\$:		
4471	024072	005203			INC R3	:BUMP BIT COUNTER
4472	024074	022703	000010		CMP #10,R3	:DONE FULL 8 BITS YET
4473	024100	001357			BNE 64\$	:BR IF NO
4474	024102	005003			CLR R3	:CLEAR BIT COUNTER
4475	024104	005300			DEC R0	:DEC CHARACTER COUNT
4476	024106	001335			BNE 12\$	:BR IF NOT DONE YET
4477						
4478					:CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW	
4479						
4480	024110	013700	027344		MOV CALBCC,R0	:PUT BCC IN R0
4481	024114	104415	000001		DATACLK,1	:SHIFT HARDWARE BCC
4482	024120	006000		68\$:	ROR R0	:SHIFT SOFT BCC
4483	024122	103005			BCC 69\$	:BR IF CARRY CLEAR
4484	024124	004737	026342		JSR PC,GETSI	:LOOK AT BIT WINDOW
4485	024130	103406			BCC 70\$	:BR IF OK (MARK)
4486	024132	104014			HLT 14	:ERROR, CRC WRONG (SPACE)
4487	024134	000404			BR 70\$	:CONTINUE
4488	024136	004737	026342		JSR PC,GETSI	:LOOK AT BIT WINDOW
4489	024142	103001		69\$:	BCC 70\$	:BR IF OK (SPACE)
4490	024144	104014			HLT 14	:ERROR, CRC WRONG (MARK)
4491	024146			70\$:		
4492	024146	005203			INC R3	:BUMP BIT COUNTER
4493	024150	022703	000020		CMP #20,R3	:FINISHED BCC YET?
4494	024154	001357			BNE 68\$	:BR IF NO
4495	024156	005003			CLR R3	:CLEAR BIT COUNTER
4496						

CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 86
 CZDMEC.211 11-SEP-78 12:38 BASIC RECEIVER TESTS

SEQ 0086

```

4497                                     :CHECK CHARACTER LOADED WITH SOM (000), IN THE BIT WINDOW
4498
4499 024160 005005                      CLR      R5                      :CHARACTER LOADED WITH SOM
4500 024162 010502                      MOV      R5,R2                      :LOAD R2 WITH CHAR
4501 024164 104415 000001              32$: DATACLK, 1                      :CLOCK TRANSMITTER
4502 024170 106002                      RORB     R2                      :SHIFT SOFT DATA
4503 024172 103005                      BCC      30$                      :BR IF SPACE
4504 024174 004737 026342              JSR      PC,GETSI                    :LOOK AT BIT WINDOW
4505 024200 103406                      BCS      31$                      :BR IF OK (MARK)
4506 024202 104006                      HLT      6                      :ERROR,BIT WINDOW WAS A SPACE
4507 024204 000404                      BR       31$                      :CONTINUE
4508 024206 004737 026342              30$: JSR      PC,GETSI                    :LOOK AT BIT WINDOW
4509 024212 103001                      BCC      31$                      :BR IF OK (SPACE)
4510 024214 104006                      HLT      6                      :ERROR,BIT WINDOW WAS A MARK
4511 024216 005203                      31$: INC      R3                      :BUMP BIT COUNTER
4512 024220 022703 000010              CMP      #10,R3                    :DONE CHARACTER YET?
4513 024224 001357                      BNE      32$                      :BR IF NO
4514 024226 005003                      CLR      R3                      :RESET BIT COUNTER
4515 024230 012700 000004              MOV      #4,R0                      :RESET CHARACTER COUNTER
4516 024234 012704 030126              MOV      #MESDAT,R4                  :LOAD MESSAGE POINTER
4517 024240 005037 024272              CLR      11$                      :CLR SOFT BCC
4518 024244 112405                      13$: MOVB    (R4)+,R5                  :LOAD CHAR IN R5
4519 024246 010502                      MOV      R5,R2                      :LOAD CHAR IN R2
4520
4521                                     :CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
4522
4523 024250 012737 120001 027342          MOV      #CRC16,XPOLY                :LOAD POLYNOMIAL
4524 024256 010537 024270              MOV      R5,76$                      :LOAD SOFT CHAR FOR BCC
4525 024262 004537 027236              JSR      R5,SIMBCC                    :CALCULATE SOFT BCC
4526 024266 000010                      10                      :SHIFT COUNT
4527 024270 000000                      76$: 0                      :CHARACTER
4528 024272 000000                      11$: 0                      :OLD BCC
4529 024274 013737 027344 024272          MOV      CALBCC,11$                  :LOAD SOFT BCC FOR NEXT SHIFT
4530 024302 104415 000001              73$: DATACLK, 1                      :SHIFT DATA IN TO BIT WINDOW
4531 024306 106002                      RORB     R2                      :SHIFT SOFT DATA
4532 024310 103005                      BCC      74$                      :BR IF A SPACE
4533 024312 004737 026342              JSR      PC,GETSI                    :LOOK AT BIT WINDOW
4534 024316 103406                      BCS      75$                      :BR IF OK (MARK)
4535 024320 104006                      HLT      6                      :ERROR, BIT WINDOW WAS A SPACE
4536 024322 000404                      BR       75$                      :CONTINUE
4537 024324 004737 026342              74$: JSR      PC,GETSI                    :LOOK AT BIT WINDOW
4538 024330 103001                      BCC      75$                      :BR IF OK (SPACE)
4539 024332 104006                      HLT      6                      :ERROR, BIT WINDOW WAS A MARK
4540 024334                      75$:
4541 024334 005203                      INC      R3                      :BUMP BIT COUNTER
4542 024336 022703 000010              CMP      #10,R3                    :DONE FULL 8 BITS YET
4543 024342 001357                      BNE      73$                      :BR IF NO
4544 024344 005003                      CLR      R3                      :CLEAR BIT COUNTER
4545 024346 005300                      DEC      R0                      :DEC CHARACTER COUNT
4546 024350 001335                      BNE      13$                      :BR IF NOT DONE YET
4547
4548                                     :CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4549
4550 024352 013700 027344              MOV      CALBCC,R0                  :PUT BCC IN R0
4551 024356 104415 000001              77$: DATACLK, 1                      :SHIFT HARDWARE BCC
4552 024362 006000                      ROR      R0                      :SHIFT SOFT BCC

```

```

4553 024364 103005          BCC      78$      ;BR IF CARRY CLEAR
4554 024366 004737 026342   JSR      PC,GETSI  ;LOOK AT BIT WINDOW
4555 024372 103406          BCS      79$      ;BR IF OK (MARK)
4556 024374 104014          HLT      14        ;ERROR, CRC WRONG (SPACE)
4557 024376 000404          BR       79$      ;CONTINUE
4558 024400 004737 026342   78$: JSR      PC,GETSI  ;LOOK AT BIT WINDOW
4559 024404 103001          BCC      79$      ;BR IF OK (SPACE)
4560 024406 104014          HLT      14        ;ERROR, CRC WRONG (MARK)
4561 024410          79$:
4562 024410 005203          INC      R3        ;BUMP BIT COUNTER
4563 024412 022703 000020   CMP      #20,R3  ;FINISHED BCC YET?
4564 024416 001357          BNE      77$      ;BR IF NO
4565 024420 005003          CLR      R3        ;CLEAR BIT COUNTER
4566
4567          ;CHECK TO SEE IF TRANSMITTER IS MARKING
4568
4569 024422 104415 000001   2$: DATACLK, 1      ;CLOCK TRANSMITTER
4570 024426 004737 026342   JSR      PC,GETSI  ;LOOK AT WINDOW
4571 024432 103401          BCS      3$      ;IT SHOULD BE MARKING
4572 024434 104024          HLT      24        ;ERROR, BIT WAS A SPACE
4573 024436 005203          3$: INC      R3        ;BUMP BIT COUNTER
4574 024440 022703 000007   CMP      #7,R3  ;DONE YET
4575 024444 001366          BNE      2$      ;BR IF NO
4576 024446 104415 000010   DATACLK, 10      ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
4577 024452 005003          CLR      R3        ;CLEAR BIT COUNTER
4578 024454 104415 000001   4$: DATACLK, 1      ;SHIFT OUT NEXT BIT
4579 024460 004737 026342   JSR      PC,GETSI  ;LOOK AT BIT WINDOW
4580 024464 103401          BCS      +4        ;BR IF IT IS A MARK
4581 024466 104024          HLT      24        ;ERROR, TRANSMITTER IS NOT MARKING
4582 024470 005203          INC      R3        ;INC BIT COUNT
4583 024472 022703 000020   CMP      #20,R3  ;DONE YET?
4584 024476 001366          BNE      4$      ;BR IF NO
4585
4586          ;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
4587          ;WAS RECEIVED CORRECTLY (0,125,252,377)
4588
4589 024500 104415 000001   DATACLK, 1      ;GET LAST BIT IN RECEIVER
4590 024504 012703 000004   MOV      #4,R3  ;R3=CHARACTER COUNT
4591 024510 012702 030126   MOV      #MESDAT,R2 ;LOAD MESSAGE POINTER IN R2
4592 024514 004737 027202   40$: JSR      PC,INRDY  ;WAIT FOR INRDY
4593 024520 104414          ROMCLK 021204      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4594 024522 021204
4595 024524 016104 000004   MOV      4(R1),R4  ;PUT 'FOUND' IN R4
4596 024530 112205          MOV      (R2)+,R5  ;PUT 'EXPECTED' IN R5
4597 024532 120504          CMP      R5,R4  ;IS RECEIVED DATA CORRECT?
4598 024534 001401          BEQ      41$      ;BR IF YES
4599 024536 104010          HLT      10        ;RECEIVE DATA ERROR
4600 024540 005303          41$: DEC      R3        ;DEC CHARACTER COUNT
4601 024542 001364          BNE      40$      ;BR IF NOT DONE YET
4602
4603          ;CHECK TO SEE THAT IN BCC MATCH IS SET
4604          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
4605
4606 024544 004737 027202   JSR      PC,INRDY  ;WAIT FOR INRDY
4607 024550 104414          ROMCLK 021204      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4608 024552 021204          ;GET FIRST HALF OF CRC

```

4609	024554	116137	000004	001252	MOV	4(R1),TEMP3	;PUT IN TEMP3
4610	024562	042737	177400	001252	BIC	#177400,TEMP3	;CLEAR HI BYTE
4611	024570	004737	027202		JSR	PC,INRDY	;WAIT FOR INRDY
4612	024574	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4613	024576	021244			021244		
4614	024600	016104	000004		MOV	4(R1),R4	;PUT 'FOUND' IN R4
4615	024604	042704	000376		BIC	#376,R4	;CLEAR UNWANTED BITS
4616	024610	012705	000001		MOV	#1,R5	;PUT 'EXPECTED' IN R5
4617	024614	120504			CMPB	R5,R4	;IS IN BCC MATCH SET?
4618	024616	001401			BEQ	50\$	
4619	024620	104015			HLT	15	;IN BCC MATCH ERROR
4620	024622						
4621	024622	104414			50\$: ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4622	024624	021204			021204		;GET LAST HALF
4623	024626	116137	000004	001251	MOV	4(R1),TEMP2+1	;PUT IN TEMP2
4624	024634	042737	000377	001250	BIC	#377,TEMP2	;CLEAR LO BYTE
4625	024642	053737	001250	001252	BIS	TEMP2,TEMP3	;16 BIT BCC NOW IN TEMP3
4626	024650	023737	027344	001252	CMP	CALBCC,TEMP3	;IS IT CORRECT?
4627	024656	001401			BEQ	45\$	;BR IF OK
4628	024660	104027			HLT	27	
4629							
4630							;CHECK THAT CHARACTER LOADED WITH SOM WAS RECEIVED (000)
4631							
4632	024662	004737	027202		45\$: JSR	PC,INRDY	;WAIT FOR INRDY
4633	024666	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4634	024670	021204			021204		;GET RECEIVE DATA
4635	024672	016104	000004		MOV	4(R1),R4	;PUT 'FOUND' IN R4
4636	024676	005005			CLR	R5	;PUT 'EXPECTED' IN R5
4637	024700	120504			CMPB	R5,R4	;IS RECEIVED DATA CORRECT?
4638	024702	001401			BEQ	42\$	;BR IF YES
4639	024704	104010			HLT	10	;RECEIVE DATA ERROR
4640							
4641							;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
4642							;WAS RECEIVED CORRECTLY (0,125,252,377)
4643							
4644	024706	012703	000004		42\$: MOV	#4,R3	;R3=CHARACTER COUNT
4645	024712	012702	030126		MOV	#MESDAT,R2	;LOAD MESSAGE POINTER IN R2
4646	024716	004737	027202		43\$: JSR	PC,INRDY	;WAIT FOR INRDY
4647	024722	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4648	024724	021204			021204		
4649	024726	016104	000004		MOV	4(R1),R4	;PUT 'FOUND' IN R4
4650	024732	112205			MOV	(R2)+,R5	;PUT 'EXPECTED' IN R5
4651	024734	120504			CMPB	R5,R4	;IS RECEIVED DATA CORRECT?
4652	024736	001401			BEQ	44\$	;BR IF YES
4653	024740	104010			HLT	10	;RECEIVE DATA ERROR
4654	024742	005303			44\$: DEC	R3	;DEC CHARACTER COUNT
4655	024744	001364			BNE	43\$	;BR IF NOT DONE YET
4656							
4657							;CHECK TO SEE THAT IN BCC MATCH IS SET
4658							;AND THAT THE BCC WAS RECEIVED CORRECTLY
4659							
4660	024746	004737	027202		JSR	PC,INRDY	;WAIT FOR INRDY
4661	024752	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4662	024754	021204			021204		;GET FIRST HALF OF CRC
4663	024756	116137	000004	001252	MOV	4(R1),TEMP3	;PUT IN TEMP3
4664	024764	042737	177400	001252	BIC	#177400,TEMP3	;CLEAR HI BYTE

```

4665 024772 004737 027202      JSR      PC,INRDY      ;WAIT FOR INRDY
4666 024776 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4667 025000 021244      021244
4668 025002 016104 000004      MOV      4(R1),R4      ;PUT 'FOUND' IN R4
4669 025006 042704 000376      BIC      #376,R4      ;CLEAR UNWANTED BITS
4670 025012 012705 000001      MOV      #1,R5      ;PUT 'EXPECTED' IN R5
4671 025016 120504      CMPB      R5,R4      ;IS IN BCC MATCH SET?
4672 025020 001401      BEQ      51$
4673 025022 104015      HLT      15      ;IN BCC MATCH ERROR
4674 025024      51$:
4675 025024 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4676 025026 021204      021204      ;GET LAST HALF
4677 025030 116137 000004 001251      MOVB      4(R1),TEMP2+1      ;PUT IN TEMP2
4678 025036 042737 000377 001250      BIC      #377,TEMP2      ;CLEAR LO BYTE
4679 025044 053737 001250 001252      BIS      TEMP2,TEMP3      ;16 BIT BCC NOW IN TEMP3
4680 025052 023737 027344 001252      CMP      CALBCC,TEMP3      ;IS IT CORRECT?
4681 025060 001401      BEQ      5$      ;BR IF OK
4682 025062 104027      HLT      27
4683 025064 104400      5$:      SCOPE      ;SCOPE THIS TEST
4684
4685
4686      ;***** TEST 54 *****
4687      ;*EMPTY SILO TEST
4688      ;*LOAD SILO WITH 2 SYNCs, 4 CHAR MESSAGE, SINGLE CLOCK
4689      ;*UNTIL THE SILO IS EMPTY, LOAD 4 MORE CHARACTERS IN THE
4690      ;*SILO. GIVE MORE TICKS, AND VERIFY THAT ONLY THE FIRST
4691      ;*4 CHARACTER MESSAGE WAS RECEIVED AND THAT RTS IS CLEAR
4692      ;*****
4693
4694      ; TEST 54
4695      ;-----
4696 025066 012737 000054 001226 TST54: MOV      #54,TSTNO
4697 025074 012737 025320 001216      MOV      #TST55,NEXT
4698
4699 025102 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
4700 025104 012711 004000      MOV      #BIT11,(R1)      ;MASTER CLEAR DMC11
4701 025110 012702 030126      MOV      #MESDAT,R2      ;SET LINE UNIT LOOP
4702 025114 012700 000004      MOV      #4,R0      ;R2 POINTS TO MESSAGE
4703 025120 004737 027510      JSR      PC,SYNLD      ;R0 = CHAR COUNT
4704 025124 004737 026526      JSR      PC,OUTPDY      ;LOAD SILO WITH TWO SYNCs
4705 025130 004537 027644      JSR      R5,MESLD      ;WAIT FOR OUTRDY
4706 025134 030126      MESDAT      ;LOAD MESSAGE IN SILO
4707 025136 000004      4      ;START OF MESSAGE
4708 025140 004737 026374      JSR      PC,OCOR      ;CHARACTER COUNT
4709 025144 104415 000065      DATACLK, 65      ;WAIT FOR OCOR
4710 025150 004537 027644      JSR      R5,MESLD      ;CLOCK DATA (EMPTY SILO)
4711 025154 030126      MESDAT      ;PUT MORE CHARACTERS IN SILO
4712 025156 000004      4
4713 025160 004737 026374      JSR      PC,OCOR
4714 025164 104415 000005      DATACLK, 5      ;CLOCK UNTIL RTS IS CLEARED
4715 025170 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4716 025172 021264      021264      ;GET RTS
4717 025174 032761 000040 000004      BIT      #BIT5,4(R1)      ;IS IT CLEAR?
4718 025202 001401      BEQ      5$      ;BR IF YES
4719 025204 104034      HLT      34      ;ERROR, RTS NOT CLEAR
4720 025206 104415 000041      5$:      DATACLK, 41      ;CLOCK XMITTER SOME MORE

```

4721	025212	004737	027202	1\$:	JSR	PC,INRDY	:OK LETS CHECK WHAT WAS RECEIVED
4722	025216	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4723	025220	021204			021204		:GET RECEIVE DATA
4724	025222	016104	000004		MOV	4(R1),R4	:PUT IT IN R4
4725	025226	112205			MOVB	(R2)+,R5	:R5 = 'EXPECTED'
4726	025230	120504			CMPB	R5,R4	:IS DATA CORRECT?
4727	025232	001401			BEQ	2\$	:BR IF OK
4728	025234	104010			HLT	10	:DATA ERROR
4729	025236	005300		2\$:	DEC	R0	:DEC CHAR COUNT
4730	025240	001364			BNE	1\$	:BR IF NOT DONE YET
4731	025242	004737	027202	3\$:	JSR	PC,INRDY	:WAIT FOR INRDY
4732	025246	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4733	025250	021204			021204		:GET RECEIVE DATA
4734	025252	016104	000004		MOV	4(R1),R4	:PUT IT IN 'FOUND'
4735	025256	012705	000377		MOV	#377,R5	:R5 = 'EXPECTED'
4736	025262	120504			CMPB	R5,R4	:SHOULD SEE 377
4737	025264	001401			BEQ	4\$	:BR IF OK
4738	025266	104010			HLT	10	:ERROR, TRANSMITTER DID NOT ABORT
4739	025270	004737	027202	4\$:	JSR	PC,INRDY	:WAIT FOR INRDY
4740	025274	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304
4741	025276	021204			021204		:GET RECEIVE DATA
4742	025300	016104	000004		MOV	4(R1),R4	:PUT IT IN 'FOUND'
4743	025304	012705	000377		MOV	#377,R5	:R5 = 'EXPECTED'
4744	025310	120504			CMPB	R5,R4	:SHOULD SEE 377
4745	025312	001401			BEQ	10\$	:BR IF OK
4746	025314	104010			HLT	10	:ERROR, TRANSMITTER DID NOT ABORT
4747	025316			10\$:			
4748	025316	104400			SCOPE		:SCOPE THIS TEST

4750
 4751
 4752
 4753
 4754
 4755
 4756
 4757
 4758
 4759
 4760
 4761
 4762
 4763
 4764
 4765
 4766
 4767
 4768
 4769
 4770
 4771
 4772
 4773
 4774
 4775
 4776

:***** TEST 55 *****
 :*HALF DUPLEX TEST
 :*SET LINE UNIT LOOP AND HALF DUPLEX, SEND SYNCs AND A
 :*MESSAGE. VERIFY THAT IN-ACTIVE AND IN-READY ARE CLEAR
 :*****

: TEST 55

4759	025320	012737	000055	001226	TST55:	MOV	#55,TSTNO	
4760	025326	012737	025436	001216		MOV	#TST56,NEXT	
4761								
4762	025334	104412				MSTCLR		:R1 CONTAINS BASE DMC11 ADDRESS
4763	025336	012702	000012			MOV	#12,R2	:MASTER CLEAR DMC11
4764	025342	012711	004000			MOV	#BIT11,(R1)	:SAVE R2 FOR TYPEOUT
4765	025346	012761	000020	000004		MOV	#BIT4,(R1)	:SET LINE UNIT LOOP
4766	025354	104414				ROMCLK		:LOAD PORT4
4767	025356	122113				122113		:NEXT WORD IS INSTRUCTION, ROMCLK PC-5304
4768	025360	004737	027510			JSR	PC,SYNLD	:SET H/D BIT
4769	025364	004737	026526			JSR	PC,OUTRDY	:LOAD 2 SYNCs
4770	025370	004537	027644			JSR	R5,MESLD	:WAIT FOR OUTRDY
4771	025374	030126				MESDAT		:LOAD 4 CHAR MESSAGE
4772	025376	000004				4		:ADDRESS OF MESSAGE
4773	025400	004737	026374			JSR	PC,OCOR	:CHARACTER COUNT
4774	025404	104415	000073			DATACLK,	73	:WAIT FOR OCOR
4775	025410	104414				ROMCLK		:SEND MESSAGE
4776	025412	021244				021244		:NEXT WORD IS INSTRUCTION, ROMCLK PC 5304

:READ LU-12

```

4777 025414 016104 000004      MOV      4(R1),R4      ;PUT 'FOUND' IN R4
4778 025420 042704 000257      BIC      #257,R4      ;CLEAR UNWANTED BITS
4779 025424 005005              CLR      R5          ;R5 = 'EXPECTED'
4780 025426 120504              CMPB     R5,R4          ;IN-ACTIVE AND IN-RDY SHOULD BE CLEAR
4781 025430 001401              BEQ      1$          ;BR IF OK
4782 025432 104035              HLT      3$          ;ERROR BOTH ARE NOT CLEAR
4783 025434 104400              1$:      SCOPE
4784
4785
4786
4787
4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798 025436 012737 000056 001226 TST56: MOV      #56,TSTNO
4799 025444 012737 026024 001216      MOV      #TST57,NEXT
4800
4801 025452 104412              MSTCLR
4802 025454 032737 040000 001366      BIT      #BIT14,STAT1 ;R1 CONTAINS BASE DMC11 ADDRESS
4803 025462 001557              BEQ      3$          ;MASTER CLEAR DMC11
4804 025464 012711 004000              MOV      #BIT11,(R1) ;SKIP TEST IF NO
4805 025470 004737 027510              JSR      PC,SYNLD ;LOOPBACK CONNECTOR ON
4806 025474 004737 027510              JSR      PC,SYNLD ;SET LINE UNIT LOOP
4807 025500 012737 120001 027342      MOV      #CRC16,XPOLY ;LOAD 2 SYNCs
4808 025506 005037 025536              CLR      6$          ;LOAD 2 MORE SYNCs
4809 025512 012703 000020              MOV      #16,R3      ;LOAD POLYNOMIAL FOR SOFT CRC CALC
4810 025516 012702 030132              MOV      #FLTDAT,R2 ;CLEAR OLD BCC
4811 025522 112237 025534              MOV      (R2)+,5$    ;CHARACTER COUNT
4812 025526 004537 027236              JSR      R5,SIMBCC ;R2= POINTER
4813 025532 000010              10
4814 025534 000000              5$:      0
4815 025536 000000              6$:      0
4816 025540 013737 027344 025536      MOV      CALBCC,6$   ;LOAD OLD BCC
4817 025546 005303              DEC      R3          ;DEC COUNT
4818 025550 001364              BNE     7$          ;BR IF NOT DONE YET
4819 025552 004537 027644              JSR      R5,MESLD ;LOAD SILO
4820 025556 030132              FLTDAT ;MESSAGE ADDRESS
4821 025560 000020              16. ;CHARACTER COUNT
4822 025562 004737 027620              JSR      PC,EOM ;LOAD AN EOM
4823 025566 004537 027644              JSR      R5,MESLD ;LOAD SILO
4824 025572 030132              FLTDAT ;MESSAGE ADDRESS
4825 025574 000020              16. ;CHARACTER COUNT
4826 025576 004737 027620              JSR      PC,EOM ;LOAD AN EOM
4827 025602 004537 027644              JSR      R5,MESLD ;LOAD SILO
4828 025606 030132              FLTDAT ;MESSAGE ADDRESS
4829 025610 000020              16. ;CHARACTER COUNT
4830 025612 004737 027620              JSR      PC,EOM ;LOAD AN EOM
4831 025616 004737 026374              JSR      PC,OCOR ;WAIT FOR OCOR
4832 025622 005011              CLR      (R1) ;CLEAR LINE UNIT LOOP

```

```

4833 025624 012700 000003      MOV    #3,R0      ;R0 = MESSAGE COUNT
4834 025630 012703 000020      MOV    #16,R3     ;R3= CHARACTER COUNT
4835 025634 012702 030132      MOV    #FLTDAT,R2  ;LOAD MESSAGE POINTER IN R2
4836 025640 004737 027202      JSR    PC,INRDY    ;WAIT FOR INRDY
4837 025644 104414      ROMCLK 021204    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4838 025646 021204      MOV    4(R1),R4   ;GET DATA FROM IN SILO
4839 025650 016104 000004      MOV    (R2)+,R5   ;PUT CHARACTER IN 'FOUND'
4840 025654 112205      MOV    R5,R4      ;PUT 'EXPECTED' IN R5
4841 025656 120504      CMP    R5,R4      ;IS RECEIVED DATA CORRECT
4842 025660 001401      BEQ    2$        ;BR IF OK
4843 025662 104025      HLT    25        ;DATA ERROR
4844 025664      2$:
4845 025664 005303      DEC    R3          ;DEC CHARACTER COUNT
4846 025666 001364      BNE    1$        ;BR IF NOT DONE THIS MESSAGE
4847 025670 012703 000020      MOV    #16,R3     ;RESET CHARACTER COUNT
4848
4849      ;CHECK TO SEE THAT IN BCC MATCH IS SET
4850      ;AND THAT THE BCC WAS RECEIVED CORRECTLY
4851
4852 025674 004737 027202      JSR    PC,INRDY    ;WAIT FOR INRDY
4853 025700 104414      ROMCLK 021204    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4854 025702 021204      MOV    4(R1),TEMP3   ;GET FIRST HALF OF CRC
4855 025704 116137 000004 001252  MOV    #177400,TEMP3 ;PUT IN TEMP3
4856 025712 042737 177400 001252  BIC    PC,INRDY    ;CLEAR HI BYTE
4857 025720 004737 027202      JSR    PC,INRDY    ;WAIT FOR INRDY
4858 025724 104414      ROMCLK 021244    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4859 025726 021244      MOV    4(R1),R4   ;PUT 'FOUND' IN R4
4860 025730 016104 000004      BIC    #376,R4      ;CLEAR UNWANTED BITS
4861 025734 042704 000376      MOV    #1,R5        ;PUT 'EXPECTED' IN R5
4862 025740 012705 000001      CMP    R5,R4      ;IS IN BCC MATCH SET?
4863 025744 120504      BEQ    25$       ;IN BCC MATCH ERROR
4864 025746 001401      HLT    15
4865 025750 104015      25$:
4866 025752      ROMCLK 021204    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4867 025752 104414      MOV    4(R1),TEMP2+1 ;GET LAST HALF
4868 025754 021204      BIC    #377,TEMP2   ;PUT IN TEMP2
4869 025756 116137 000004 001251  BIS    TEMP2,TEMP3 ;CLEAR LO BYTE
4870 025764 042737 000377 001250  CMP    CALBCC,TEMP3 ;16 BIT BCC NOW IN TEMP3
4871 025772 053737 001250 001252  BEQ    4$        ;IS IT CORRECT?
4872 026000 023737 027344 001252  HLT    27        ;BR IF OK
4873 026006 001401      4$:
4874 026010 104027      MOV    #FLTDAT,R2   ;RESET MESSAGE POINTER
4875 026012 012702 030132      DEC    R0          ;DECREMENT COUNTER
4876 026016 005300      BNE    1$        ;BR IF NOT DONE
4877 026020 001307      3$:
4878 026022 104400      SCOPE
4879
4880
4881      ;***** TEST 57 *****
4882      ;*DDCMP CABLE DATA TEST
4883      ;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
4884      ;*4 SYNCs, 59 DATA CHARACTERS, EOM WITH GARAGE CHARACTER
4885      ;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
4886      ;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
4887      ;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
4888      ;*****

```

```

4889
4890
4891      : TEST 57
4892      :-----
4892 026024 012737 000057 001226 TST57: MOV #57,TSTNO
4893 026032 012737 003364 001216 MOV #.EOP,NEXT
4894
4895 026040 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
4896 026042 032737 040000 001366 BIT #BIT14,STAT1 ;MASTER CLEAR DMC11
4897 026050 001533 BEQ 3$ ;SKIP TEST IF NO
4898 026052 012711 004000 MOV #BIT11,(R1) ;LOOPBACK CONNECTOR ON
4899 026056 004737 027510 JSR PC,SYNLD ;SET LINE UNIT LOOP
4900 026062 004737 027510 JSR PC,SYNLD ;LOAD 2 SYNCs
4901 026066 012737 120001 027342 MOV #CRC16,XPOLY ;LOAD 2 MORE SYNCs
4902 026074 005037 026124 CLR 6$ ;LOAD POLYNOMIAL FOR SOFT CRC CALC
4903 026100 012703 000073 MOV #59.,R3 ;CLEAR OLD BCC
4904 026104 012702 030126 MOV #MESDAT,R2 ;CHARACTER COUNT
4905 026110 112237 026122 1$: MOV (R2)+,5$ ;R2= POINTER
4906 026114 004537 027236 JSR R5,SIMBCC ;LOAD CHAR FOR SOFT BCC CALC.
4907 026120 000010 10 ;CALC SOFT BCC
4908 026122 000000 5$: 0 ;SHIFT COUNT
4909 026124 000000 6$: 0 ;CHARACTER
4910 026126 013737 027344 026124 MOV CALBCC,6$ ;OLD BCC
4911 026134 005303 DEC R3 ;LOAD OLD BCC
4912 026136 001364 BNE 7$ ;DEC COUNT
4913 026140 004537 027644 JSR R5,MESLD ;BR IF NOT DONE YET
4914 026144 030126 MESDAT ;LOAD SILO
4915 026146 000073 59. ;MESSAGE ADDRESS
4916 026150 004737 027620 JSR PC,EOM ;CHARACTER COUNT
4917 026154 004737 026374 JSR PC,OCOR ;LOAD AN EOM
4918 026160 005011 (R1) ;WAIT FOR OCOR
4919 026162 012700 000073 CLR (R1) ;CLEAR LINE UNIT LOOP
4920 026166 012702 030126 MOV #59.,R0 ;R0= CHARACTER COUNT
4921 026172 004737 027202 1$: MOV #MESDAT,R2 ;LOAD MESSAGE POINTER IN R2
4922 026176 104414 ROMCLK PC,INRDY ;WAIT FOR INRDY
4923 026200 021204 021204 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4924 026202 016104 000004 MOV 4(R1),R4 ;GET DATA FROM IN SILO
4925 026206 112205 MOV (R2)+,R5 ;PUT CHARACTER IN 'FOUND'
4926 026210 120504 CMPB R5,R4 ;PUT 'EXPECTED' IN R5
4927 026212 001401 BEQ 2$ ;IS RECEIVED DATA CORRECT
4928 026214 104025 HLT 25 ;BR IF OK
4929 026216 2$: ;DATA ERROR
4930 026216 005300 DEC R0 ;DECREMENT COUNTER
4931 026220 001364 BNE 1$ ;BR IF NOT DONE
4932
4933 ;CHECK TO SEE THAT IN BCC MATCH IS SET
4934 ;AND THAT THE BCC WAS RECEIVED CORRECTLY
4935
4936 026222 004737 027202 JSR PC,INRDY ;WAIT FOR INRDY
4937 026226 104414 ROMCLK 021204 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4938 026230 021204 021204 ;GET FIRST HALF OF CRC
4939 026232 116137 000004 001252 MOVB 4(R1),TEMP3 ;PUT IN TEMP3
4940 026240 042737 177400 001252 BIC #177400,TEMP3 ;CLEAR HI BYTE
4941 026246 004737 027202 JSR PC,INRDY ;WAIT FOR INRDY
4942 026252 104414 ROMCLK 021244 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4943 026254 021244 000004 MOV 4(R1),R4 ;PUT 'FOUND' IN R4
4944 026256 016104

```

CZDME MACY11 30A(1052) 11-SEP-78 12:45 PAGE 94
 CZDMEC.P11 11-SEP-78 12:38 BASIC RECEIVER TESTS

SEQ 0094

```

4945 026262 042704 000376      BIC      #376,R4      ;CLEAR UNWANTED BITS
4946 026266 012705 000001      MOV      #1,R5      ;PUT 'EXPECTED' IN R5
4947 026272 120504              CMPB     R5,R4      ;IS IN BCC MATCH SET?
4948 026274 001401              BEQ      25$
4949 026276 104015              HLT      15      ;IN BCC MATCH ERROR
4950 026300
4951 026300 104414      25$: ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4952 026302 021204      021204      ;GET LAST HALF
4953 026304 116137 000004 001251      MOVB     4(R1),TEMP2+1 ;PUT IN TEMP2
4954 026312 042737 000377 001250      BIC      #377,TEMP2 ;CLEAR LC BYTE
4955 026320 053737 001250 001252      BIS      TEMP2,TEMP3 ;16 BIT BCC NOW IN TEMP3
4956 026326 023737 027344 001252      CMP      CALBCC,TEMP3 ;IS IT CORRECT?
4957 026334 001401              BEQ      3$      ;BR IF OK
4958 026336 104027              HLT      27
4959 026340 104400      3$: SCOPE      ;SCOPE THIS TEST
4960
4961
4962      ;SUBROUTINES
4963      ;-----
4964
4965 026342      GETSI:
4966      ;THIS SUBROUTINE READS LU 17, AND PUTS IT INTO NITCH.
4967      ;NITCH IS ROTATED LEFT UNTILL THE SI BIT IS IN CARRY
4968
4969 026342 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4970 026344 021364      021364      ;PORT4 LU 17
4971 026346 017737 153040 026372      MOV      @DMP04,NITCH ;STORE LU 17
4972 026354 106137 026372      ROLB     NITCH
4973 026360 106137 026372      ROLB     NITCH
4974 026364 106137 026372      ROLB     NITCH      ;PUT SI IN THE CARRY BIT
4975 026370 000207      RTS      PC
4976 026372 000000      NITCH: 0
4977
4978
4979 026374      OCOR:
4980      ;THIS SUBROUTINE SPINS ON OCOR
4981
4982 026374 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4983 026376 021364      021364      ;PORT4 LU 17
4984 026400 032777 000020 153004      BIT      #BIT4,@DMP04 ;IS OCOR SET?
4985 026406 001772      BEQ      OCOR      ;BR IF NO
4986 026410 000207      RTS      PC      ;OK OCOR IS SET, GO BACK
4987
4988
4989 026412      SYNC:
4990      ;THIS SUBROUTINE LOADS THE SILO WITH THE NUMBER OF SYNC
4991      ;CHARACTERS PASSED TO I1 IN THE WORD AFTER THE JSR CALL
4992      ;AND A NON-SYNC CHARACTER (301)
4993
4994 026412 013637 001246      MOV      @ (SP)+,TEMP1 ;GET COUNT
4995 026416 062746 000002      ADD      #2,-(SP) ;ADJUST STACK
4996 026422 012761 000026 000004      MOV      #26,4(R1) ;LOAD PORT4
4997 026430 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4998 026432 122114      122114      ;LOAD SYNC REGISTER
4999 026434 004737 026526      JSR      PC,OUTRDY ;WAIT FOR OUTRDY
5000 026440 012761 000001 000004      MOV      #1,4(R1) ;LOAD PORT4

```

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

```

5057 026664 004737 026374      JSR    PC,OCOR      ;WAIT FOR OCOR
5058 026670 000207              RTS    PC
5059
5060
5061 026672      CHARS:      ;THIS SUBROUTINE LOADS THE SILO WITH THE CHARACTER PASSED TO IT.
5062
5063
5064 026672 013637 001250      MOV    @ (SP)+,TEMP2      ;GET CHARACTER
5065 026676 062746 000002      ADD    #2,-(SP)      ;ADJUST STACK
5066 026702 004737 026526      JSR    PC,OUTRDY      ;WAIT FOR OUTRDY
5067 026706 013761 001250 000004      MOV    TEMP2,4(R1)      ;LOAD PORT4
5068 026714 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5069 026716 122110              122110      ;LOAD OUT DATA
5070 026720 004737 026526      JSR    PC,OUTRDY      ;WAIT FOR OUTRDY
5071 026724 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5072 026726 122110              122110      ;LOAD GARBAGE CHAR
5073 026730 004737 026374      JSR    PC,OCOR      ;WAIT FOR OCOR
5074 026734 000207              RTS    PC
5075
5076
5077 026736      SILOLD:      ;THIS SUBROUTINE FILLS THE OUT SILO
5078                          ; WITH A BINARY COUNT PATTERN
5079
5080
5081 026736 012737 000073 001250      MOV    #73,TEMP2      ;LOAD COUNT
5082 026744 005737 027176              TST    SCHAR      ;FIRST TIME HERE?
5083 026750 100470              BMI     4$      ;BR IF BITSTUFF
5084 026752 001032              BNE     2$      ;BR IF NO
5085 026754 062737 000002 001250      ADD    #2,TEMP2      ;ADD 2 TO CHARACTER COUNT
5086 026762 012737 000003 001246      MOV    #3,TEMP1      ;SET FOR 3 SYNCs
5087 026770 012761 000026 000004      MOV    #26,4(R1)      ;LOAD PORT4
5088 026776 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5089 027000 122114              122114      ;LOAD SYNC REGISTER
5090 027002 004737 026526 1$:      JSR    PC,OUTRDY      ;WAIT FOR OUTRDY
5091 027006 012761 000001 000004      MOV    #1,4(R1)      ;LOAD PORT4
5092 027014 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5093 027016 122111              122111      ;SET SOM
5094 027020 012761 000026 000004      MOV    #26,4(R1)      ;LOAD PORT4
5095 027026 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5096 027030 122110              122110      ;LOAD OUT DATA
5097 027032 005337 001246              DEC    TEMP1      ;ALL DONE?
5098 027036 001361              BNE     1$      ;BR IF NOT
5099 027040 004737 026526 2$:      JSR    PC,OUTRDY      ;WAIT FOR OUTRDY
5100 027044 013761 027176 000004      MOV    SCHAR,4(R1)      ;LOAD PORT4
5101 027052 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5102 027054 122110              122110      ;LOAD OUT DATA
5103 027056 005737 027200              TST    STUFLG      ;BITSTUFF???
5104 027062 001407              BEQ     6$      ;BR IF NO
5105 027064 013737 027176 027076      MOV    SCHAR,5$      ;IT IS SLD SO CHECK BITSTUFFING
5106 027072 004537 027726              JSR    R5,STFFCL      ;ADD ANY BIT STUFF CLOCK TICKS
5107 027076 000000 5$:      0      ;CHARACTER
5108 027100 000010              10      ;SHIFT COUNT
5109 027102 005237 027176 6$:      INC    SCHAR      ;NEXT CHARACTER
5110 027106 022737 000400 027176      CMP    #400,SCHAR      ;ALL DONE?
5111 027114 001403              BEQ     3$
5112 027116 005337 001250              DEC    TEMP2      ;DECREMENT COUNT

```

```

5113 027122 001346      BNE      2$      ;BR IF NOT DONE
5114 027124 004737 026374 3$: JSR      PC,OCOR ;WAIT FOR OCOR
5115 027130 000207      RTS      PC
5116 027132 005037 027176 4$: CLR      SCHAR ;START PATTERN AT ZERO
5117 027136 012737 177777 027200 MOV     #-1,STUFLG ;SET BITSTUFF FLAG
5118 027144 005037 030124 CLR      BITCON ;CLEAR STUFF COUNT
5119 027150 062737 000002 001250 ADD     #2,TEMP2 ;ADD 2 TO CHARACTER COUNT
5120 027156 012761 000001 000004 MOV     #1,4(R1) ;SET BIT0 IN PORT4
5121 027164 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5122 027166 122111 122111 ;SET SOM!
5123 027170 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5124 027172 122110 122110 ;LOAD GARBAGE CHAR
5125 027174 000721 BR      2$      ;GO LOAD SILO
5126 027176 000000 SCHAR: 0
5127 027200 000000 STUFLG: 0
5128
5129
5130 027202 INRDY:
5131 ;THIS SUBROUTINE SPINS ON INRDY
5132 ;IF INRDY FAILS TO SET THE DELAY TIMES OUT AND AN
5133 ;ERROR IS REPORTED. FOR BETTER SCOPE LOOPS THIS
5134 ;DELAY CAN BE MADE SHORTER BY ALTERING THE NUMBER
5135 ;INITIALLY LOADED INTO TEMP1, THE SMALLER THE NUMBER
5136 ;THE SHORTER THE DELAY. 0 IS THE LONGEST DELAY.
5137
5138 027202 012737 000000 001246 1$: MOV     #0,TEMP1 ;SET UP DELAY COUNTER
5139 027210 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5140 027210 104414 021244 ;PORT4 LU12
5141 027212 032777 000020 152170 BIT      #BIT4,@DMP04 ;IS INRDY SET?
5142 027222 001004 BNE      2$      ;BR IF YES
5143 027224 005237 001246 INC      TEMP1 ;INC DELAY
5144 027230 001367 BNE      1$      ;TRY AGAIN
5145 027232 104037 HLT      37 ;ERROR,NO INRDY
5146 027234 000207 2$: RTS      PC ;RETURN
5147
5148
5149
5150 027236 SIMBCC:
5151 ;THIS SUBROUTINE CALCULATES THE CRC USING POLYNOMIAL GIVEN
5152 ;IN XPOLY. THE CORRECT CRC IS RETURNED IN CALBCC, AND THE
5153 ;STATE OF THE LSB OF THE BCC IS RETURNED IN THE C BIT.
5154
5155 027236 010046 MOV     R0,-(SP) ;SAVE R0 ON STACK
5156 027240 012537 001246 MOV     (R5)+,TEMP1 ;TEMP1 = SHIFT COUNT
5157 027244 012537 001250 MOV     (R5)+,TEMP2 ;TEMP2 = CHARACTER
5158 027250 012537 027344 MOV     (R5)+,CALBCC ;CALBCC = OLD BCC
5159 027254 013700 027344 1$: MOV     CALBCC,R0 ;PUT OLD BCC IN R0
5160 027260 000241 CLC
5161 027262 006037 027344 ROR      CALBCC ;SHIFT OLD BCC
5162 027266 006037 001250 ROR      TEMP2 ;SHIFT CHARACTER
5163 027272 005500 ADC      R0 ;ADD CHAR CARRY TO OLD BCC
5164 027274 006000 ROR      R0 ;PUT BIT0 TO CARRY BIT
5165 027276 103011 BCC      2$      ;CARRY IS FEEDBACK BIT
5166 027300 013700 027342 MOV     XPOLY,R0 ;IF FEEDBACK = 1
5167 027304 043700 027344 BIC      CALBCC,R0 ;EXCLUSIVLY OR XPOLY TO CALBCC
5168 027310 043737 027342 027344 BIC      XPOLY,CALBCC

```

```

5169 027316 050037 027344      BIS      RO,CALBCC
5170 027322 005337 001246      2$:    DEC      TEMP1      ;DEC SHIFT COUNT
5171 027326 001352              BNE      1$      ;BR IF NOT DONE
5172 027330 013700 027344      MOV      CALBCC,RO      ;PUT RESULT IN RO
5173 027334 006000              ROR      RO      ;SHIFT BITO TO CARRY
5174 027336 012600              MOV      (SP)+,RO      ;RESTORE RO
5175 027340 000205              RTS      R5      ;RETURN
5176 027342 000000      XPOLY:    0
5177 027344 000000      CALBCC:    0
5178              LRC8=200
5179              CRC16=120001
5180              CRC.CCITT-102010
5181
5182
5183 027346      BCCLD:
5184              ;THIS SUBROUTINE LOADS THE OUT SILO WITH 2 SYNCs
5185              ;WITH SOM SET, AND ONE CHARACTER PASSED TO IT
5186              ;WITH THE SOM BIT CLEAR (ENABLE CRC)
5187
5188 027346 013637 001250      MOV      @ (SP)+,TEMP2      ;GET CHARACTER
5189 027352 062746 000002      ADD      #2,-(SP)      ;ADJUST STACK
5190 027356 012737 000002 001246      MOV      #2,TEMP1      ;SET FOR 2 SYNCs
5191 027364 012761 000026 000004      MOV      #26,4(R1)      ;LOAD PORT4
5192 027372 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5193 027374 122114              122114      ;LOAD SYNC REGISTER
5194 027376 004737 026526      1$:    JSR      PC,OUTRDY      ;WAIT FOR OUTRDY
5195 027402 012761 000001 000004      MOV      #1,4(R1)      ;LOAD PORT4
5196 027410 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5197 027412 122111              122111      ;SET SOM
5198 027414 012761 000026 000004      MOV      #26,4(R1)      ;LOAD PORT4
5199 027422 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5200 027424 122110              122110      ;LOAD OUT DATA
5201 027426 005337 001246      DEC      TEMP1      ;ALL DONE?
5202 027432 001361              BNE      1$      ;BR IF NOT
5203 027434 004737 026526      JSR      PC,OUTRDY      ;WAIT FOR OUTRDY
5204 027440 013761 001250 000004      MOV      TEMP2,4(R1)      ;LOAD PORT4
5205 027446 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5206 027450 122110              122110      ;LOAD OUT DATA
5207 027452 004737 026374      JSR      PC,OCOR      ;WAIT FOR OCOR
5208 027456 000207              RTS      PC
5209
5210
5211 027460      GETQO:
5212              ;THIS SUBROUTINE READS THE STATE OF THE TRANSMIT
5213              ;BCC LSB AND PUTS IT IN THE CARRY BIT
5214
5215 027460 104414              ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5216 027462 021364              021364      ;PORT4 LU-17
5217 027464 106177 151722      ROLB      @DI:P04      ;PUT QO IN CARRY
5218 027470 000207              RTS      PC      ;RETURN
5219
5220
5221 027472      GETQI:
5222              ;THIS SUBROUTINE READS THE STATE OF THE RECEIVE
5223              ;BCC LSB AND PUTS IT IN THE CARRY BIT
5224

```

5225	027472	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5226	027474	021364			021364		:PORT4 LU-17
5227	027476	106177	151710		ROLB	@DMP04	:PUT Q0 IN CARRY
5228	027502	106177	151704		ROLB	@DMP04	:PUT Q1 IN CARRY
5229	027506	000207			RTS	PC	:RETURN
5230							
5231							
5232	027510				SYNLD:		
5233							:THIS SUBROUTINE LOADS OUT SILO WITH
5234							:2 SYNC CHARACTERS WITH SOM SET
5235							
5236	027510	012737	000002	001246	MOV	#2,TEMP1	:LOAD COUNTER FOR 2 SYNCs
5237	027516	012761	000026	000004	MOV	#26,4(R1)	:PORT4 26
5238	027524	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5239	027526	122114			122114		:LOAD SYNC REG
5240	027530	004737	026526		JSR	PC,OUTRDY	:WAIT FOR OUTRDY
5241	027534	012761	000001	000004	MOV	#1,4(R1)	:LOAD PORT4
5242	027542	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5243	027544	122111			122111		:SET SOM
5244	027546	012761	000026	000004	MOV	#26,4(R1)	:PORT 26
5245	027554	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5246	027556	122110			122110		:LOAD OUT DATA WITH SYNC
5247	027560	005337	001246		DEC	TEMP1	:DECREMENT COUNTER
5248	027564	001361			BNE	1\$	:BR IF NOT DONE
5249	027566	000207			RTS	PC	:RETURN
5250							
5251							
5252	027570				SOM:		
5253							:THIS SUBROUTINE LOADS SOM AND OUT DATA WITH A
5254							:GARBAGE CHARACTER (0)
5255							
5256	027570	004737	026526		JSR	PC,OUTRDY	:WAIT FOR OUTRDY
5257	027574	012761	000001	000004	MOV	#1,4(R1)	:PORT4 1
5258	027602	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5259	027604	122111			122111		:SET SOM
5260	027606	005061	000004		CLR	4(R1)	:CLEAR DATA CHAR
5261	027612	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5262	027614	122110			122110		:LOAD GARBAGE CHARACTER
5263	027616	000207			RTS	PC	:RETURN
5264							
5265							
5266	027620				EOM:		
5267							:THIS SUBROUTINE LOADS EOM AND OUT DATA WITH A
5268							:GARBAGE CHARACTER (2) TO ENABLE TRANSMISSION OF BCC
5269							
5270	027620	004737	026526		JSR	PC,OUTRDY	:WAIT FOR OUTRDY
5271	027624	012761	000002	000004	MOV	#2,4(R1)	:PORT4 2
5272	027632	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5273	027634	122111			122111		:SET EOM
5274	027636	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5275	027640	122110			122110		:LOAD GARBAGE CHARACTER
5276	027642	000207			RTS	PC	:RETURN
5277							
5278							
5279	027644				MESLD:		
5280							:THIS SUBROUTINE LOADS SILO WITH MESSAGE

```

5281                                     ;THE FIRST ARGUMENT IS THE ADDRESS OF THE MESSAGE
5282                                     ;THE SECOND ARGUMENT IS THE NUMBER OF CHARACTERS IN THE MESSAGE
5283
5284 027644 010046      MOV      R0,-(SP)      ;SAVE R0
5285 027646 012500      MOV      (R5)+,R0      ;R0=MESSAGE POINTER
5286 027650 012537 001246  MOV      (R5)+,TEMP1    ;TEMP1=CHARACTER COUNT
5287 027654 004737 026526 1$:      JSR      PC,OUTRDY    ;WAIT FOR OUT RDY
5288 027660 112061 000004      MOV      (R0)+,4(R1)    ;LOAD PORT4 WITH CHARACTER
5289 027664 104414      ROMCLK    122110    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5290 027666 122110      ;LOAD OUT DATA SILO
5291 027670 005337 001246      DEC      TEMP1      ;DEC CHAR COUNT
5292 027674 001367      BNE      1$              ;BR IF NOT DONE
5293 027676 004737 026374      JSR      PC,OCOR      ;WAIT FOR OCOR
5294 027702 012600      MOV      (SP)+,R0      ;RESTORE R0
5295 027704 000205      RTS      R5              ;RETURN
5296
5297
5298 027706      CLRIO:      ;THIS SUBROUTINE SETS IN CLR AND OUT CLR TO
5299                                     ;CLEAR THE TRANSMIT AND RECEIVE BCC REGISTERS
5300
5301
5302 027706 012761 000200 000004  MOV      #BIT7,4(R1)    ;LOAD PORT4
5303 027714 104414      ROMCLK    122112    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5304 027716 122112      ;SET IN CLR!
5305 027720 104414      ROMCLK    122111    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5306 027722 122111      ;SET OUT CLR!
5307 027724 000207      RTS      PC              ;RETURN
5308
5309
5310 027726      STFFCL:      ;THIS SUBROUTINE ADDS ANY NECESSARY BIT STUFF CLOCK TICKS
5311                                     ;FIRST ARGUMENT IS CHAR, SECOND ARGUMENT IS SHIFT COUNT.
5312
5313
5314 027726 010046      MOV      R0,-(SP)      ;SAVE R0
5315 027730 012500      MOV      (R5)+,R0      ;PUT CHAR IN R0
5316 027732 012537 001252  MOV      (R5)+,TEMP3    ;PUT SHIFT COUNT IN TEMP3
5317 027736 106000      1$:      RORB      R0      ;LOOK AT NEXT BIT
5318 027740 103403      BCS      2$              ;BR IF A MARK
5319 027742 005037 030124      CLR      BITCON    ;IT WAS A SPACE, CLEAR 1'S COUNTER
5320 027746 000412      BR      3$              ;CONTINUE
5321 027750 005237 030124 2$:      INC      BITCON    ;INC CONSECUTIVE 1'S COUNTER
5322 027754 022737 000005 030124  CMP      #5,BITCON    ;IS IT 5 YET?
5323 027762 001004      BNE      3$              ;BR IF NO
5324 027764 005037 030124      CLR      BITCON    ;YES! SO START AGAIN
5325 027770 104415 000001      DATACLK, 1    ;GIVE EXTRA TICK TO STUFF ZERO
5326 027774 005337 001252 3$:      DEC      TEMP3      ;DEC SHIFT COUNT
5327 030000 001356      BNE      1$              ;BR IF NOT DONE
5328 030002 012600      MOV      (SP)+,R0      ;RESTORE R0
5329 030004 000205      RTS      R5              ;RETURN
5330
5331
5332 030006      STFFCK:      ;THIS SUBROUTINE CHECKS TO SEE IF TRANSMITTER
5333                                     ;IS STUFFING ZEROS WHEN IT SHOULD. FIRST ARGUMENT
5334                                     ;IS THE CHARACTER, SECOND ARGUMENT IS SHIFT COUNT.
5335
5336

```

```

5337 030006 010046      MOV      R0, -(SP)      ;SAVE R0
5338 030010 012500      MOV      (R5)+, R0      ;PUT CHAR IN R0
5339 030012 012537 001252  MOV      (R5)+, TEMP3    ;PUT SHIFT COUNT IN TEMP3
5340 030016 106000      1$:      RORB      R0      ;SHIFT OUT NEXT BIT
5341 030020 103403      BCS      2$      ;BR IF IT IS A MARK
5342 030022 005037 030124  CLR      BITCON    ;IT WAS A SPACE, CLEAR 1'S COUNTER
5343 030026 000416      BR      3$      ;CONTINUE
5344 030030 005237 030124  2$:      INC      BITCON    ;INC CONSECUTIVE 1'S COUNTER
5345 030034 022737 000005 030124  CMP      #5, BITCON  ;5 IN A ROW YET?
5346 030042 001010      BNE      3$      ;BR IF NO
5347 030044 005037 030124  CLR      BITCON    ;YES, SO START OVER
5348 030050 104415 000001  DATA CLK, 1      ;EXTRA TICK TO STUFF ZERO
5349 030054 004737 026342  JSR      PC, GETSI    ;LOOK AT WINDOW
5350 030060 103001      BCC      3$      ;IS IT A ZERO, BR IF YES
5351 030062 104030      HLT      30      ;NO, ERROR ZERO WAS NOT STUFFED
5352 030064 005337 001252  3$:      DEC      TEMP3    ;DEC SHIFT COUNT
5353 030070 001352      BNE      1$      ;BR IF NOT DONE
5354 030072 012600      MOV      (SP)+, R0      ;RESTORE R0
5355 030074 000205      RTS      R5      ;RETURN
5356
5357
5358 030076      CTSDLY:      ;THIS SUBROUTINE WASTES TIME UNTIL CTS SETS,
5359                          ;BUT HOPEFULLY NOT SO LONG THAT THE SILO RUNS OUT
5360
5361
5362 030076 010046      MOV      R0, -(SP)      ;SAVE R0
5363 030100 012700 000032      MOV      #32, R0      ;LOAD R0 WITH COUNT
5364 030104 027777 151074 151072  1$:      CMP      @TKCSR, @TKCSR    ;WASTE TIME
5365 030112 005300      DEC      R0      ;DECREMENT COUNTER
5366 030114 001373      BNE      1$      ;DO IT AGAIN IF NOT = 0
5367 030116 012600      MOV      (SP)+, R0      ;RESTORE R0
5368 030120 000207      RTS      PC      ;RETURN
5369
5370
5371 030122 000176      FLAG:      ^B<01111110>      ;FLAG CHARACTER
5372 030124 000000      BITCON: 0
5373 030126 000      125      252  MESDAT: .BYTE 0, 125, 252, 377
5374 030131 377
5375 030132 001      002      004  FLTDAT: .BYTE 1, 2, 4, 10, 20, 40, 100, 200, 376, 375, 373, 367, 357, 337, 277, 177
5376 030135 010      020      040
5377 030140 100      200      376
5378 030143 375      373      367
5379 030146 357      337      277
5380 030151 177
5381 030152 100      140      160  STUFDT: .BYTE 100, 140, 160, 170, 3, 300, 174, 176, 177, 1
5382 030155 170      003      300
5383 030160 174      176      177
5384 030163 001
5385 030164 363      347      317      .BYTE 363, 347, 317, 200, 0, 377, 377, 377, 200, 37
5386 030167 200      000      377
5387 030172 377      377      200
5388 030175 037
5389
5390 030176 046377 047111 020105 .EVEN
      030234 046377 047111 020105 EM1: .ASCIZ <377>/LINE UNIT INITIALIZATION TEST/
      030277 377 044514 042516 EM2: .ASCIZ <377>*LINE UNIT REGISTER READ/ONLY TEST^
      EM3: .ASCIZ <377>*LINE UNIT REGISTER WRITE/READ TEST^
  
```

030343	377	044514	042516	EM4:	.ASCIIZ	<377>/LINE UNIT INTERNAL CLOCK FAILURE/
030405	377	051124	047101	EM5:	.ASCIIZ	<377>/TRANSMITTER DATA ERROR/
030435	377	042522	042503	EM6:	.ASCIIZ	<377>/RECEIVER TEST/
030454	051377	041505	044505	EM7:	.ASCIIZ	<377>/RECEIVER DATA ERROR/
030501	377	047515	042504	EM10:	.ASCIIZ	<377>/MODEM SIGNAL ERROR/
030525	377	051124	047101	EM11:	.ASCIIZ	<377>/TRANSMITTER CRC ERROR/
030554	051377	041505	044505	EM12:	.ASCIIZ	<377>/RECEIVER CRC ERROR/
030600	044777	020116	041502	EM13:	.ASCIIZ	<377>/IN BCC MATCH ERROR (LU REG 12)/
030640	052377	040522	051516	EM14:	.ASCIIZ	<377>/TRANSMITTER FAILED TO GO TO MARK STATE/
030710	041777	041101	042514	EM15:	.ASCIIZ	<377>/CABLE DATA TEST/
030731	377	046106	043501	EM16:	.ASCIIZ	<377>/FLAG ERROR/
030745	377	051124	047101	EM17:	.ASCIIZ	<377>/TRANSMITTER FAILED TO STUFF A ZERO/
031011	377	053523	052111	EM20:	.ASCIIZ	<377>/SWITCH PAC TEST/
031032	040777	047502	052122	EM21:	.ASCIIZ	<377>/ABORT ERROR/
031047	377	051124	047101	EM22:	.ASCIIZ	<377>/TRANSMITTER ERROR/
031072	044377	046101	020106	EM23:	.ASCIIZ	<377>/HALF DUPLEX TEST/
031114	047777	052125	051040	EM24:	.ASCIIZ	<377>/OUT READY NOT SET/
031137	377	047111	051040	EM25:	.ASCIIZ	<377>/IN READY NOT SET/
031161	377	054105	042520	DH1:	.ASCIIZ	<377>/EXPECTED FOUND/
031202	042777	050130	041505	DH2:	.ASCIIZ	<377>/EXPECTED FOUND LU-REGISTER/
031240	041777	040510	040522	DH3:	.ASCIIZ	<377>/CHARACTER BIT THAT FAILED/
031276	041777	051117	042522	DH4:	.ASCIIZ	<377>/CORRECT CRC BIT THAT FAILED/
031336	042777	050130	041505	DH5:	.ASCIIZ	<377>/EXPECTED FOUND SHIFT/
031370	042777	050130	041505	DH6:	.ASCIIZ	<377>/EXPECTED FOUND CHARACTER SHIFT/
031436	041377	047514	045503	DH7:	.ASCIIZ	<377>/BLOCK END NOT SET/
031461	377	052122	020123	DH10:	.ASCIIZ	<377>/RTS DID NOT CLEAR/
031504	000002			DT1:	2	
031506	003	007			.BYTE	3,7
031510	001272				SAVR5	
031512	003	002			.BYTE	3,2
031514	001270				SAVR4	
031516	000003			DT2:	3	
031520	003	007			.BYTE	3,7
031522	001272				SAVR5	
031524	003	010			.BYTE	3,10
031526	001270				SAVR4	
031530	003	002			.BYTE	3,2
031532	001264				SAVR2	
031534	000002			DT3:	2	
031536	003	017			.BYTE	3,17
031540	001272				SAVR5	
031542	002	002			.BYTE	2,2
031544	001266				SAVR3	
031546	000002			DT4:	2	
031550	006	021			.BYTE	6,21
031552	027344				CALBCC	
031554	002	002			.BYTE	2,2
031556	001266				SAVR3	
031560	000003			DT5:	3	
031562	001	011			.BYTE	1,11
031564	001300				ZERO	
031566	001	011			.BYTE	1,11
031570	001302				ONE	

031572	002	002		.BYTE	2,2
031574	001260			SAVR0	
031576	000003		DT6:	3	
031600	001	011		.BYTE	1,11
031602	001302			ONE	
031604	001	011		.BYTE	1,11
031606	001300			ZERO	
031610	002	002		.BYTE	2,2
031612	001260			SAVR0	
031614	000004		DT7:	4	
031616	001	011		.BYTE	1,11
031620	001300			ZERO	
031622	001	011		.BYTE	1,11
031624	001302			ONE	
031626	003	007		.BYTE	3,7
031630	001272			SAVR5	
031632	002	001		.BYTE	2,1
031634	001266			SAVR3	
031636	000004		DT10:	4	
031640	001	011		.BYTE	1,11
031642	001302			ONE	
031644	001	011		.BYTE	1,11
031646	001300			ZERO	
031650	003	007		.BYTE	3,7
031652	001272			SAVR5	
031654	002	001		.BYTE	2,1
031656	001266			SAVR3	
031660	000002		DT11:	2	
031662	003	007		.BYTE	3,7
031664	030122			FLAG	
031666	002	002		.BYTE	2,2
031670	001266			SAVR3	
031672	000002		DT12:	2	
031674	006	004		.BYTE	6,4
031676	027344			CALBCC	
031700	006	002		.BYTE	6,2
031702	001252			TEMP3	
031704			.ERRTAB:		
031704	000000			0	
031706	000000			0	
031710	000000			0	
031712	030176		EM1		
031714	031202		DH2	:HLT	1
031716	031516		DT2		
031720	030234		EM2		
031722	031202		DH2	:HLT	2
031724	031516		DT2		
031726	030277		EM3		
031730	031202		DH2	:HLT	3
031732	031516		DT2		
031734	030343		EM4		
031736	000000		0	:HLT	4
031740	000000		0		
031742	030405		EM5		
031744	031202		DH2	:HLT	5

031746	031516	DT2		
031750	030405	EM5		
031752	031240	DH3	;HLT	6
031754	031534	DT3		
031756	030435	EM6		
031760	031161	DH1	;HLT	7
031762	031504	DT1		
031764	030454	EM7		
031766	031161	DH1	;HLT	10
031770	031504	DT1		
031772	030501	EM10		
031774	031161	DH1	;HLT	11
031776	031504	DT1		
032000	030525	EM11		
032002	031336	DH5	;HLT	12
032004	031560	DT5		
032006	030554	EM12		
032010	031336	DH5	;HLT	13
032012	031560	DT5		
032014	030525	EM11		
032016	031276	DH4	;HLT	14
032020	031546	DT4		
032022	030600	EM13		
032024	031161	DH1	;HLT	15
032026	031504	DT1		
032030	030525	EM11		
032032	031336	DH5	;HLT	16
032034	031576	DT6		
032036	030554	EM12		
032040	031336	DH5	;HLT	17
032042	031576	DT6		
032044	030525	EM11		
032046	031370	DH6	;HLT	20
032050	031614	DT7		
032052	030525	EM11		
032054	031370	DH6	;HLT	21
032056	031636	DT10		
032060	030554	EM12		
032062	031370	DH6	;HLT	22
032064	031614	DT7		
032066	030554	EM12		
032070	031370	DH6	;HLT	23
032072	031636	DT10		
032074	030640	EM14		
032076	000000	0	;HLT	24
032100	000000	0		
032102	030710	EM15		
032104	031161	DH1	;HLT	25
032106	031504	DT1		
032110	030731	EM16		
032112	031240	DH3	;HLT	26
032114	031660	DT11		
032116	030554	EM12		
032120	031161	DH1	;HLT	27
032122	031672	DT12		
032124	030745	EM17		

032126	000000	0	;HLT	30
032130	000000	0		
032132	031011	EM20		
032134	031161	DH1	;HLT	31
032136	031504	DT1		
032140	031032	EM21		
032142	031436	DH7	;HLT	32
032144	000000	0		
032146	031032	EM21		
032150	031240	DH3	;HLT	33
032152	031534	DT3		
032154	031047	EM22		
032156	031461	DH10	;HLT	34
032160	000000	0		
032162	031072	EM23		
032164	031202	DH2	;HLT	35
032166	031516	DT2		
032170	031114	EM24		
032172	000000	0	;HLT	36
032174	000000	0		
032176	031137	EM25		
032200	000000	0	;HLT	37
032202	000000	0		
032204	030435	EM6		
032206	031202	DH2	;HLT	40
032210	031516	DT2		
032212	030405	EM5		
032214	031336	DH5	;HLT	41
032216	031560	DT5		
032220	030600	EM13		
032222	031161	DH1	;HLT	42
032224	031504	DT1		

032226	000001	CORMAX:		
		.END		

[illegible]

CREAM	001320	789#	1077*	1883*	1884	1886*	1890							
CSR	006510	1781#	1989											
CSRMAP	010514	1964#												
CTSDLY	030076	5358#												
CYCLE	010060	1314	1355	1356	1874#									
DATABP	005216	1643*	1646	1668	1671#									
DATACL=	104415	835#	2687	2731	2735	2795	2796	2810	2846	2847	2861	2897	2898	2912
		2948	2949	2963	3005	3008	3049	3077	3105	3133	3162	3210	3247	3284
		3321	3360	3361	3404	3405	3442	3583	3584	3612	3613	3658	3659	3687
		3688	3733	3734	3762	3763	3808	3809	3837	3838	3892	3894	3958	3960
		4022	4036	4057	4075	4082	4084	4119	4198	4212	4233	4263	4284	4302
		4309	4311	4322	4446	4460	4481	4501	4530	4551	4569	4576	4578	4589
		4709	4714	4720	4774	5325	5348							
		1642*	1664	1667#										
		831#												
DATAHD	005204	1471*	1506	1516#										
DELAY =	104413	1177	1242#											
DEVADR	004370	5390#												
DEVTAB	003010	5390#												
DH1	031161	5390#												
DH10	031461	5390#												
DH2	031202	5390#												
DH3	031240	5390#												
DH4	031276	5390#												
DH5	031336	5390#												
DH6	031370	5390#												
DH7	031436	5390#												
DISPLA	001200	736#	1092*	1098*	1329*									
DISPRE	000174	722#	1098											
DMACTV	001306	783#	1115	1270*	1271	1874	1888	1970*	2171*	2177*	2178*	2182	2207	
DMCM	007320	1228	1781#											
DMCRO0	001500	873#												
DMCRO1	001510	878#												
DMCRO2	001520	883#												
DMCRO3	001530	888#												
DMCRO4	001540	893#												
DMCRO5	001550	898#												
DMCRO6	001560	903#												
DMCRO7	001570	908#												
DMCR10	001600	913#												
DMCR11	001610	918#												
DMCR12	001620	923#												
DMCR13	001630	928#												
DMCR14	001640	933#												
DMCR15	001650	938#					</							

DT2	031516	5390#										
DT3	031534	5390#										
DT4	031546	5390#										
DT5	031560	5390#										
DT6	031576	5390#										
DT7	031614	5390#										
EM1	030176	5390#										
EM10	030501	5390#										
EM11	030525	5390#										
EM12	030554	5390#										
EM13	030600	5390#										
EM14	030640	5390#										
EM15	030710	5390#										
EM16	030731	5390#										
EM17	030745	5390#										
EM2	030234	5390#										
EM20	031011	5390#										
EM21	031032	5390#										
EM22	031047	5390#										
EM23	031072	5390#										
EM24	031114	5390#										
EM25	031137	5390#										
EM3	030277	5390#										
EM4	030343	5390#										
EM5	030405	5390#										
EM6	030435	5390#										
EM7	030454	5390#										
EOM	027620	4019	4117	4191	4195	4438	4443	4822	4826	4830	4916	5266#
ERCT00	001704	960#										
ERCT01	001710	963#										
ERCT02	001714	966#										
ERCT03	001720	969#										
ERCT04	001724	972#										
ERCT05	001730	975#										
ERCT06	001734	978#										
ERCT07	001740	981#										
ERCT10	001744	984#										
ERCT11	001750	987#										
ERCT12	001754	990#										
ERCT13	001760	993#										
ERCT14	001764	996#										
ERCT15	001770	999#										
ERCT16	001774	1002#										
ERCT17	002000	1005#										
ERR	002700	1186	1212#	1216								
ERRCNT	001232	757#	1341	1368	1681*	1899*						
ERRFLG	001325	796#	1075*	1327*	1389*	1631*	1644	1658*	1717*			
ERRMSG	005172	1641*	1659	1662#								
ERRPC	002770	1218	1234#									
ERTAB0	005322	1656	1691#									
EXIT =	000205	690#										
EXITER	005252	1676	1681#									
FLAG	030122	5371#	5390									
FLOAT	002536	1179#	1185									
FLTDAT	030132	4810	4820	4824	4828	4935	4875	5375#				
FY	002566	1188#	1199	1203	1209							

GETQI	027472	3620	3524	3695	3699	3770	3774	3845	3849	3966	3970	5221#		
GETQO	027460	3591	3595	3666	3670	3741	3745	3816	3820	3900	3904	5211#		
GETSI	026342	2799	2803	2850	2854	2901	2905	2952	2956	3011	3015	4039	4043	4060
		4064	4076	4085	4215	4219	4236	4240	4266	4270	4287	4291	4303	4312
		4463	4467	4484	4488	4504	4508	4533	4537	4554	4558	4570	4579	4965#
		5349												
HALTS	005222	1627	1673#											
HILIM	004366	1470*	1497	1515#										
ICOUNT	001222	753#	1387	1392*										
INBUF	007502	1440	1476	1796#										
INCHAR	010020	1825	1853#											
INIFLG	001324	795#	1101	1121	1128*									
INRDY	027202	3362	3406	3443	3451	3459	4120	4142	4147	4325	4339	4344	4368	4382
		4387	4592	4606	4611	4632	4646	4660	4665	4721	4731	4739	4836	4852
		4857	4921	4936	4941	5130#								
INSTER=	104404	817#	1491											
INSTR =	104403	815#	1923	1975	1988	1997	2076	2085						
INSTR2	004166	1447	1459#											
INTTY	012266	2008	2025	2035	2048	2064	2249#							
KMCM	007330	1231	1781#											
LIMITS	004314	1486	1497#											
LINE	007016	1781#	2077											
LOBITS	004372	1472*	1501	1517#	1518									
LOCK	001220	752#	1391*	1408	1410	1650	2373*	2389*	2415*	2431*	2457*	2480*	2515*	2535*
		3572*	3604*	3647*	3679*	3722*	3754*	3797*	3829*					
LOKFLG	001326	797#												
LOLIM	004364	1469*	1499	1514#										
LPCNT	001224	754#	1386*	1387	1390*									
LRC8 =	000200	5178#												
LSTERR	001234	758#	1084*	1326*	1628	1630*	1718*							
LUTYPE=	***** U	595	5150											
MASKX	001244	766#												
MASTEK	006142	1652	1781#											
MCRLF	005672	1425	1548	1648	1649	1657	1781#	1922	1984					
MCSRX	006072	1331	1781#											
MDATA	007544	1578	1588	1798#										
MEMLIM	001304	782#	1294*											
MEPASS	005733	1330	1781#											
MERRPC	006217	1655	1781#											
MERRX	006117	1337	1781#											
MERR2	005760	1781#	2184											
MERR3	006005	1267	1781#											
MESDAT	030126	3543	4011	4017	4110	4115	4183	4189	4193	4249	4324	4367	4430	4436
		4441	4516	4591	4645	4701	4706	4711	4771	4904	4914	4920	5373#	
MESLD	027644	3542	4016	4114	4188	4192	4435	4440	4705	4710	4770	4879	4823	4827
		4913	5279#											
MILK	001322	790#	1078*	1339	1882*	1887*	1891							
MLOCK	006043	1308	1781#											
MNEW	006144	1262	1781#											
MODU	006704	1781#	2047											
MPASSX	006106	1335	1781#											
MPFAIL	005675	1715	1781#											
MQM	005666	1455	1781#	1946	2021	2031	2041	2056	2070					
MR	005755	1317	1781#	1940										
MRESET	004000	690#												
MSTCLR	104412	829#	1720	2319	2343	2375	2417	2459	2517	2568	2590	2612	2646	2679

		2723	2781	2832	2883	2934	2986	3044	3072	3100	3128	3157	3205	3242
		3279	3316	3353	3394	3438	3482	3529	3574	3649	3724	3799	3872	3938
		4006	4108	4178	4425	4699	4762	4801	4895					
MTITLE	001000	730#	1105											
MTSTN	006130	1653	1781#	1924										
MTSTPC	006031	1781#												
MVECX	006100	1333	1781#											
NEXT	001216	751#	1393	1686	2270*	2294*	2317*	2341*	2372*	2414*	2456*	2514*	2566*	2588*
		2610*	2644*	2677*	2721*	2779*	2830*	2881*	2932*	2984*	3042*	3070*	3098*	3126*
		3155*	3203*	3240*	3277*	3314*	3351*	3392*	3436*	3480*	3527*	3571*	3646*	3721*
		3796*	3870*	3936*	4004*	4106*	4176*	4423*	4697*	4760*	4799*	4893*		
NITCH	026372	4971*	4972*	4973*	4974*	4976#								
NDACT	007154	1117	1781#	1876										
NODEV	002674	1171	1210#											
NUM	006450	1781#	1976											
OCOR	026374	2686	2730	2792	2843	2894	2945	3004	3891	3957	4020	4118	4196	4444
		4708	4713	4773	4831	4917	4979#	4985	5015	5057	5073	5114	5207	5293
OK	002646	1197	1204#	1233										
ONE	001302	781#	5390											
OUTRDY	026526	2783	2788	2834	2839	2885	2890	2936	2941	2991	2995	3000	3023	4015
		4113	4187	4434	4704	4769	4999	5008	5019#	5044	5053	5066	5070	5090
		5099	5194	5203	5240	5256	5270	5287						
PACT00	001702	959#												
PACT01	001706	962#												
PACT02	001712	965#												
PACT03	001716	968#												
PACT04	001722	971#												
PACT05	001726	974#												
PACT06	001732	977#												
PACT07	001736	980#												
PACT10	001742	983#												
PACT11	001746	986#												
PACT12	001752	989#												
PACT13	001756	992#												
PACT14	001762	995#												
PACT15	001766	998#												
PACT16	001772	1001#												
PACT17	001776	1004#												
PARAM =	104405	819#	1925	1977	1990	1999	2078	2087						
PARAM1	004234	1475#	1492											
PARBIT=	040000	690#												
PARERR	004310	1478	1480	1482	1491#	1498	1500	1502						
PASCNT	001230	756#	1328*	1329	1340	1365	1898*							
PERFOR=	004537	690#												
PFTAB	005430	1716	1722#											
POPRO =	012600	666#	1680											
POP1SP=	005726	664#												
POP2SP=	022626	668#	1395											
PRI0	006547	1781#	2007											
PS =	177776	657#	1070*	1305*	2211*	2221*								
PUSHRO=	010046	665#	1677											
PUSH1S=	005746	663#												
PUSH2S=	024646	667#												
QV.FLG	001327	798#	1076*	1344*	1384									
RESREG	005220	1669	1672#											
RESTAR	005350	1702	1708#											

RESTR	003534	1343	1347	1355#										
RES05 =	104407	823#	1672											
RETURN	001214	750#	1086*	1314*	1318	1355*	1393*	1397	1686*	1689	1771	1939*	1949*	1951
ROMCLK=	104414	833#	1728	1731	1768	1773	2274	2297	2321	2345	2378	2380	2391	2393
		2420	2422	2433	2435	2465	2467	2486	2488	2522	2524	2540	2542	2569
		2591	2615	2624	2649	2651	2655	2682	2684	2689	2699	2726	2728	2733
		2737	2747	2757	2785	2790	2811	2836	2841	2862	2887	2892	2913	2938
		2943	2964	2993	2997	3002	3025	3050	3078	3106	3134	3164	3174	3176
		3212	3221	3249	3258	3286	3295	3323	3332	3363	3401	3407	3444	3452
		3460	3489	3492	3501	3504	3536	3540	3548	3881	3885	3889	3919	3947
		3951	3955	3985	4121	4129	4143	4148	4157	4326	4340	4345	4354	4369
		4383	4388	4397	4593	4607	4612	4621	4633	4647	4661	4666	4675	4715
		4722	4732	4740	4766	4775	4837	4853	4858	4867	4922	4937	4942	4951
		4969	4982	4997	5001	5004	5010	5013	5024	5042	5046	5049	5055	5068
		5071	5088	5092	5095	5101	5121	5123	5140	5192	5196	5199	5205	5215
		5225	5238	5242	5245	5258	5261	5272	5274	5289	5303	5305		
RUN	001316	787#	1079*	1880*	1881*	1888								
SAVACT	001312	785#	1265	2182*										
SAVNUM	001314	786#	1073*	1342*	1345*	2175*								
SAVPC	001276	779#	1216*	1236	1523*	1693								
SAVRO	001260	772#	1532*	1537	5390									
SAVR1	001262	773#	1222*	1531*	1538									
SAVR2	001264	774#	1530*	1539	5390									
SAVR3	001266	775#	1529*	1540	5390									
SAVR4	001270	776#	1528*	1541	5390									
SAVR5	001272	777#	1527*	1542	5390									
SAVSP	001274	778#												
SAV05 =	104406	821#	1632											
SCHAR	027176	3354*	3395*	5082	5100	5105	5109*	5110	5116*	5126#				
SCOPE	104400	809#	2282	2305	2329	2360	2402	2444	2502	2554	2576	2598	2631	2664
		2708	2765	2816	2867	2918	2969	3030	3058	3086	3114	3142	3191	3228
		3265	3302	3339	3378	3422	3467	3515	3558	3633	3708	3783	3858	3924
		3990	4091	4159	4405	4683	4748	4783	4878	4959				
SCOP1 =	104401	811#	2388	2401	2430	2443	2476	2497	2531	2549	3603	3632	3678	3707
		3753	3782	3828	3857									
SILOLD	026736	3359	3375	3403	3419	5077#								
SIMBCC	027236	3586	3615	3661	3690	3736	3765	3811	3840	3895	3961	4031	4207	4258
		4455	4525	4812	4906	5150#								
SKIP	002632	1167	1170	1191	1194	1200#								
SOFTSW	010052	1823	1862#											
SOM	027570	4439	5252#											
SPACNT-	004713	1558*	1582	1585*	1599#									
SPEED	007340	1781#	2034											
STACK -	001200	658#	1071	1284	1687	1710								
STAT	001240	764#												
STAT1	001366	845#	1895*	2350	2352	3483	3485	3509	3530	3532	4802	4896		
STAT2	001370	846#	1896*	2572	2594									
STAT3	001372	847#	1897*											
STFFCK	030006	5332#												
STFFCL	027726	5106	5310#											
STRTSW	001236	763#	1107*	1110*	1111	1113	1123	1125	1260	1306	1315	1917	1941*	1971
		2195												
STUFDT	030152	5381#												
STUFLG	027200	3355*	3396*	5103	5117*	5127#								
SV05	004402	1527#												
SWFLG	010016	1074*	1419	1818*	1845*	1851#								

SWMES		007205	1781#	1821											
SWMES1		007215	1781#	1824											
SWR		001202	737#	1091*	1093	1097*	1107	1265	1270	1375	1382	1406	1421	1621	1626
			1675	1682	1684	1746	1806	1844*							
SWREG		000176	723#	1097	1806	1864									
SW00	=	000001	639#	1111	1971	2195									
SW01	=	000002	638#	1315	1917	1941									
SW02	=	000004	637#												
SW03	=	000010	636#	1260											
SW04	=	000020	635#												
SW05	=	000040	634#												
SW06	=	000100	633#	1746											
SW07	=	000200	632#												
SW08	=	000400	631#	1682											
SW09	=	001000	630#	1406											
SW10	=	002000	629#	1684											
SW11	=	004000	628#	1382											
SW12	=	010000	627#	1421	1621										
SW13	=	020000	626#	1626											
SW14	=	040000	625#												
SW15	=	100000	624#												
SYNC		026412	3047	3075	3103	3131	4989#								
SYNLD		027510	3879	3945	4014	4112	4186	4433	4703	4768	4805	4806	4899	4900	5232#
TEMP		001416	865#	1565	1712*	1713*	1754*	1759*	1765*	1777*	2613*	2619*	2622*	2628*	
TEMP1		001246	767#	1131*	1783	2207*	2208*	4994*	5006*	5040*	5051*	5086*	5097*	5138*	5144*
			5156*	5170*	5190*	5201*	5236*	5247*	5286*	5291*					
TEMP2		001250	768#	1132*	1785	4356*	4357*	4358	4399*	4400*	4401	4623*	4624*	4625	4677*
			4678*	4679	4869*	4870*	4871	4953*	4954*	4955	5038*	5054	5064*	5067	5081*
			5085*	5112*	5119*	5157*	5162*	5188*	5204						
TEMP3		001252	769#	1134*	1160*	1212	1221*	1787	1980	1983	2095*	4145*	4146*	4342*	4343*
			4358*	4359	4385*	4386*	4401*	4402	4609*	4610*	4625*	4626	4663*	4664*	4679*
			4680	4855*	4856*	4871*	4872	4939*	4940*	4955*	4956	5316*	5326*	5339*	5352*
			5390												
TEMP4		001254	770#	1135*	1789	1993	1996	2002	2005	2					

XLUC	003022
XPASS	003562
XPOLY	027342

XSTATQ 007454

XTSTN 005330

XVEC 003554

ZERO 001300

SCOD = ★★★★★ U

SCRAP = 177777

SENDAD 003522

SN = 000057

SS - 000061

SY 000017

032226

1187*	1205*	1208	1239	1252#								
1336	1363#											
3578*	3607*	3653*	3682*	3728*	3757*	3803*	3832*	3878*	3944*	4029*	4205*	4256*
4453*	4523*	4807*	4901*	5166	5168	5176#						
1138	1781#											
1654	1694#											
1334	1360#											
780#	5390											
595												
595#	2259#	2262	2265#	2283#	2286	2289#	2306#	2309	2312#	2330#	2333	2336#
2361#	2364	2367#	2403#	2406	2409#	2445#	2448	2451#	2503#	2506	2509#	2555#
2558	2561#	2577#	2580	2583#	2599#	2602	2605#	2632#	2635	2639#	2665#	2668
2672#	2709#	2712	2716#	2766#	2769	2774#	2817#	2820	2825#	2868#	2871	2876#
2919#	2922	2927#	2970#	2973	2979#	3031#	3034	3037#	3059#	3062	3065#	3087#
3090	3093#	3115#	3118	3121#	3143#	3146	3150#	3192#	3195	3198#	3229#	3232
3235#	3266#	3269	3272#	3303#	3306	3309#	3340#	3343	3346#	3379#	3382	3387#
3423#	3426	3431#	3468#	3471	3475#	3516#	3519	3522#	3559#	3562	3566#	3634#
3637	3641#	3709#	3712	3716#	3784#	3787	3791#	3859#	3862	3865#	3925#	3928
3931#	3991#	3994	3999#	4092#	4095	4101#	4160#	4163	4171#	4406#	4409	4418#
4684#	4687	4692#	4749#	4752	4755#	4784#	4787	4794#	4879#	4882	4888#	
717	1103	1349#	1673									
595#	2259	2265	2267	2272#	2283	2289	2291	2296#	2306	2312	2314	2319
2320#	2330	2336	2338	2343	2344#	2361	2367	2369	2375	2376#	2403	2409
2411	2417	2418#	2445	2451	2453	2459	2460#	2503	2509	2511	2517	2518#
2555	2561	2563	2568	2569#	2577	2583	2585	2590	2591#	2599	2605	2607
2612	2613#	2632	2639	2641	2646	2647#	2665	2672	2674	2679	2680#	2709
2716	2718	2723	2724#	2766	2774	2776	2781	2782#	2817	2825	2827	2832
2833#	2868	2876	2878	2883	2884#	2919	2927	2929	2934	2935#	2970	2979
2981	2986	2987#	3031	3037	3039	3044	3045#	3059	3065	3067	3072	3073#
3087	3093	3095	3100	3101#	3115	3121	3123	3128	3129#	3143	3150	3152
3157	3158#	3192	3198	3200	3205	3206#	3229	3235	3237	3242	3243#	3266
3272	3274	3279	3280#	3303	3309	3311	3316	3317#	3340	3346	3348	3353
3354#	3379	3387	3389	3394	3395#	3423	3431	3433	3438	3439#	3468	3475
3477	3482	3483#	3516	3522	3524	3529	3530#	3559	3566	3568	3574	3575#
3634	3641	3643	3649	3650#	3709	3716	3718	3724	3725#	3784	3791	3793
3799	3800#	3859	3865	3867	3872	3873#	3925	3931	3933	3938	3939#	3991
3999	4001	4006	4007#	4092	4101</							

[illegible]

DME ND	595#	1319													
DME RNT	595#														
HLT	669#	2281	2304	2328	2359	2387	2400	2429	2442	2475	2496	2530	2548	2575	2597
	2621	2630	2662	2696	2706	2744	2754	2763	2801	2805	2815	2852	2856	2866	2903
	2907	2917	2954	2958	2968	3013	3017	3057	3085	3113	3141	3171	3183	3190	3219
	3227	3256	3264	3293	3301	3330	3338	3369	3413	3450	3458	3466	3499	3514	3557
	3593	3597	3622	3626	3668	3672	3697	3701	3743	3747	3772	3776	3818	3822	3847
	3851	3902	3906	3968	3972	4041	4045	4062	4066	4078	4087	4127	4136	4155	4217
	4221	4238	4242	4268	4272	4289	4293	4305	4314	4332	4352	4361	4375	4395	4404
	4465	4469	4486	4490	4506	4510	4535	4539	4556	4560	4572	4581	4599	4619	4628
	4639	4653	4673	4682	4719	4728	4738	4746	4782	4843	4865	4874	4928	4949	4958
	5030	5146	5351												
SABORT	595#														
SAUTO	595#	1141													
SBCC	595#	3559	3634	3709	3784										
SBINCR	595#	3859	3925												
SBINWI	595#	2970													
SBUFE	595#	1793													
SCDATA	595#	4784	4879												
SCLOCK	595#	2599													
SCOMP	595#	3166	3178	3185	3214	3223	3251	3260	3288	3297	3325	3334	3446	3454	3462
	3494	3506	4131	4150	4347	4390	4614	4668	4860	4914					
SCRC	595#	3576	3605	3651	3680	3726	3755	3801	3830						
SCRCSH	595#	3873	3939												
SCYCLE	595#	1865													
SEMPY	595#	4684													
SEOP	595#	1319													
SFINI	595#	4960													
SFLAG	595#														
SFLOAT	595#	2461	2481	2519	2536										
SGETPA	595#														
SHALF	595#	4749													
SHEADE	595#														
SINACT	595#	3031	3059	3087	3115										
SINIT	595#														
SLINE1	595#	2445	2503												
SLU1	595#	2259	2283	2306	2330										
SLU12	595#	2361													
SLU17	595#	2403													
SMARHI	595#														
SMARK	595#	3423													
SMATCH	595#	4139	4335	4378	4602	4656	4848	4932							
SMOCK	595#														
SMODEM	595#	3468													
SMSG	595#	1781													
SMULT	595#														
SPATTE	595#	3340	3379												
SPFAIL	595#	1697													
SQOQI	595#	5211	5221												
SQUEST	595#	1975	1988	1997	2076	2085									
SRAMCL	595#	1725													
SRCLK	595#	1728	1731	1768	1773	2274	2297	2321	2345	2378	2380	2391	2393	2420	2422
	2433	2435	2465	2467	2486	2488	2522	2524	2540	2542	2569	2591	2615	2624	2649
	2651	2655	2682	2684	2689	2699	2726	2728	2733	2737	2747	2757	2785	2790	2811
	2836	2841	2862	2887	2892	2913	2938	2943	2964	2993	2997	3002	3025	3050	3078
	3106	3134	3164	3174	3176	3212	3221	3249	3258	3286	3295	3323	3332	3363	3401

	3407	3444	3452	3460	3489	3492	3501	3504	3536	3540	3547	3881	3885	3889	3919
	3947	3951	3955	3985	4121	4129	4143	4148	4157	4326	4340	4345	4354	4369	4383
	4388	4397	4493	4607	4612	4621	4633	4647	4661	4666	4675	4715	4722	4732	4740
	4766	4775	4837	4853	4858	4867	4922	4937	4942	4951	4969	4982	4997	5001	5004
	5010	5013	5024	5042	5046	5049	5055	5068	5071	5088	5092	5095	5101	5121	5123
	5140	5192	5196	5199	5205	5215	5225	5238	5242	5245	5258	5261	5272	5274	5289
	5303	5305													
\$SRCRC	595#	4092													
\$REC	595#	3143	3192	3229	3266	3303									
\$SCOPE	595#	1369													
\$SIMBC	595#	5150													
\$SINAC	595#														
\$SOFTC	595#	1801													
\$STUFF	595#														
\$SWPAC	595#	2555	2577												
\$TCHAR	595#	4010	4109	4182	4429										
\$TCRC	595#	3991	4160	4406											
\$TRANW	595#	4029	4205	4256	4453	4523									
\$TRAN1	595#	2632	2665	2709											
\$TRPDE	595#	809	811	813	815	817	819	821	823	825	827	829	831	833	835
	837														
\$TSTN	595#	2267	2291	2314	2338	2369	2411	2453	2511	2563	2585	2607	2641	2674	2718
	2776	2827	2878	2929	2981	3039	3067	3095	3123	3152	3200	3237	3274	3311	3348
	3389	3433	3477	3524	3568	3643	3718	3793	3867	3933	4001	4103	4173	4420	4694
	4757	4796	4890												
\$VARIA	595#	728													
\$WINDO	595#	2766	2817	2868	2919										
\$XZ	595#	2259	2265	2283	2289	2306	2312	2330	2336	2361	2367	2403	2409	2445	2451
	2503	2509	2555	2561	2577	2583	2599	2605	2632	2639	2665	2672	2709	2716	2766
	2774	2817	2825	2868	2876	2919	2927	2970	2979	3031	3037	3059	3065	3087	3093
	3115	3121	3143	3150	3192	3198	3229	3235	3266	3272	3303	3309	3340	3346	3379
	3387	3423	3431	3468	3475	3516	3522	3559	3566	3634	3641	3709	3716	3784	3791
	3859	3865	3925	3931	3991	3999	4092	4101	4160	4171	4406	4418	4684	4692	4749
	4755	4784	4794	4879	4888										
\$ZEROS	595#														

. ABS. 032226 000

ERRORS DETECTED: 0

CZDMEC/I,CZDMEC.SEQ/CRF/SOL=CZDMEC

RUN-TIME: 14 19 1 SECONDS

RUN-TIME RATIO: 55/35=1.5

CORE USED: 32K (63 PAGES)