

DMC11

DDCMP MODE LINE UNIT
MD-11-DZDME-B

EP-DZDME-B-DL-A

COPYRIGHT © 76-77

FICHE 1 OF 1

AUG 1977

digital

MADE IN USA

801

ECF1CZDMCBSEQ
PDP10 PAGE: 0001

00010000 770712

PDP10 411

HDR1DZDMEBSEQ

00010000 770712

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZDME-B-D
PRODUCT NAME: DDCMP MODE LINE UNIT TESTS
DATE: MAY 1977
MAINTAINER: DIAGNOSTICS
AUTHOR: FAY BASHAW

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, 1977 by Digital Equipment Corporation

1. ABSTRACT

The function of the DMC11 diagnostics is to verify that the option operates according to specifications. The diagnostics verify that there are no malfunctions and the all operations of the DMC11 are correct in its environment.

Parameters must be set up to alert the diagnostics to the DMC11 configuration. These parameters are contained in the STATUS TABLE and are generated in two ways: 1) Manual Input - the operator answers questions. 2) Autosizing - the program determines the parameters automatically.

DZDME 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. DZDME 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. DZDMC [REV] Basic W/R and Micro-processor tests
2. DZDME [REV] DDCMP 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 PROCEEDURE

3.1 METHOD

All programs are in absolute format and are loaded using the ABSOLUTE LOADER. NOTE: if the diagnostics are on a media such as DISK, MAGTAPE, DECTAPE, or CASSETTE; follow instructions for the monitor which has been provided on that specific media.

ABSOLUTE LOADER starting address *500

MEMORY * SIZE

4k	17
8k	37
12k	57
16k	77
20k	117
24k	137
28k	157

- 3.1.1 Place address of ABS loader into switch register.
(also place 'HALT' SW up)
- 3.1.2 Depress 'LOAD ADDRESS' key on console and release.
- 3.1.3 Depress 'START KEY' on console and release (program should now be loading into CPU)

4. STARTING PROCEDURE

- a. Set switch register to 000200
- b. Depress 'LOAD ADDRESS' key and release
- c. Set SWR to zero for 'AUTO SIZING' or SWR bit0=1 for manual input (questions) or SWR bit7=1 to use existing parameters set up by a previous start or a previously run 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?16C010
VECTOR ADDRESS?310
BR PRIORITY LEVEL? (4,5,6,7)?5
DOES MICRO-PROCESSOR HAVE CRAM? (Y OR N)N
WHICH LINE UNIT? IF NONE TYPE "N", IF MB201 TYPE "1", IF MB202 TYPE "2"?1
IS THE LOOP BACK CONNECTOR ON?Y
SWITCH PAC#1 (DDCMP LINE#)?377
SWITCH PAC#2 (BM873 BOOT ADD)?377

Following the questions the status map is printed out as described above, the information in the map reflects the answers to the questions. If the diagnostic was started with SW00=0 and SW07=0 (AUTO-SIZING) then no questions are asked and only the status-map is printed out. If AUTO-SIZING is used the status information must be verified to be correct (match the hardware). If it does not match the hardware the diagnostic must be restarted with SW00=1 and the questions answered.

4.1 CONTROL SWITCH SETTINGS

SW 15 Set: Halt on error
 SW 14 Set: Loop on current test
 SW 13 Set: Inhibit error print out
 SW 12 Set: Inhibit type out/abail 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 carriage return to begin execution at the selected test.

SW 02 LOCK ON SELECTED TEST. This switch when used with SW01 will cause the program to constantly loop on the selected test. Hitting any key on the console will let it advance to the next test and loop until a key is hit again. If SW02=0 when SW01 is used. The program will begin at the selected test and continue normal operations.

SW 03 RESELECT 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 DMC11's; 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 switches greater than SW 03 in the up position. this would be a fatal error. do not select more active DMC11s than there is information on in the status table.

METHOD: A: Load address 200
B: Start with SW 00=1
C: Program will type message
D: Set a switch for each DMC desired active.
EXAMPLE: If you have 4 DMC's but only want to run the first and the last set SWR bits 0 and 3 = 1. PRESS CONTINUE
E: Number (IF VALID) will be in data lights (excluding 11/05)
F: Set with any other switch settings desired. PRESS CONTINUE.

4.1.3 DYNAMIC SWITCHES

ERROR SWITCHES

1. SW 12 Delete print out/bell on error.
2. SW 13 Delete error printout.
3. SW 15 Halt on the error.
4. SW 08 Goto beginning of the test(on error).
5. SW 10 Goto next test(on error).

SCOPE SWITCHES

1. SW06 Halt in ROMCLK routine before clocking micro-processor instruction. This allows the operator to scope a micro-processor instruction in the static state before it is clocked. Hit continue to resume running.
2. SW09 (if enabled by 'SCOPI') on an error; If an '*' is printed in front of the test no. (ex. *TEST NO. 10) SW09 is incorporated in that test and therefore SW09 is usually the best switch for the scope loop (SW14=0, SW10=0, SW09=1, SW08=0). If SW09 is not 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 iterations.
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.

5. ERRORS

As described previously there will always be a TEST NUMBER and PC typed out at the time of an error (providing SW 13=0 and SW 12=0). In most cases additional information will be supplied in the error message to give the operator an indication of the error.

5.2 ERROR RECOVERY

If for some reason the DMC11 should 'HANG THE BUS' (gain control of bus so that console manual functions are inhibited) an init or power down/up is necessary for operator to regain control of cpu. If this should happen; look in location 'TSTNO' (address 1226) for the number of the test that was running at the time of the catastrophic error. In this way the operator will have an idea as to what the DMC11 was doing at the time of the error.

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

See section 4. (PLEASE)
Status table should be verified regardless of how program was started. Also it is important to use this listing along with the information printed on the TTY to completely isolate problems.

7.2 OPERATING RESTRICTIONS

The first time a DMC11 diagnostic is loaded into core and run the STATUS TABLE must be set up. This is done by manual input (SW00=1) or by autosizing (SW00=0 and SW07=0). Thereafter however the status table need not be setup by subsequent restarts or even loading the next DMC diagnostic because the STATUS TABLE is overlayed. The current parameters in the STATUS TABLE are used when SW07=1 on start up.

7.3 HARDWARE CONFIGURATION RESTRICTIONS

DMC11(M8200)- Jumper W1 must be in, and switch 7 of E76 must be in the OFF position.

KMC(M8204)- Jumper W1 must be in.

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 DZDMC 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
DMSTOO-DMST17
(1500)-(1640)

These locations contain the information needed to test up to 16 (decimal) DMC11s sequentially. they contain the CSR, VECTOR and STATUS concerning the configuration of each DMC11.

DMACTV (1306) Each bit set in this location indicates that the associated DMC11 will be tested in turn. EXAMPLE: (DMACTV) 1276/0000000000011111 means that DMC11 no. 00,01,02,03,04 will be tested. EXAMPLE: (DMACTV) 1276/0000000000010001 Means that DMC11 no. 00,04 will be tested.

DMCSR (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

Each map entry contains 4 words which contain the status information for 1 DMC11. The PC shows where in core memory the first of the 4 words is. In the example above the first DMC'S status is in locations, 1500, 1502, 1504, and 1506. The second DMC status is located at 1510, 1512, 1514, and 1516. The information contained in each 4 word entry is defined as follows:

CSR: Contains DMC11 CSR address

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

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

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

8.5 METHOD OF AUTO SIZING

8.5.1 FINDING THE CONTROL STATUS REGISTER.

The auto-sizing routine finds a DMC11 as follows: It starts at address 160000 and tests all address in increments of 10 up to and including address 167760. If the address does not time out, the following is done, the first CROM address is written to a 125252 then it is read back. If it contains a -1 or 125252 or 626 or 16520 a DMC11 has been found, if not, the address is updated by 10 and the search continues. A -1 indicates a DMC11 with no CROM or CRAM, a 125252 indicates a KMC11 with CRAM, a 626 indicates a DMC11-AL and a 16520 indicates a DMC11-AR. Further tests are performed at this point to determine which line unit, if any, is installed, if a loop-back connector is installed and various switch settings on the line unit. THIS IS WHY THE STATUS TABLE MUST BE VERIFIED BY THE USER AND IF ANY OF THE INFORMATION DOES NOT AGREE WITH THE HARDWARE THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS MUST BE ANSWERED. All DMC11's in the system will be found by the auto-sizer. If it does not find a DMC11 the diagnostic must be restarted and the questions answered.

8.5.2 FINDING THE VECTOR AND BR LEVEL

The vector area (address 300-776) is filled with the instruction IOT and '+2' (next address). The processor status is started at 7 and the DMC is programmed to interrupt. The PS is lowered by 1 until the DMC interrupts, a delay is made and if no interrupt occurs at PS level 3 (because of a bad DMC11) the program assumes vector address 300 at BR level 5 and the problem should be fixed in the diagnostic. Once the problem is fixed, the program should be re-set up again to get correct vector. If an interrupt occurred, 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.

8.5 SOFTWARE SWITCH REGISTER

If the diagnostic is run on an 11/04 or other CPU without a switch register then a software switch register is used to allow user the same switch options as described previously. If the hardware switch register does not exist or if one does and it contains all ones (177777) this software switch register is used.

Control:

To obtain control at any allowable time during execution of the diagnostic the operator types a CTRL G on the console terminal keyboard. As soon as the CTRL G is recognized, by the diagnostic, the following message will be displayed:

SWR=XXXXXX NEW?

Where XXXXXX is the current contents of the software switch register in octal. The software control routine will then await operator action. At which time the operator is required to type one or more of the legal characters: 1) 0 - 7, 2) line feed(<LF>), 3) carriage return(<CR>), or 4) control-U (CTRL U). No check is made for legality. If the input character is not a <LF>, <CR>, or CTRL U it is assumed to be an octal digit.

To change the contents of the SSR the operator simply types the new desired value in octal - leading zeros need not be typed. And terminates the input string with a <CR> or <LF> depending on the program action desired as described below. The input value will be truncated to the last 6 digits typed. At least one digit must be typed on any given input string prior to the terminator before a change to the SSR will occur.

When the input string is terminated with a <CR> the diagnostic will continue execution from the point at which it was interrupted. If a <CR> is the only thing typed the program will continue without changing the SSR. The <LF> differs from the <CR> by restarting the program as if it were restarted at address 200.

If a CTRL U is typed at any point in the input string prior to the terminator the input value will be disregarded and the prompt displayed (SWR = XXXXXX NEW?).

To set the SSR for the starting switches, first load the diagnostic, then hit CTRL G, then start the diagnostic.

DZDME LST

B02

DECD0C VER 00.04 11-JUL-77 11:53 PAGE 01 PAGE: 0014

DOCUMENT

DZDME LST

COPYRIGHT 1977
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

6 MAINDEC-11-DZDME-B DMC11 DDCMP LINE UNIT TESTS
COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754

1667 ***** TEST 1 *****
OUT CONTROL REGISTER READ/ONLY TEST
DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
BITS ARE IN THE CORRECT STATE

1691 ***** TEST 2 *****
IN CONTROL REGISTER READ/ONLY TEST
DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
BITS ARE IN THE CORRECT STATE

1714 ***** TEST 3 *****
MODEM CONTROL REGISTER READ/ONLY TEST
DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
BITS ARE IN THE CORRECT STATE

1738 ***** TEST 4 *****
MAINTENANCE REGISTER READ/ONLY TEST
DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
BITS ARE IN THE CORRECT STATE

1769 ***** TEST 5 *****
LINE UNIT REGISTER WRITE/READ TEST
SET BITS IN LU REGISTER 12, VERIFY IT IS SET
CLEAR BITS IN LU REGISTER 12, VERIFY IT IS CLEAR

1811 ***** TEST 6 *****
LINE UNIT REGISTER WRITE/READ TEST
SET BIT1 IN LU REGISTER 17, VERIFY IT IS SET
CLEAR BIT1 IN LU REGISTER 17, VERIFY IT IS CLEAR

1853 ***** TEST 7 *****
LINE UNIT REGISTER WRITE/READ TEST
FLOAT A 1 THROUGH LINE UNIT REGISTER 13
FLOAT A 0 THROUGH LINE UNIT REGISTER 13

1911 ***** TEST 10 *****
LINE UNIT REGISTER WRITE/READ TEST
FLOAT A 1 THROUGH LINE UNIT REGISTER 14
FLOAT A 0 THROUGH LINE UNIT REGISTER 14

1963 ***** TEST 11 *****
SWITCH PAC TEST
THIS TEST READS SWITCH PAC#1
THIS SWITCH PAC CONTAINS THE DDCMP LINE #

1985 ***** TEST 12 *****
SWITCH PAC TEST
THIS TEST READS SWITCH PAC#2
THIS SWITCH PAC CONTAINS THE BM873 BOOT ADD

2007 ***** TEST 13 *****
LINE UNIT CLOCK TEST
THIS TEST VERIFYS THAT THE LU INTERNAL CLOCK
(BIT 1 IN LU-17) IS WORKING

2040 ***** TEST 14 *****
OUT DATA SILO TEST
SET SOM AND LOAD OUT DATA SILO
VERIFY THAT OCOR SET, INDICATING THAT THE
CHARACTER IS AT THE BOTTOM OF THE OUT SILO

2073 ***** TEST 15 *****
DDCMP TEST OF RTS AND OUT ACTIVE
SET SOM AND LOAD OUT DATA SILO
SINGLE STEP 2 DATA CLOCKS, VERIFY
THAT RTS AND ACTIVE ARE SET

2117 ***** TEST 16 *****
TEST OF OUT CLEAR
SET SOM AND LOAD OUT DATA SILO
SINGLE STEP DATA CLOCK, SET OUT CLEAR
VERIFY THAT OCOR, RTS, AND ACTIVE ARE CLEARED

2174 ***** TEST 17 *****
DDCMP TRANSMITTER TEST
SINGLE CLOCK THE CHARACTER 0
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

2225 ***** 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

2276 ***** TEST 21 *****
DDCMP TRANSMITTER TEST
SINGLE CLOCK THE CHARACTER 252

2279 VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

2327 ***** TEST 22 *****
DDCMP TRANSMITTER TEST
SINGLE CLOCK THE CHARACTER 377
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

2378 ***** TEST 23 *****
DDCMP TRANSMITTER TEST
SINGLE CLOCK A BINARY COUNT PATTERN
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
AND R5 CONTAINS THE CHARACTER THAT FAILED

2439 ***** TEST 24 *****
DDCMP STRIP SYNC TEST
SET LU LOOP, SINGLE STEP 5 SYNCs,
VERIFY THAT IN ACTIVE DOES NOT SET

2467 ***** TEST 25 *****
DDCMP IN ACTIVE TEST
SET LU LOOP, SINGLE STEP 5 SYNCs AND A NON-SYNC (301)
VERIFY THAT IN ACTIVE IS SET

2495 ***** 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

2523 ***** TEST 27 *****
DDCMP IN ACTIVE TEST
SET LU LOOP, SINGLE STEP 2 SYNCs AND A NON-SYNC (301)
VERIFY THAT IN ACTIVE IS SET

2551 ***** 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

2600 ***** TEST 31 *****
DDCMP BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 0
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

2637 ***** TEST 32 *****
DDCMP BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 125
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

2674 ***** TEST 33 *****
DDCMP BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 252
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

2711 ***** TEST 34 *****
DDCMP BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 377
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

2748 ***** TEST 35 *****
DDCMP DATA TEST
THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
CHECKING EACH CHARACTER AS IT IS RECEIVED

2787 ***** TEST 36 *****
DDCMP DATA TEST
THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
CHECKING EACH CHARACTER AS IT IS RECEIVED
THIS TEST IS EXACTLY THE SAME AS THE LAST TEST,
EXCEPT LINE UNIT LOOP IS SET IN LU REGISTER 12

2831 ***** TEST 37 *****
TRANSMITTER MARK TEST
SINGLE CLOCK 3 SYNCs AND A 301 AND 20 EXTRA
CLOCK TICKS, VERIFY THAT A 301, A 377 AND A 377
WERE RECEIVED INDICATING THAT THE TRANSMITTER WENT
TO A MARK STATE FOR 16 BITS WHEN OUT SILO WAS EMPTY

2876 ***** TEST 40 *****
CABLE TURNAROUND TEST
CLEAR LINE UNIT LOOP, SET DTR
VERIFY THAT MODEM READY IS SET
CLEAR DTR, VERIFY THAT MRDY IS CLEARED

2924 ***** TEST 41 *****
CABLE TURNAROUND TEST
CLEAR LINE UNIT LOOP, LOAD OUT DATA SILO
VERIFY THAT ALL MODEM SIGNALS ARE SET

2967 ***** TEST 42 *****
TEST OF CRC OPERATION
USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
0, VERIFY THE LSB OF THE BCC ON EACH SHIFT
TEST TRANSMITTER FIRST THEN THE RECEIVER BCC

3042 ***** 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

3117 ***** TEST 44 *****
TEST OF CRC OPERATION

3119 USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
125, VERIFY THE LSB OF THE BCC ON EACH SHIFT
TEST TRANSMITTER FIRST THEN THE RECEIVER BCC

3192 ***** TEST 45 *****
TEST OF CRC OPERATION
USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
252, VERIFY THE LSB OF THE BCC ON EACH SHIFT
TEST TRANSMITTER FIRST THEN THE RECEIVER BCC

3267 ***** TEST 46 *****
TRANSMITTER CRC TEST
USING THE CRC16 POLYNOMIAL, SINGLE CLOCK A BINARY
COUNT PATTERN, VERIFY THE LSB OF THE TRANSMITTER BCC ON EACH SHIFT

3333 ***** 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

3399 ***** TEST 50 *****
TRANSMITTER DDCMP CRC TEST
THIS TEST TRANSMITS A FOUR CHARACTER MESSAGE WITH CRC
BOTH DATA AND THE BCC ARE VERIFIED IN THE BIT
WINDOW. THE FOUR CHARACTERS ARE 0,125,252,377
THE TRANSMITTER IS CHECKED FOR GOING TO A MARK STATE AFTER THE BCC

3500 ***** TEST 51 *****
RECEIVER DDCMP CRC TEST
THIS TEST CLOCKS A FOUR CHARACTER MESSAGE WITH BCC
AND VERIFYS CORRECT DATA RECEPTION AND BCC MATCH
THE FOUR CHARACTER MESSAGE IS 0,125,252,377

3557 ***** TEST 52 *****
DDCMP EOM FUNCTION TEST
THIS TEST LOADS OUT SILO WITH: 2 SYNC, 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 CHARACTERS 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

3803 ***** TEST 53 *****
DDCMP EOM FUNCTION TEST
THIS TEST LOADS OUT SILO WITH: 2 SYNC, 4 CHAR MESSAGE, EOM
SOM, 4 CHAR MESS, EOM. THE DATA STREAM IS CHECKED TO BE
4 CHAR, BCC, 4 CHAR, BCC, MARKS. THIS TEST VERIFYS THAT
THE CHARACTERS LOADED WITH EOM SET ARE LOST
ALSO THAT THE CHAR LOADED WITH SOM IS NOT IN THE BCC
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

- 4081 ***** TEST 54 *****
EMPTY SILO TEST
LOAD SILO WITH 2 SYNCs, 4 CHAR MESSAGE, SINGLE CLOCK
UNTIL THE SILO IS EMPTY, LOAD 4 MORE CHARACTERS IN THE
SILO. GIVE MORE TICKS, AND VERIFY THAT ONLY THE FIRST
4 CHARACTER MESSAGE WAS RECEIVED AND THAT RTS IS CLEAR
- 4146 ***** 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
- 4181 ***** TEST 56 *****
DDCMP CABLE DATA TEST
- 4183 THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
4 SYNCs, 16 CHAR, EOM, 16 CHAR, EOM, 16 CHAR, EOM
THE 16 CHARACTERS INCLUDE A FLOATING ONE AND ZERO
THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
- 4276 ***** TEST 57 *****
DDCMP CABLE DATA TEST
THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
4 SYNCs, 59 DATA CHARACTERS, EOM WITH GARBAGE CHARACTER
THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST

*MAINDEC-11-DZDME-8 DMC11 DDCMP LINE UNIT TESTS
 *COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
 *-----

; STARTING PROCEDURE
 ; LOAD PROGRAM
 ; LOAD ADDRESS 000200
 ; SWR=0 AUTOSIZE DMC11
 ; SW07=1 USE CURRENT DMC11 PARAMETERS
 ; SW00=1 INPUT NEW DMC11 PARAMETERS
 ; PRESS START
 ; PROGRAM WILL TYPE "MAINDEC-11-DZDME-8 DMC11 DDCMP LINE UNIT TESTS"
 ; PROGRAM WILL TYPE STATUS MAP
 ; PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
 ; AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
 ; AND THEN RESUME TESTING
 ; SUBSEQUENT RESTARTS WILL NOT TYPE PROGRAM TITLE

; SWITCH REGISTER OPTIONS
 ;-----

100000
 040000
 020000
 010000
 004000
 002000
 001000
 000400
 000200
 000100
 000040
 000020
 000010
 000004
 000002
 000001

SW15=100000 ;=1, HALT ON ERROR
 SW14=40000 ;=1, LOOP ON CURRENT TEST
 SW13=20000 ;=1, INHIBIT ERROR TYPEOUT
 SW12=10000 ;=1, DELETE TYPEOUT/BELL ON ERROR.
 SW11=4000 ;=1, INHIBIT ITERATIONS
 SW10=2000 ;=1, ESCAPE TO NEXT TEST ON ERROR
 SW09=1000 ;=1, LOOP WITH CURRENT DATA
 SW08=400 ;=1, LOOP ON ERROR
 SW07=200 ;=1, USE CURRENT DMC11 PARAMETERS, =0, AUTOSIZE DMC11
 SW06=100 ;=1, HALT BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION
 SW05=40
 SW04=20
 SW03=10
 SW02=4
 SW01=2
 SW00=1
 ; RESELECT DMC11'S TO BE TESTED (ACTIVE)
 ; LOCK ON TEST SELECT
 ; RESTART PROGRAM AT SELECTED TEST
 ; INPUT DMC11 PARAMETERS

DZDME MACY11 30(1046) 11-JUL-77 11:40 PAGE 3
 DZDME.P11 12-MAY-77 14:18 GENERAL DEFINATIONS AND EQUIVALENCIES

:REGISTER DEFINITIONS

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

:LOCATION EQUIVALENCIES

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

:INSTRUCTION DEFINITIONS

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

:BIT DEFINITIONS

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

```

98
99
100
101
102
103
104
105
106
107
108      000000
109
110
111
112      000024
113      000024 005336
114      000026 000340
115      000030 004750
116      000032 000340
117      000034 004716
118      000036 000340
119      000040 000040
120      000040 000000
121      000042 000000
122      000044 000000
123      000046 003522
124      000052 000052
125      000052 000000
126
127      000174
128      000174 000000
129      000176 000000
130
131      000200
132      000200 000137 002002
133
134
135      001000
136      001000 005377 040515 047111
137      (2) 001025 104 041515 030461
138      (2)
139      001200
140
141
142      001200 177570
143      001202 177570

```

```

;*****
;-----
;TRAPCATCAER FOR ILLEGAL INTERRUPTS
;THE STANDARD "TRAP CATCHER" IS PLACED
;BETWEEN ADDRESS 0 TO ADDRESS 776.
;IT LOOKS LIKE "PC+2 HALT".
;-----
;*****
.=0
;STANDARD INTERRUPT VECTORS
;-----
.=24
      .PFAIL      ;POWER FAIL HANDLER
      340          ;SERVICE AT LEVEL 7
      .HLT        ;ERROR HANDLER
      340          ;SERVICE AT LEVEL 7
      .TRPSRV     ;GENERAL HANDLER DISPATCH SERVICE
      340          ;SERVICE AT LEVEL 7
.=40
      0            ;SAVE FOR ACT-11 OR XXDP
      0            ;RETURN ADDRESS IF UNDER ACT-11 OR XXDP
      0            ;SAVE FOR ACT-11 OR XXDP
      $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>/MAINDEC-11-DZDME-B/<377>
        .ASCIIZ /DMC11 DDCMP LINE UNIT TESTS/<377>
.=1200
;INDIRECT POINTERS TO SWITCH REGISTER AND LIGHT DISPLAY
;-----
DISPLAY:177570
SWR: 177570

```

```

144
145 ;INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS
146 ;-----
147
148 001204 177560 TKCSR: 177560 ;TELETYPE KEYBOARD CONTROL REGISTER
149 001206 177562 TKDBR: 177562 ;TELETYPE KEYBOARD DATA BUFFER
150 001210 177564 TPCSR: 177564 ;TELEPRINTER CONTROL REGISTER
151 001212 177566 TPDBR: 177566 ;TELEPRINTER DATA BUFFER
152
153 ;PROGRAM CONTROL PARAMETERS
154 ;-----
155
156 001214 000000 RETURN: 0 ;SCOPE ADDRESS FOR LOOP ON TEST
157 001216 000000 NEXT: 0 ;ADDRESS OF NEXT TEST TO BE EXECUTED
158 001220 000000 LOCK: 0 ;ADDRESS FOR LOCK ON CURRENT DATA
159 001222 000003 ICOUNT: 3 ;NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE
160 001224 000000 LPCNT: 0 ;NUMBER OF ITERATIONS COMPLETED
161 001226 000000 TSTNO: 0 ;NUMBER OF TEST IN PROGRESS
162 001230 000000 PASCNT: 0 ;NUMBER OF PASSES COMPLETED
163 001232 000000 ERRCNT: 0 ;TOTAL NUMBER OF ERRORS
164 001234 000000 LSTERR: 0 ;PC OF LAST ERROR CALL
165
166 ;PROGRAM VARIABLES
167 ;-----
168
169 001236 000000 STRTSW: 0 ;SWITCHES AT START OF PROGRAM
170 001240 000000 STAT: 0 ;DM STATUS WORD STORAGE
171 001242 000000 CLKX: 0
172 001244 000000 MASKX: 0
173 001246 000000 TEMP1: 0 ;TEMPORARY STORAGE
174 001250 000000 TEMP2: 0 ;TEMPORARY STORAGE
175 001252 000000 TEMP3: 0 ;TEMPORARY STORAGE
176 001254 000000 TEMP4: 0 ;TEMPORARY STORAGE
177 001256 000000 TEMPS: 0 ;TEMPORARY STORAGE
178 001260 000000 SAVR0: 0 ;R0 STORAGE
179 001262 000000 SAVR1: 0 ;R1 STORAGE
180 001264 000000 SAVR2: 0 ;R2 STORAGE
181 001266 000000 SAVR3: 0 ;R3 STORAGE
182 001270 000000 SAVR4: 0 ;R4 STORAGE
183 001272 000000 SAVR5: 0 ;R5 STORAGE
184 001274 000000 SAVSP: 0 ;STACK POINTER STORAGE
185 001276 000000 SAVPC: 0 ;PROGRAM COUNTER STORAGE
186 001300 000000 ZERO: 0
187 001302 000001 ONE: 1
188 001304 000000 MEMLIM: 0 ;HIGHEST LOCATION FOR NPR'S
189 001306 000001 DMACTV: .BLKW 1 ;DMC11'S SELECTED ACTIVE.
190 001310 000001 DMNUM: .BLKW 1 ;OCTAL NUMBER OF DMC11'S.
191 001312 000001 SAVACT: .BLKW 1 ;ORIGINAL ACTV DEVICES
192 001314 000001 SAVNUM: .BLKW 1 ;WORKABLE NUMBER
193 001316 000000 RUN: 0 ;POINTER TO RUNNING DEVICE.
194
195 001320 001472 .EVEN
196 001322 001676 CREAM: DM.MAP-6 ;TABLE POINTER.
MILK: CNT.MAP-4 ;TABLE POINTER

```

```

197
198
199
200
201 001324 000
202 001325 000
203 001326 000
204 001327 000
205
206
207
208
209
210
211
212
213
214 001330
215 001330 104400
216 001330 003576
217 001330 104401
218 001332 003736
219 001332 104402
220 001334 003766
221 001334 104403
222 001336 004050
223 001336 104404
224 001340 004154
225 001340 104405
226 001342 004174
227 001342 104406
228 001344 004374
229 001344 104407
230 001346 004434
231 001346 104410
232 001350 004466
233 001350 104411
234 001352 004472
235 001352 104412
236 001354 005466
237 001354 104413
238 001356 005436
239 001356 104414
240 001360 005504
241 001360 104415
242 001362 005552
243 001362 104416
244 001364 005616
245
246
247

```

```

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

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

;*****
;-----
;TRPTAB:
SCOPE=TRAP+0 ;CALL TO SCOPE LOOP AND ITERATION HANDLER
;SCOPE
SCOPI=TRAP+1 ;CALL TO LOOP ON CURRENT DATA HANDLER
;SCOPI
TYPE=TRAP+2 ;CALL TO TELETYPE OUTPUT ROUTINE
;TYPE
INSTR=TRAP+3 ;CALL TO ASCII STRING INPUT ROUTINE
;INSTR
INSTER=TRAP+4 ;CALL TO INPUT ERROR HANDLER
;INSTER
PARAM=TRAP+5 ;CALL TO NUMERICAL DATA INPUT ROUTINE
;PARAM
SAVOS=TRAP+6 ;CALL TO REGISTER SAVE ROUTINE
;SAVOS
RESOS=TRAP+7 ;CALL TO REGISTER RESTORE ROUTINE
;RESOS
CONVRT=TRAP+10 ;CALL TO DATA OUTPUT ROUTINE
;CONVRT
CNVRT=TRAP+11 ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
;CNVRT
MSTCLR=TRAP+12 ;CALL TO ISSUE A MASTER CLEAR
;MSTCLR
DELAY=TRAP+13 ;CALL TO DELAY
;DELAY
ROMCLK=TRAP+14 ;CALL TO CLOCK ROM ONCE
;ROMCLK
DATACLK=TRAP+15 ;CALL TO CLK DATA
;DATACLK
TIMER=TRAP+16 ;CALL TO DELAY A CLOCK TICK
;TIMER
;-----
;*****

```

```

248 ;DMC11 CONTROL INDICATORS FOR CURRENT DMC11 UNDER TEST
249 ;-----
250
251 001366 000000 STAT1: 0
252 001370 000000 STAT2: 0
253 001372 000000 STAT3: 0
254
255 ;DMC11 VECTOR AND REGISTER INDIRECT POINTERS
256 ;-----
257
258 001374 000000 DMRVEC: 0 ; POINTER TO DMC11 RECEIVER INTERRUPT VECTOR
259 001376 000000 DMRLVL: 0 ; POINTER TO DMC11 RECEIVER INTERRUPT SERVICE PS
260 001400 000000 DMTVEC: 0 ; POINTER TO DMC11 TRANSMITTER INTERRUPT VECTOR
261 001402 000000 DMTLVL: 0 ; POINTER TO DMC11 TRANSMITTER INTERRUPT SERVICE PS
262 001404 000000 DMCSR: 0 ; POINTER TO DMC11 CONTROL STATUS REGISTER
263 001406 000000 DMCSRH: 0 ; POINTER TO DMC11 CONTROL STATUS REGISTER HIGH BYTE.
264 001410 000000 DMCTL: 0 ; POINTER TO DMC11 CONTROL OUT REGISTER
265 001412 000000 DMP04: 0 ; POINTER TO DMC11 PORT REGISTER(SEL 4)
266 001414 000000 DMP06: 0 ; POINTER TO DMC11 PORT REGISTER(SEL 6)
267
268 ;TEMP STORAGE
269 ;-----
270
271 001416 000000 TEMP: 0
272 001460 . = .+40
273
274 ;DMC11 STATUS TABLE AND ADDRESS ASSIGNMENTS
275 ;-----
276
277 . = 1500
278 001500 DM.MAP:
279 001500 000001 DMCRO0: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 00
280 001502 000001 DMS100: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 00
281 001504 000001 DMS200: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 00
282 001506 000001 DMS300: .BLKW 1 ; 3RD STATUS WORD
283
284 001510 000001 DMCRO1: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 01
285 001512 000001 DMS101: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 01
286 001514 000001 DMS201: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 01
287 001516 000001 DMS301: .BLKW 1 ; 3RD STATUS WORD
288
289 001520 000001 DMCRO2: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 02
290 001522 000001 DMS102: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 02
291 001524 000001 DMS202: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 02
292 001526 000001 DMS302: .BLKW 1 ; 3RD STATUS WORD
293
294 001530 000001 DMCRO3: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 03
295 001532 000001 DMS103: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 03
296 001534 000001 DMS203: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 03
297 001536 000001 DMS303: .BLKW 1 ; 3RD STATUS WORD
298
299 001540 000001 DMCRO4: .BLKW 1 ; CONTROL STATUS REGISTER FOR DMC11 NUMBER 04
300 001542 000001 DMS104: .BLKW 1 ; VECTOR FOR DMC11 NUMBER 04
301 001544 000001 DMS204: .BLKW 1 ; DDCMP LINE# FOR DMC11 NUMBER 04
302 001546 000001 DMS304: .BLKW 1 ; 3RD STATUS WORD
303

```

304	001550	000001	DMCR05: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 05
305	001552	000001	DMS105: .BLKW	1	;VECTOR FOR DMC11 NUMBER 05
306	001554	000001	DMS205: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 05
307	001556	000001	DMS305: .BLKW	1	;3RD STATUS WORD
308					
309	001560	000001	DMCR06: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 06
310	001562	000001	DMS106: .BLKW	1	;VECTOR FOR DMC11 NUMBER 06
311	001564	000001	DMS206: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 06
312	001566	000001	DMS306: .BLKW	1	;3RD STATUS WORD
313					
314	001570	000001	DMCR07: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 07
315	001572	000001	DMS107: .BLKW	1	;VECTOR FOR DMC11 NUMBER 07
316	001574	000001	DMS207: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 07
317	001576	000001	DMS307: .BLKW	1	;3RD STATUS WORD
318					
319	001600	000001	DMCR10: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 10
320	001602	000001	DMS110: .BLKW	1	;VECTOR FOR DMC11 NUMBER 10
321	001604	000001	DMS210: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 10
322	001606	000001	DMS310: .BLKW	1	;3RD STATUS WORD
323					
324	001610	000001	DMCR11: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 11
325	001612	000001	DMS111: .BLKW	1	;VECTOR FOR DMC11 NUMBER 11
326	001614	000001	DMS211: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 11
327	001616	000001	DMS311: .BLKW	1	;3RD STATUS WORD
328					
329	001620	000001	DMCR12: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 12
330	001622	000001	DMS112: .BLKW	1	;VECTOR FOR DMC11 NUMBER 12
331	001624	000001	DMS212: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 12
332	001626	000001	DMS312: .BLKW	1	;3RD STATUS WORD
333					
334	001630	000001	DMCR13: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 13
335	001632	000001	DMS113: .BLKW	1	;VECTOR FOR DMC11 NUMBER 13
336	001634	000001	DMS213: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 13
337	001636	000001	DMS313: .BLKW	1	;3RD STATUS WORD
338					
339	001640	000001	DMCR14: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 14
340	001642	000001	DMS114: .BLKW	1	;VECTOR FOR DMC11 NUMBER 14
341	001644	000001	DMS214: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 14
342	001646	000001	DMS314: .BLKW	1	;3RD STATUS WORD
343					
344	001650	000001	DMCR15: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 15
345	001652	000001	DMS115: .BLKW	1	;VECTOR FOR DMC11 NUMBER 15
346	001654	000001	DMS215: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 15
347	001656	000001	DMS315: .BLKW	1	;3RD STATUS WORD
348					
349	001660	000001	DMCR16: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 16
350	001662	000001	DMS116: .BLKW	1	;VECTOR FOR DMC11 NUMBER 16
351	001664	000001	DMS216: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 16
352	001666	000001	DMS316: .BLKW	1	;3RD STATUS WORD
353					
354	001670	000001	DMCR17: .BLKW	1	;CONTROL STATUS REGISTER FOR DMC11 NUMBER 17
355	001672	000001	DMS117: .BLKW	1	;VECTOR FOR DMC11 NUMBER 17
356	001674	000001	DMS217: .BLKW	1	;DDCMP LINE# FOR DMC11 NUMBER 17
357	001676	000001	DMS317: .BLKW	1	;3RD STATUS WORD
358					
359	001700	000000	DM.END: 000000		

```

360
361
362
363
364 001702 CNT.MAP:
365 001702 000000 PACT00: 0 ;PASS COUNT FOR DMC11 NUMBER 00
366 001704 000000 ERCT00: 0 ;ERROR COUNT FOR DMC11 NUMBER 00
367
368 001706 000000 PACT01: 0 ;PASS COUNT FOR DMC11 NUMBER 01
369 001710 000000 ERCT01: 0 ;ERROR COUNT FOR DMC11 NUMBER 01
370
371 001712 000000 PACT02: 0 ;PASS COUNT FOR DMC11 NUMBER 02
372 001714 000000 ERCT02: 0 ;ERROR COUNT FOR DMC11 NUMBER 02
373
374 001716 000000 PACT03: 0 ;PASS COUNT FOR DMC11 NUMBER 03
375 001720 000000 ERCT03: 0 ;ERROR COUNT FOR DMC11 NUMBER 03
376
377 001722 000000 PACT04: 0 ;PASS COUNT FOR DMC11 NUMBER 04
378 001724 000000 ERCT04: 0 ;ERROR COUNT FOR DMC11 NUMBER 04
379
380 001726 000000 PACT05: 0 ;PASS COUNT FOR DMC11 NUMBER 05
381 001730 000000 ERCT05: 0 ;ERROR COUNT FOR DMC11 NUMBER 05
382
383 001732 000000 PACT06: 0 ;PASS COUNT FOR DMC11 NUMBER 06
384 001734 000000 ERCT06: 0 ;ERROR COUNT FOR DMC11 NUMBER 06
385
386 001736 000000 PACT07: 0 ;PASS COUNT FOR DMC11 NUMBER 07
387 001740 000000 ERCT07: 0 ;ERROR COUNT FOR DMC11 NUMBER 07
388
389 001742 000000 PACT10: 0 ;PASS COUNT FOR DMC11 NUMBER 10
390 001744 000000 ERCT10: 0 ;ERROR COUNT FOR DMC11 NUMBER 10
391
392 001746 000000 PACT11: 0 ;PASS COUNT FOR DMC11 NUMBER 11
393 001750 000000 ERCT11: 0 ;ERROR COUNT FOR DMC11 NUMBER 11
394
395 001752 000000 PACT12: 0 ;PASS COUNT FOR DMC11 NUMBER 12
396 001754 000000 ERCT12: 0 ;ERROR COUNT FOR DMC11 NUMBER 12
397
398 001756 000000 PACT13: 0 ;PASS COUNT FOR DMC11 NUMBER 13
399 001760 000000 ERCT13: 0 ;ERROR COUNT FOR DMC11 NUMBER 13
400
401 001762 000000 PACT14: 0 ;PASS COUNT FOR DMC11 NUMBER 14
402 001764 000000 ERCT14: 0 ;ERROR COUNT FOR DMC11 NUMBER 14
403
404 001766 000000 PACT15: 0 ;PASS COUNT FOR DMC11 NUMBER 15
405 001770 000000 ERCT15: 0 ;ERROR COUNT FOR DMC11 NUMBER 15
406
407 001772 000000 PACT16: 0 ;PASS COUNT FOR DMC11 NUMBER 16
408 001774 000000 ERCT16: 0 ;ERROR COUNT FOR DMC11 NUMBER 16
409
410 001776 000000 PACT17: 0 ;PASS COUNT FOR DMC11 NUMBER 17
411 002000 000000 ERCT17: 0 ;ERROR COUNT FOR DMC11 NUMBER 17
412

```

413

FORMAT OF STATUS TABLE

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	CSR
I	C	O	N	T	R	O	L	I	R	E	G	I	S	T	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	*	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	B	I	M	I	A	D	I	D	*	I	*	I	*	STAT2
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	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	
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 CROM
 BIT15=0 MICRO-PROCESSOR HAS CROM
 BIT14=1 ??? TURNAROUND CONNECTOR IS ON
 BIT14=0 NO TURNAROUND CONNECTOR
 BIT13=0 LINE UNIT IS AN M8201
 BIT13=1 LINE UNIT IS AN M8202
 BIT12=1 NO LINE UNIT
 BITS 09-11 IS DMC11 BR PRIORITY LEVEL
- STAT2: LOW BYTE IS SWITCH PAC#1 (DDCMP LINE NUMBER)
 HIGH BYTE IS SWITCH PAC#2 (BM873 BOOT ADD.)
- STAT3: BIT0=1 DO FREE RUNNING TESTS ON KMC
 (MUST BE SET TO A ONE MANUALLY [PROGRAM DZDME 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

```

468      ;PROGRAM INITIALIZATION
469      ;LOCK OUT INTERRUPTS
470      ;SET UP PROCESSOR STACK
471      ;SET UP POWER FAIL VECTOR
472      ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
473      ;TYPE TITLE MESSAGE
474
475      .START: MOV      #340,PS      ;LOCK OUT INTERRUPTS
476              MOV      #STACK,SP    ;SET UP STACK
477              MOV      #.PFAIL,2#24 ;SET UP POWER FAIL VECTOR
478              MOV      DMNUM,SAVNUM  ;SAVE NUMBER OF DEVICES IN SYSTEM.
479              CLR      SWFLG         ;CLEAR SOFT TIMEOUT FLAG
480              CLRB     ERRFLG        ;CLEAR ERROR FLAG
481              CLRB     QV.FLG        ;ZERO QUICK VERIFY FLAG
482              MOV      #DM.MAP-10,CREAM ;GET MAP POINTER.
483              MOV      #CNT.MAP-4,MILK ;GET PASS COUNT MAP POINTER
484              MOV      #BIT15,RUN     ;POINT POINTER TO FIRST DEVICE.
485              MOV      #CNT.MAP,RO    ;PASS COUNT POINTER TO RO
486              CLR      (RO)+          ;CLEAR TABLE
487              CMP      #CNT.MAP+100,RO ;DONE YET?
488              BNE     23$            ;KEEP GOING
489              CLR      LSTERR         ;CLEAR LAST ERROR POINTER
490              MOV      #1,TSTNO       ;SET UP FOR TEST 1
491              MOV      #.START,RETURN ;SET UP FOR POWER FAIL BEFORE
492              ;TESTING STARTS
493              ;SAVE CURRENT VECTORS
494              MOV      2#6,-(SP)
495              MOV      2#4,-(SP)
496              MOV      2#5,2#4      ;SET UP FOR TIMEOUT
497              MOV      #177570,SWR   ;SET SWR TO HARD SWR ADDRESS
498              MOV      #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
499              CMP      #-1,2#SWR     ;REFERENCE HARDWARE SWITCH REGISTER
500              BEQ     6$+2           ;IF = -1 USE SOFT SWR ANYWAY
501              BR      7$            ;IF IT EXISTS AND NOT = -1 USE HARD SWR
502              CMP      (SP)+,(SP)+   ;ADJUST STACK
503              MOV      #SWREG,SWR    ;POINTER TO SOFT SWR
504              MOV      #DISPREG,DISPLAY ;POINTER TO SOFT DISPLAY REG
505              MOV      (SP)+,2#4     ;RESTORE VECTORS
506              MOV      (SP)+,2#6
507              TSTB     INIFLG        ;HAS INITIALIZATION BEEN PERFORMED
508              BNE     20$            ;BR IF YES
509              CMP      #SENDAD,2#42  ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
510              BEQ     20$
511              TYPE     MTITLE        ;TYPE TITLE MESSAGE
512              JSR      PC,CKSWR      ;CHECK FOR SOFT SWR
513              MOV      2#SWR,STRTSW  ;STORE STARTING SWITCHES
514              TST      2#42         ;IS IT RUNNING IN AUTO MODE?
515              BEQ     .+6            ;BR IF NO
516              CLR      STRTSW        ;IF YES, CLEAR SWITCHES
517              BIT      #SW00,STRTSW  ;IF SW00=1, QUESTIONS ARE ASKED.
518              BNE     17$            ;BR IF SW00=1
519              TSTB     STRTSW        ;BIT7=1??
520              BPL      17$            ;BR IF SW07=0
521              TST      DMACTV        ;ARE ANY DEVICES SELECTED?
522              BNE     16$            ;BR IF YES
523              TYPE     NOACT         ;NO DEVICES SELECTED.

```

```

524 002312 000000      HALT                ;STOP THE SHOW
525 002314 000776      BR                    ;DISQUALIFY CONTINUE SWITCH
526 002316 004737 010512 17$: JSR          PC,AUTO.SIZE ;GO DO THE AUTO SIZE
527 002322 105737 001324 16$: TSTB        INIFLG      ;FIRST TIME?
528 002326 001410      BEQ                    ;BR IF YES
529 002330 105737 001236 TSTB        STARTSW        ;IF USING SAME PARAMETERS DONT TYPE MAP
530 002334 100431      BMI                    ;
531 002336 032737 000006 001236 BIT      #BIT1:BIT2,STARTSW ;IS TEST NO. OR LOCK SELECTED
532 002344 001403      BEQ                    ;IF NO THEN TYPE STATUS
533 002346 000424      BR                    ;IF YES DO NOT TYPE STATUS
534 002350 005137 001324 21$: COM          INIFLG      ;SET FLAG
535 002354 104402 006224 24$: TYPE        XHEAD      ;TYPE HEADER
536 002360 012704 001500      MOV          #DM.MAP,R4   ;SET POINTER
537 002364 010437 001246 5$:  MOV          R4,TEMP1     ;SET ADDRESS
538 002370 012437 001250      MOV          (R4)+,TEMP2   ;SET CSR
539 002374 001411      BEQ                    ;ALL DONE IF ZERO
540 002376 012437 001252      MOV          (R4)+,TEMP3   ;SET STAT1
541 002402 012437 001254      MOV          (R4)+,TEMP4   ;SET STAT2
542 002406 012437 001256      MOV          (R4)+,TEMP5   ;SET STAT3
543 002412 104410      CONVRT                ;TYPE OUT STATUS MAP
544 002414 007454      XSTATO                ;
545 002416 000762      BR                    ;
546 002420 012700 001500 1$:  MOV          #DM.MAP,R0   ;R0 POINTS TO STATUS TABLE

```

```

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

```

```

563 002424 013746 000004      MOV          @#4,-(SP)    ;SAVE LOC 4
564 002430 013746 000006      MOV          @#6,-(SP)    ;SAVE LOC 6
565 002434 005037 000006      CLR          @#6         ;CLEAR VEC+2
566 002440 005037 001252      CLR          TEMP3        ;CLEAR FLAG
567 002444 005005      CLR          R5                 ;R5=0=DMC, R5=-1=KMC
568 002446 011037 001404 AUSTRT: MOV        (R0),DMCSR   ;GET NEXT DMC CSR
569 002452 001564      BEQ          AUDONE             ;BR IF DONE
570 002454 005705      TST          R5                 ;DMC OR KMC?
571 002456 001005      BNE          1$                 ;BR IF KMC
572 002460 032760 100000 000002 BIT        #BIT15,2(R0) ;CHECK FOR DMC CSR
573 002466 001061      BNE          SKIP               ;SKIP IF NOT DMC
574 002470 000404      BR          2$                 ;ITS A DMC SO CONTINUE
575 002472 032760 100000 000002 1$: BIT      #BIT15,2(R0) ;CHECK FOR KMC CSR
576 002500 001454      BEQ          SKIP               ;SKIP IF NOT KMC
577 002502 012737 002674 000004 2$: MOV      #NODEV,@#4 ;SET UP FOR TIMEOUT
578 002510 005705      TST          R5                 ;DMC OR KMC?
579 002512 001003      BNE          3$                 ;BR IF KMC

```

580	002514	012703	000006		MOV	#6,R3	;R3 IS COUNT OF DEVICES BEFORE DMC
581	002520	000402			BR	4\$	;GO ON
582	002522	012703	000010	3\$:	MOV	#10,R3	;R3 IS COUNT OF DEVICES BEFORE KMC
583	002526	012702	003010	4\$:	MOV	#DEV TAB,R2	;R2 IS DEVICE TABLE POINTER
584	002532	012701	160010		MOV	#160010,R1	;START WITH ADDRESS 160010
585	002536	005711		FLOAT:	TST	(R1)	;CHECK ADDRESS IN R1
586	002540	111204			MOVB	(R2),R4	;IF NO TIMEOUT, GET NEXT ADDRESS
587	002542	060401			ADD	R4,R1	;IN R1
588	002544	005201			INC	R1	
589	002546	040401			BIC	R4,R1	
590	002550	005703			TST	R3	;ANY MORE DEVICES TO CHECK FOR?
591	002552	001371			BNE	FLOAT	;BR IF YES
592	002554	012737	002700 000004		MOV	#ERR,2#4	;OK ONLY DMC'S ARE LEFT, SET UP FOR TIMEOUT
593	002562	010137	003022		MOV	R1,XLOC	;SAVE FIRST DMC/KMC ADDRESS
594	002566	005705		FY:	TST	R5	;DMC OR KMC?
595	002570	001005			BNE	1\$	;BR IF KMC
596	002572	032760	100000 000002		BIT	#BIT15,2(R0)	;CHECK FOR DMC CSR
597	002600	001014			BNE	SKIP	;SKIP IF NOT DMC
598	002602	000404			BR	2\$	;ITS A DMC SO CONTINUE
599	002604	032760	100000 000002	1\$:	BIT	#BIT15,2(R0)	;CHECK FOR KMC CSR
600	002612	001407			BEQ	SKIP	;SKIP IF NOT KMC
601	002614	005711		2\$:	TST	(R1)	;CHECK DMC ADDRESS
602	002616	020137	001404		CMP	R1,DMCSR	;DOES IT MATCH
603	002622	001411			BEQ	OK	;BR IF YES
604	002624	062701	000010		ADD	#10,R1	;GET NEXT DMC ADDRESS
605	002630	000756			BR	FY	;DO IT AGAIN
606	002632	062700	000010	SKIP:	ADD	#10,R0	;SKIP TO NEXT CSR IN TABLE
607	002636	011037	001404		MOV	(R0),DMCSR	;GET NEXT CSR
608	002642	001470			BEQ	AUDONE	;BR IF DONE
609	002644	000750			BR	FY	;ELSE CONTINUE
610	002646	062700	000010	OK:	ADD	#10,R0	;SKIP TO NEXT DMC CSR
611	002652	062737	000010 003022		ADD	#10,XLOC	;UPDATE EXPECTED DMC/KMC ADDRESS
612	002660	011037	001404		MOV	(R0),DMCSR	;GET NEXT DMC/KMC CSR
613	002664	001457			BEQ	AUDONE	;BR IF DONE
614	002666	013701	003022		MOV	XLOC,R1	;GET EXPECTED DMC/KMC ADDRESS
615	002672	000735			BR	FY	;CONTINUE
616	002674	122243		NODEV:	CMPB	(R2)+,-(R3)	;ON TIMEOUT, INC R2, DEC R3
617	002676	000002			RTI		;RETURN
618	002700	005737	001252	ERR:	TST	TEMP3	;CHECK FLAG IF = 0 TYPE HEADER
619	002704	001014			BNE	1\$	;SKIP HEADER
620	002706	104402			TYPE		;TYPEOUT HEADER MESSAGE
621	002710	007223			CONERR		;CONFIGURATION ERROR!!!!
622	002712	012737	002700 001276		MOV	#ERR,SAVPC	;SAVE PC FOR TYPEOUT
623	002720	104411			CNVRT		;TYPE OUT ERROR PC
624	002722	002770			ERRPC		
625	002724	104402			TYPE		;TYPE REST OF HEADER
626	002726	007277			CNERR		
627	002730	012737	177777 001252		MOV	#-1,TEMP3	;SET FLAG SO IT ONLY GETS TYPED ONCE
628	002736	010137	001262	1\$:	MOV	R1,SAVR1	;SAVE R1 FOR TYPEOUT
629	002742	104410			CNVRT		
630	002744	002776			CON TAB		;TYPE CSR VALUES
631	002746	005705			TST	R5	;DMC OR KMC ?
632	002750	001003			BNE	3\$	;BR IF KMC
633	002752	104402			TYPE		
634	002754	007320			DMCM		
635	002756	000402			BR	4\$	;CONTINUE

636	002760	104402		3\$: TYPE		
637	002762	007330		KMCM		
638	002764	022626		4\$: CMP	(SP)+, (SP)+	:ADJUST STACK
639	002766	000727		BR	OK	:BR TO GET CLT
640	002770	000001		ERRPC: 1		
641	002772	006	002	.BYTE	6,2	
642	002774	001276		SAVPC		
643	002776	000002		CONTAB: 2		
644	003000	006	004	.BYTE	6,4	
645	003002	003022		XLOC		
646	003004	006	002	.BYTE	6,2	
647	003006	001404		DMCSR		
648	003010	007		DEVTAB: .BYTE	7	:DJ
649	003011	017		.BYTE	17	:DH
650	003012	007		.BYTE	7	:DQ
651	003013	007		.BYTE	7	:DL
652	003014	007		.BYTE	7	:DUP
653	003015	007		.BYTE	7	:LK
654	003016	007		.BYTE	7	:DMC
655	003017	007		.BYTE	7	:DZ
656	003020	007		.BYTE	7	:KMC
657		003022		.EVEN		
658	003022	000000		XLOC: 0		
659	003024	005705		AUDONE: TST	R5	:DMC?
660	003026	001005		BNE	1\$	:BR IF KMC AND ALL DONE
661	003030	012705	177777	MOV	#-1, R5	:SET R5 TO -1 (KMC)
662	003034	012700	001500	MOV	#DM.MAP, R0	:RESET R0 TO START OF TABLE
663	003040	000602		BR	AUSTRT	:GO DO KMC'S
664	003042	012637	000006	1\$: MOV	(SP)+, #6	:RESTORE LOC 6
665	003046	012637	000004	MOV	(SP)+, #4	:RESTORE LOC 4
666	003052	032737	000010 001236	BIT	#SW03, STRTSW	:SELECT SPECIFIC DEVICES??
667	003060	001422		BEG	3\$	:BR IF NO.
668	003062	104402	006144	TYPE	MNEW	:TYPE THE MESSAGE.
669	003066	005000		CLR	R0	:ZERO DATA LIGHTS
670	003070	000000		HALT		:WAIT FOR USER TO TELL WHAT DEVICES TO RUN
671	003072	027737	176104 001312	CMP	#SWR, SAVACT	:IS THE NUMBER VALID?
672	003100	101404		BLOS	2\$	:BR IF NUMBER IS OK.
673	003102	104402	006005	TYPE	.MERP3	:TELL USER OF INVALID NUMBER.
674	003106	000000		HALT		:STOP EVERY THING.
675	003110	000776		BR	-2	:RESTART THE PROGRAM AGAIN.
676	003112	017737	176064 001306	2\$: MOV	#SWR, DMACTV	:GET NEW DEVICE PATTERN
677	003120	013700	001306	MOV	DMACTV, R0	:SHOW THE USER WHAT HE SELECTED.
678	003124	000000		HALT		:CONTINUE DYNAMIC SWITCHES.
679	003126	012700	000300	3\$: MOV	#300, R0	:PREPARE TO CLEAR THE FLOATING
680	003132	012701	000302	MOV	#302, R1	:VECTOR AREA. 300-776
681	003136	010120		4\$: MOV	R1, (R0)+	:START PUTTING "PC+2 - HALT"
682	003140	005021		CLR	(R1)+	:IN VECTOR AREA.
683	003142	022021		CMP	(R0)+, (R1)+	:POP POINTERS
684	003144	022700	001000	CMP	#1000, R0	:ALL DONE??
685	003150	001372		BNE	4\$	:BR IF NO.
686						
687				:TEST START AND RESTART		
688				:-----		
689						
690	003152	012706	001200	.BEGIN: MOV	#STACK, SP	:SET UP STACK
691	003156	013746	000006	MOV	#6, -(SP)	:SAVE LOC 6

692	003162	013746	000004		MOV	2#4, -(SP)	:SAVE LOC 4
693	003166	005000			CLR	RO	:START AT 0
694	003170	012737	003234	000004	MOV	2#5, 2#4	:SET UP FOR TIME OUT
695	003176	005037	000006		CLR	2#6	:TO AUTOSIZE MEMORY
696	003202	005720			6\$: TST	(RO)+	:CHECK ADDRESS IN RO
697	003204	022700	157776		CMP	2#157776, RO	:IS IT AT LEAST 28K
698	003210	001374			BNE	6\$	:BR IF NO
699	003212	162700	007776		SUB	2#7776, RO	:SAVE 2K FOR MONITORS
700	003216	010037	001304		7\$: MOV	RO, MEM LIM	:STORE MEMORY LIMIT
701	003222	012637	000004		MOV	(SP)+, 2#4	:RESTORE LOC 4
702	003226	012637	000006		MOV	(SP)+, 2#6	:RESTORE LOC 6
703	003232	000413			BR	10\$	:CONTINUE
704	003234	022626			2\$: CMP	(SP)+, (SP)+	:ADJUST STACK
705	003236	162700	000004		SUB	2#4, RO	:GET LAST GOOD ADDRESS
706	003242	162700	007776		SUB	2#7776, RO	:SAVE 2K FOR MONITORS
707	003246	022700	030000		CMP	2#30000, RO	:IS IT 8K?
708	003252	001361			BNE	7\$	:BR IF NO
709	003254	012700	037400		MOV	2#37400, RO	:IF 8K DON'T SAVE 2K
710	003260	000756			BR	7\$	
711	003262	012737	000340	177776	10\$: MOV	2#340, PS	:LOCK OUT INTERRUPTS
712	003270	032737	000004	001236	BIT	2#BIT2, STRTSW	:CHECK FOR LOCK ON TEST
713	003276	001411			BEG	1\$	:BR IF NO LOCK DESIRED.
714	003300	104402	006043		TYPE	, MLOCK	:TYPE LOCK SELECTED.
715	003304	012737	000240	003612	MOV	2#NOP, TTST	:ADJUST SCOPE ROUTINE.
716	003312	012737	000240	003614	MOV	2#NOP, TTST+2	:SET UP TO LOCK
717	003320	000406			BR	3\$	:CONTINUE ALONG.
718	003322	013737	003730	003612	1\$: MOV	BRW, TTST	:PREPARE NORMAL SCOPE ROUTINE
719	003330	013737	003732	003614	MOV	BRX, TTST+2	:LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
720	003336	012737	010060	001214	3\$: MOV	2#CYCLE, RETURN	:START AT "CYCLE" FIND WHICH DEVICE TO TEST
721	003344	032737	000002	001236	4\$: BIT	2#SW01, STRTSW	:IS TEST NO. SELECTED?
722	003352	001002			BNE	5\$	:BR IF YES
723	003354	104402	005755		TYPE	, MR	:TYPE R
724	003360	000177	175630		5\$: JMP	2#RETURN	:START TESTING

```

725                                     ;END OF PASS
726                                     ;TYPE NAME OF TEST
727                                     ;UPDATE PASS COUNT
728                                     ;CHECK FOR EXIT TO ACT-11
729                                     ;RESTART TEST
730
731 003364 000005 .EOP: RESET                                     ;MAKE THE WORLD CLEAN AGAIN.
732 003366 005037 CLR LSTERR                                     ;CLEAR LAST ERROR PC
733 003372 105037 CLRB ERRFLG                                     ;CLEAR ERROR FLAG
734 003376 005237 INC PASCNT                                     ;UPDATE PASS COUNT
735 003402 013777 MOV PASCNT,ADISPLAY ;DISPLAY PASS COUNT
736 003410 104402 TYPE ,MEPASS ;TYPE END PASS
737 003414 104402 TYPE ,MCSRX ;TYPE CSR
738 003420 104411 CNVRT ,XCSR ;SHOW IT
739 003424 104402 TYPE ,MVECX ;TYPE VECTOR
740 003430 104411 CNVRT ,XVEC ;SHOW IT
741 003434 104402 TYPE ,MPASSX ;TYPE PASSES
742 003440 104411 CNVRT ,XPASS ;SHOW IT
743 003444 104402 TYPE ,MERRX ;TYPE ERRORS
744 003450 104411 CNVRT ,XERR ;SHOW IT
745 003454 013700 MOV MILK,RO ;GET POINTER TO PASS COUNT
746 003460 013720 MOV PASCNT,(RO)+ ;STORE PASS COUNT FOR THIS DMC1!
747 003464 013720 MOV ERRCNT,(RO)+ ;STORE ERROR COUNT FOR THIS DMC1!
748 003470 005337 DEC SAVNUM ;ARE ALL DEVICES TESTED?
749 003474 001017 BNE RESTRT ;BR IF NO.
750 003476 112737 000377 001327 MOVVB #377,QV.FLG ;SET THE QUICK VERIFY FLAG.
751 003504 013737 001310 001314 MOV DMNUM,SAVNUM ;RESTORE THE COUNT
752 003512 013701 000042 MOV #42,R1 ;CHECK FOR ACT-11 OR DDP
753 003516 001406 BEQ RESTRT ;IF NOT, CONTINUE TESTING
754 003520 000005 RESTRT ;STOP THE SHOW--CLEAR THE WORLD
755
756 003522 004711 SENDAD: JSR PC,(R1)
757 003524 000240 NOP
758 003526 000240 NOP
759 003530 000240 NOP
760 003532 000240 NOP
761 003534 012737 010060 001214 RESTRT: MOV #CYCLE,RETURN
762 003542 000137 010060 JMP CYCLE
763 003546 000001 XCSR: 1
764 003550 006 002 .BYTE 6,2
765 003552 001404 DMCSR
766 003554 000001 XVEC: 1
767 003556 004 002 .BYTE 4,2
768 003560 001374 DMRVEC
769 003562 000001 XPASS: 1
770 003564 006 002 .BYTE 6,2
771 003566 001230 PASCNT
772 003570 000001 XERR: 1
773 003572 006 002 .BYTE 6,2
774 003574 001232 ERRCNT
775
776                                     ;SCOPE LOOP AND INTERATION HANDLER
777                                     ;-----
778
779 003576 004737 007606 .SCOPE: JSR PC,CKSWR ;CHECK FOR SOFT SWP
780 003602 010016 MOV RO,(SP) ;SAVE RO ON THE STACK

```

```

781 003604 032777 040000 175370      TTST:  BIT    #BIT14,2SWR      ;"LOOP ON THIS TEST"?
782 003612 001407                      BEQ     1$          ;BR IF NO. (IF LOCK SW01=1; THIS LOC =240)
783 003614 000437                      BR      3$          ;GOTO 3$ (IF LOCK SW01=1; THIS LOC =240)
784 003616 005737 003734              TST     DONE         ;WAS TKCSR DONE SET?
785 003622 001434                      BEQ     3$          ;BR IF NO (LOCKED ON TEST)
786 003624 005037 003734              CLR     DONE         ;YES, CLEAR FLAG
787 003630 000415                      BR      2$          ;GO TO NEXT TEST
788 003632 032777 004000 175342      1$:  BIT     #SW11,2SWR      ;DELETE ITERATION? (QUICK PASS)
789 003640 001011                      BNE     2$          ;BR IF YES
790 003642 105737 001327              TSTB    QV.FLG       ;HAVE PASSES BEEN COMPLETED?
791 003646 001406                      BEQ     2$          ;BR IF QUICK PASS.
792 003650 005237 001224              INC     LPCNT        ;UPDATE ITERATION COUNTER
793 003654 023737 001224 001222      CMP     LPCNT,ICOUNT  ;ARE ALL ITERATIONS DONE??
794 003662 101414                      BLOS    3$          ;BR IF NOT YET
795 003664 105037 001325      2$:  CLRB     ERRFLG       ;PREPARE FOR NEW TEST
796 003670 005037 001224              CLR     LPCNT        ;START ICOUNTER AT 0
797 003674 005037 001220              CLR     LOCK        ;
798 003700 012737 000020 001222      MOV     #20,ICOUNT    ;RESET ITERATIONS
799 003706 013737 001216 001214      MOV     NEXT,RETURN  ;GET NEXT TEST
800 003714 011600                      MOV     (SP),R0       ;POP R0 OFF OF THE STACK
801 003716 022626                      POP2SP    ;FAKE AN "RTI"
802 003720 013701 001404              MOV     DMCSR,R1      ;R1 CONTAINS BASE DMC ADDRESS
803 003724 000177 175264              JMP     @RETURN     ;GO DO THE TEST
804 003730 001407                      BRW:    1407
805 003732 000437                      BRX:    437
806 003734 000000                      DONE:   0
807
808                                     ;CHECK FOR FREEZE ON CURRENT DATA
809                                     ;-----
810
811 003736 004737 007606      .SCOPI: JSR     PC,CKSWR      ;CHECK FOR SOFT SWR
812 003742 032777 001000 175232      BIT     #SW09,2SWR      ;IS SW09=1(SET)?
813 003750 001405                      BEQ     1$          ;BR IF NOT SET.
814 003752 005737 001220              TST     LOCK        ;
815 003756 001402                      BEQ     1$          ;
816 003760 013716 001220              MOV     LOCK,(SP)    ;GOTO THE ADDRESS IN LOCK.
817 003764 000002      1$:  RTI                    ;GO BACK.
818
819                                     ;TELETYPE OUTPUT ROUTINE
820                                     ;-----
821
822 003766 010546      .TYPE:  MOV     R5,-(SP)      ;SAVE R5 ON THE STACK.
823 003770 017605      MOV     @2(SP),R5          ;GET ADDRESS OF MESSAGE.
824 003774 062766      ADD     #2,2(SP)          ;POP OVER ADDRESS.
825 004002 005737 010016      4$:  TST     SWFLG       ;SOFT SWR MESSAGE?
826 004006 001004                      BNE     1$          ;IF YES TYPE IT OUT REGARDLESS OF SW12
827 004010 032777 010000 175164      BIT     #SW12,2SWR      ;INHIBIT ALL PRINT OUT??
828 004016 001012                      BNE     3$          ;BR IF NO PRINT OUT WANTED (SW12=1)
829 004020 105715      1$:  TSTB    (R5)          ;IS NUMBER MINUS? (MSB=1(BIT7))
830 004022 100002                      BPL     2$          ;BR IF NUMBER IS PLUS
831 004024 104402 005672      TYPE     MCRLF        ;TYPE A CR/LF!
832 004030 105777 175154      2$:  TSTB    @TPCSR      ;TTY READY?
833 004034 100375                      BPL     2$          ;BR IF NO.
834 004036 112577 175150      MOVB     (R5)+,@TPDBR    ;PRINT CURRENT CHAR.
835 004042 001357                      BNE     4$          ;IF NOT ZERO KEEP PRINTING!
836 004044 012605      3$:  MOV     (SP)+,R5        ;END OF OUTPUT. RESTORE R5

```

```

837 004046 000002 RTI ;GO HOME
838 ;-----
839
840 004050 010346 .INSTR: MOV R3,-(SP) ;SAVE R3 ON STACK
841 004052 010446 MOV R4,-(SP) ;SAVE R4 ON STACK
842 004054 017637 000004 004072 MOV #4(SP),.MSG
843 004062 062766 000002 000004 ADD #2,4(SP)
844 004070 104402 .INST1: TYPE
845 004072 000000 .MSG: 0
846 004074 012704 007502 MOV #INBUF,R4
847 004100 012703 000007 MOV #7,R3
848 004104 105777 175074 1$: TSTB @TKCSR
849 004110 100375 BPL 1$
850 004112 117714 175070 MOVB @TKDBR,(R4)
851 004116 142714 000200 BICB #200,(R4)
852 004122 122427 000015 CMPB (R4),#15
853 004126 001417 BEQ INSTR2
854 004130 105777 175054 2$: TSTB @TPCSR
855 004134 100375 BPL 2$
856 004136 017777 175044 175046 MOV @TKDBR,@TPDBR
857 004144 005303 DEC R3
858 004146 001356 BNE 1$
859 004150 012604 MOV (SP)+,R4
860 004152 012603 MOV (SP)+,R3
861 004154 104402 005666 .INSTE: TYPE MQM
862 004160 010346 MOV R3,-(SP)
863 004162 010446 MOV R4,-(SP)
864 004164 000741 BR .INST1
865 004166 012604 INSTR2: MOV (SP)+,R4 ;RESTORE R4
866 004170 012603 MOV (SP)+,R3 ;RESTORE R3
867 004172 000002 RTI
868
869 ;CONVERT ASCII STRING TO OCTAL
870 ;-----
871
872 004174 010546 .PARAM: MOV R5,-(SP)
873 004176 010446 MOV R4,-(SP)
874 004200 016605 000004 MOV 4(SP),R5
875 004204 012537 004364 MOV (R5)+,LOLIM
876 004210 012537 004366 MOV (R5)+,HILIM
877 004214 012537 004370 MOV (R5)+,DEVADR
878 004220 112537 004372 MOV (R5)+,LOBITS
879 004224 112537 004373 MOV (R5)+,ADRCNT
880 004230 010566 000004 MOV R5,4(SP)
881 004234 005005 PARAM1: CLR R5
882 004236 012704 007502 MOV #INBUF,R4
883 004242 122714 000015 CMPB #15,(R4)
884 004246 001420 BEQ PARERR
885 004250 121427 000060 1$: CMPB (R4),#60
886 004254 002415 BLT PARERR
887 004256 121427 000067 CMPB (R4),#67
888 004262 003012 BGT PARERR
889 004264 142714 000060 BICB #60,(R4)
890 004270 152405 BISB (R4),R5
891 004272 122714 000015 CMPB #15,(R4)
892 004276 001406 BEQ LIMITS

```

```

893 004300 006305          ASL      R5
894 004302 006305          ASL      R5
895 004304 006305          ASL      R5
896 004306 000760          BR       1$
897 004310 104404          PARERR: INSTER
898 004312 000750          BR       PARAM1
899
900                          ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
901                          ;-----
902
903 004314 020537 004366    LIMITS: CMP      R5,HILIM
904 004320 101373          BHI      PARERR
905 004322 020537 004364    CMP      R5,LOLIM
906 004326 103770          BLO      PARERR
907 004330 133705 004372    BITB     LOBITS,R5
908 004334 001365          BNE      PARERR
909
910                          ;STORE NUMBER AT SPECIFIED ADDRESS
911
912 004336 013704 004370    1$:      MOV     DEVADR,R4
913 004342 010524          MOV     R5,(R4)+
914 004344 062705 000002    ADD      #2,R5
915 004350 105337 004373    DECB     ADRCNT
916 004354 001372          BNE      1$
917 004356 012604          MOV     (SP)+,R4
918 004360 012605          MOV     (SP)+,R5
919 004362 000002          RTI
920 004364 000000          LOLIM: 0
921 004366 000000          HILIM: 0
922 004370 000000          DEVADR: 0
923 004372 000000          LOBITS: 0
924                          ADRCNT=LOBITS+1
925
926                          ;SAVE PC OF TEST THAT FAILED AND R0-R5
927                          ;-----
928
929 004374 016637 000004 001276 .SAV05: MOV     4(SP),SAVPC      ;SAVE R7 (PC)
930
931                          ;SAVE R0-R5
932
933 004402 010537 001272    SV05:  MOV     R5,SAVR5      ;SAVE R5
934 004406 010437 001270    MOV     R4,SAVR4      ;SAVE R4
935 004412 010337 001266    MOV     R3,SAVR3      ;SAVE R3
936 004416 010237 001264    MOV     R2,SAVR2      ;SAVE R2
937 004422 010137 001262    MOV     R1,SAVR1      ;SAVE R1
938 004426 010037 001260    MOV     R0,SAVR0      ;SAVE R0
939 004432 000002          RTI                     ;LEAVE.
940
941                          ;RESTORE R0-R5
942
943 004434 013700 001260    .RES05: MOV     SAVR0,R0      ;RESTORE R0
944 004440 013701 001262    MOV     SAVR1,R1      ;RESTORE R1
945 004444 013702 001264    MOV     SAVR2,R2      ;RESTORE R2
946 004450 013703 001266    MOV     SAVR3,R3      ;RESTORE R3
947 004454 013704 001270    MOV     SAVR4,R4      ;RESTORE R4
948 004460 013705 001272    MOV     SAVR5,R5      ;RESTORE R5

```

```

949 004464 000002 RTI ;LEAVE
950
951 ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
952 ;-----
953
954 004466 104402 005672 .CONVR: TYPE MCRLF
955 004472 010046 .CNVRT: MOV R0,-(SP)
956 004474 010146 MOV R1,-(SP)
957 004476 010346 MOV R3,-(SP)
958 004500 010446 MOV R4,-(SP)
959 004502 010546 MOV R5,-(SP)
960 004504 017601 000012 MOV @12(SP),R1
961 004510 062766 000002 000012 ADD #2,12(SP)
962 004516 012137 004710 MOV (R1)+,WRDCNT
963 004522 112137 004712 1$: MOVB (R1)+,CHRCNT
964 004526 112137 004713 MOVB (R1)+,SPACNT
965 004532 013137 004714 MOV @ (R1)+,BINWRD
966 004536 122737 000003 004712 CMPB #3,CHRCNT
967 004544 001003 BNE 2$
968 004546 042737 177400 004714 BIC #177400,BINWRD
969 004554 013704 004714 2$: MOV BINWRD,R4
970 004560 113705 004712 MOVB CHRCNT,R5
971 004564 012700 001416 MOV #TEMP,R0
972 004570 010403 3$: MOV R4,R3
973 004572 042703 177770 BIC #177770,R3
974 004576 062703 000060 ADD #060,R3
975 004602 110320 MOVB R3,(R0)+
976 004604 000241 CLC
977 004606 006004 ROR R4
978 004610 000241 CLC
979 004612 006004 ROR R4
980 004614 000241 CLC
981 004616 006004 ROR R4
982 004620 005305 DEC R5
983 004622 001362 BNE 3$
984 004624 012703 007544 MOV #MDATA,R3
985 004630 114023 4$: MOVB -(R0),(R3)+
986 004632 105337 004712 DECB CHRCNT
987 004636 001374 BNE 4$
988 004640 105737 004713 TSTB SPACNT
989 004644 001405 BEQ 6$
990 004646 112723 5$: MOVB #040,(R3)+
991 004652 105337 004713 DECB SPACNT
992 004656 001373 BNE 5$
993 004660 105013 6$: CLRB (R3)
994 004662 104402 007544 TYPE ,MDATA
995 004666 005337 004710 DEC WRDCNT
996 004672 001313 BNE 1$
997 004674 012605 MOV (SP)+,R5
998 004676 012604 MOV (SP)+,R4
999 004700 012603 MOV (SP)+,R3
1000 004702 012601 MOV (SP)+,R1
1001 004704 012600 MOV (SP)+,R0
1002 004706 000002 RTI
1003 004710 000000 WRDCNT: 0
1004 004712 000000 CHRCNT: 0

```

1005		004713			SPACNT=CHRCNT+1
1006	004714	000000			BINWRD: 0
1007					
1008					
1009					: TRAP DISPATCH SERVICE
1010					: ARGUMENT OF TRAP IS EXTRACTED
1011					: AND USED AS OFFSET TO OBTAIN POINTER
1012					: TO SELECTED SUBROUTINE
1013					
1014	004716	011646			.TRPSR: MOV (SP), -(SP) ; GET PC OF RETURN
1015	004720	162716	000002		SUB #2, (SP) ; =PC OF TRAP
1016	004724	017616	000000		MOV @ (SP), (SP) ; GET TRP
1017	004730	006316			TRPOK: ASL (SP) ; MULTIPLY TRAP ARG BY 2
1018	004732	042716	177001		BIC #177001, (SP) ; CLEAR UNWANTED BITS
1019	004736	062716	001330		ADD #.TRPTAB, (SP) ; POINTER TO SUBROUTINE ADDRESS
1020	004742	017616	000000		MOV @ (SP), (SP) ; SUBROUTINE ADDRESS
1021	004746	000136			JMP @ (SP) ; GO TO SUBROUTINE
1022					
1023					: ERROR HANDLER
1024					: -----
1025					
1026	004750	004737	007606		.HLT: JSR PC, CKSWR ; CHECK FOR SOFT SWR
1027	004754	032777	010000	174220	BIT #SW12, @SWR ; BELL ON ERROR?
1028	004762	001406			BEQ XBK ; BR IF NO BELL
1029	004764	105777	174220		TSTB @TPCSR ; TTY READY.
1030	004770	100003			BPL XBK ; DON'T WAIT IF TTY NOT READY.
1031	004772	112777	000207	174212	MOVB #207, @TPDBR ; PUSH A BELL AT THE TTY.
1032	005000	032777	020000	174174	XBK: BIT #SW13, @SWR ; DELETE ERROR PRINT OUT?
1033	005006	001105			BNE HALTS ; BR IF NO PRINT OUT WANTED.
1034	005010	021637	001234		CMP (SP), LSTERR ; WAS THIS ERROR FOUND LAST TIME?
1035	005014	001404			BEQ 1\$; BR IF YES
1036	005016	011637	001234		MOV (SP), LSTERR ; RECORD BEING HERE
1037	005022	105037	001325		CLRB ERRFLG ; PREPARE HEADER
1038	005026	104406			1\$: SAVOS ; SAVE ALL PROC REGISTERS
1039	005030	011605			MOV (SP), R5 ; GET THE PC OF ERROR
1040	005032	162705	000002		SUB #2, R5 ; GET ADDRESS OF TRAP CALL
1041	005036	011504			MOV (R5), R4 ; GET HLT INSTRUCTION
1042	005040	006304			ASL R4 ; MULT BY TWO
1043	005042	061504			ADD (R5), R4 ; DOUBLE IT
1044	005044	006304			ASL R4 ; MULT AGAIN
1045	005046	042704	177001		BIC #177001, R4 ; CLEAR JUNK
1046	005052	062704	031660		ADD #.ERRTAB, R4 ; GET POINTER
1047	005056	012437	005172		MOV (R4)+, ERRMSG ; GET ERROR MESSAGE
1048	005062	012437	005204		MOV (R4)+, DATAHD ; GET DATA HEADREP
1049	005066	011437	005216		MOV (R4), DATABP ; GET DATA TABLE
1050	005072	105737	001325		TSTB ERRFLG ; TYPE HEADREER
1051	005076	001403			BEQ TYPMSG ; BR IF YES
1052	005100	005737	005216		TST DATABP ; DOES DATA TABLE EXIST?
1053	005104	001040			BNE TYPDAT ; BR IF YES.
1054	005106	104402	005672		TYPMSG: TYPE ,MCRLF
1055	005112	104402	005672		TYPE ,MCRLF
1056	005116	005737	001220		TST LOCK
1057	005122	001402			BEQ 1\$
1058	005124	104402	006142		TYPE ,MASTEK
1059	005130	104402	006130		1\$: TYPE ,MTSTN
1060	005134	104411	005330		CNVR+ ,XTSTN ; SHOW IT

1061	005140	104402	006217		TYPE	,MERRPC	:TYPE PC.
1062	005144	104411	005322		CONVRT	,ERTABO	:SHOW IT
1063	005150	104402	005677		TYPE	,MCRLF	:GIVE A CR/LF
1064	005154	112737	177777	001325	MOVB	,-1,ERRFLG	:NO MORE HEADER UNLESS NO DATA TABLE.
1065	005162	005737	005172		TST	ERRMSG	:IS THERE AN ERROR MESSAGE?
1066	005166	001402			BEQ	WRKO.FM	:BR IF NO.
1067	005170	104402			TYPE		:TYPE
1068	005172	000000			ERRMSG: 0		:ERROR MESSAGE
1069	005174				WRKO.FM:		
1070	005174	005737	005204		TST	DATAHD	:DATA HEADER?
1071	005200	001402			BEQ	TYPOAT	:BR IF NO
1072	005202	104402			TYPE		:TYPE
1073	005204	000000			DATAHD: 0		:DATA HEADER
1074	005206	005737	005216		TYPDAT: TST	DATABP	:DATA TABLE?
1075	005212	001402			BEQ	RESREG	:BR IF NO.
1076	005214	104410			CONVRT		:SHOW
1077	005216	000000			DATABP: 0		:DATA TABLE
1078	005220	104407			RESREG: RESOS		:RESTORE PROC REGISTERS
1079	005222	022737	003522	000042	HALTS: CMP	,SENDAD,2*42	:IF ACT-11 AUTOMATIC MODE, HALT!!
1080	005230	001403			BEQ	1\$	
1081	005232	005777	173744		TST	2\$WR	:HALT ON ERROR?
1082	005236	100005			BPL	EXITER	:BR IF NO HALT ON ERROR
1083	005240	010046			1\$: PUSHRO		:SAVE R0
1084	005242	016600	000002		MOV	2(SP),R0	:SHOW ERROR PC IN DATA LIGHTS
1085	005246	000000			HALT		:HALT
1086	005250	012600			POPPO		:GET R0
1087	005252	005237	001232		EXITER: INC	ERRCNT	:UPDATE ERROR COUNT
1088	005256	032777	000400	173716	BIT	,SW08,2\$WR	:GOTO TOP OF TEST?
1089	005264	001007			BNE	1\$	:BR IF YES
1090	005266	032777	002000	173706	BIT	,SW10,2\$WR	:GOTO NEXT TEST?
1091	005274	001411			BEQ	2\$	:BR IF NO
1092	005276	013737	001216	001214	MOV	NEXT,RETURN	:SET FOR NEXT TEST
1093	005304	012706	001200		1\$: MOV	,STACK,SP	:RESET SP
1094	005310	013701	001404		MOV	DMCSR,R1	:SET UP R1
1095	005314	000177	173674		JMP	2\$RETURN	:GOTO SPECIFIED TEST
1096	005320	000002			2\$: RTI		:RETURN
1097	005322	000001			ERTABO: 1		
1098	005324	006	002		.BYTE	6,2	
1099	005326	001276			SAVPC		
1100	005330	000001			XTSTN: 1		
1101	005332	003	002		.BYTE	3,2	
1102	005334	001226			TSTNO		
1103					:ENTER HERE ON POWER FAILURE		
1104					:-----		
1105							
1106							
1107	005336				.PFAIL:		
1108	005336	012737	005350	000024	MOV	,RESTART,24	:SET UP FOR POWER UP TRAP
1109	005344	000000			HALT		:HALT ON POWER DOWN NORMAL
1110	005346	000777			BR	.	
1111							
1112					:PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED		
1113							
1114	005350				RESTAR:		
1115	005350	012737	005336	000024	MOV	,PFAIL,24	:SET UP FOR POWER FAILURE
1116	005356	012706	001200		MOV	,STACK,SP	:RESET THE STACK POINTER

```

1117 005362 013701 001404      MOV      DMCSR,R1      ;RESTORE R1
1118 005366 005037 001416      CLR      TEMP        ;READY FOR TIMMER
1119 005372 005237 001416      INC      TEMP        ;PLUS ONE TO THE TIMER!
1120 005376 001375              BNE      .-4           ;BR IF MORE TO GO
1121 005400 104402 005675      TYPE     ,MPFAIL      ;TYPE THE MESSAGE
1122 005404 104411 005430      CNVRT    ,PFTAB      ;TELL WHAT TEST TO RETURN TO.
1123 005410 105037 001325      CLRB     ERRFLG      ;START CLEAN
1124 005414 005037 001234      CLR      LSTERR      ;*****
1125 005420 005011              CLR      (R1)          ;CLEAR MAINT BITS
1126 005422 104412              MSTCLR   ;START CLEAN UP OF DEVICE
1127 005424 000177 173564      JMP      @RETURN      ;START DOING THAT TEST AGAIN.
1128 005430 000001              PFTAB: 1
1129 005432 003      002      .BYTE  3,2
1130 005434 001226              TSTNO
1131
1132 005436              .DELAY:
1133 005436 012777 000020 173746      MOV      #20,@DMP04
1134 005444 104414              ROMCLK   121111      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1135 005446 121111              ;POKE CLOCK DELAY BIT
1136 005450              1$:
1137 005450 104414              ROMCLK   121224      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1138 005452 121224              ;PORT4+IBUS*11
1139 005454 032777 000020 173730      BIT      #BIT4,@DMP04      ;IS CLOCK BIT SET?
1140 005462 001772              BEQ      1$          ;BR IF NO
1141 005464 000002              RTI
1142
1143 005466              .MSTCLR:
1144 005466 152777 000100 173712      BISB     #BIT6,@DMCSRH      ;SET MASTER CLEAR
1145 005474 142777 000300 173704      BICB     #BIT6!BIT7,@DMCSRH ;CLEAR MASTER CLEAR AND RUN
1146 005502 000002              RTI              ;RETURN
1147
1148 005504              .ROMCLK:
1149 005504 152777 000002 173674      BISB     #BIT1,@DMCSRH      ;SET ROMI
1150 005512 013677 173676      MOV      @((SP)+,@DMP06      ;LOAD INSTRUCTION IN SEL6
1151 005516 062746 000002              ADD     #2,-(SP)          ;ADJUST STACK
1152 005522 032777 000100 173452      BIT      #SW06,@SWR      ;HALT IF SW06 =1
1153 005530 001401              BEQ      1$          ;BR IF SW06 =0
1154 005532 000000              HALT          ;HALT BEFORE CLOCKING INSTRUCTION
1155 005534 152777 000003 173644      1$:  BISB     #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
1156 005542 142777 000007 173636      BICB     #BIT2!BIT1!BIT0,@DMCSRH ;CLEAR ROM0, ROMI. STEP
1157 005550 000002              RTI
1158
1159 005552              .DATACLK:
1160 005552 013637 001416      MOV      @((SP)+,TEMP      ;PUT TICK COUNT IN TEMP
1161 005556 062746 000002              ADD     #2,-(SP)          ;ADJUST STACK
1162 005562 152777 000020 173616      1$:  BISB     #BIT4,@DMCSRH      ;SET STEP LU
1163 005570 027777 173610 173606      CMP      @DMCSR,@DMCSR      ;WASTE TIME
1164 005576 142777 000020 173602      BICB     #BIT4,@DMCSRH      ;CLEAR STEP LU
1165 005604 005337 001416      DEC      TEMP        ;DEC TICK COUNT
1166 005610 001364              BNE      1$          ;BR IF NOT DONE
1167 005612 000002              RTI              ;RETURN
1168 005614 000001              3$:  .BLKW 1
1169
1170 005616              .TIMER:
1171 005616 013637 001416      MOV      @((SP)+,TEMP      ;MOVE COUNT TO TEMP
1172 005622 062746 000002              ADD     #2,-(SP)          ;ADJUST STACK

```

```

1173 005626          15:          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1174 005626 104414    021364    ;PORT4+IBUS* REG11
1175 005630 021364    ;IS PGM CLOCK BIT CLEAR?
1176 005632 032777 000002 173552 BIT          #2,2DMP04
1177 005640 001772    BEQ          15          ;BR IF YES
1178 005642          25:          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1179 005642 104414    021364    ;PORT4+IBUS* REG11
1180 005644 021364    ;IS PGM CLOCK BIT SET?
1181 005646 032777 000002 173536 BIT          #2,2DMP04
1182 005654 001372    BNE          25          ;BR IF YES
1183 005656 005337 001416    DEC          TEMP    ;DEC COUNT
1184 005662 001361    BNE          15          ;BR IF NOT DONE
1185 005664 000002    RTI          ;RETURN
1186
1.87 005666 020040 000077    MQM:          .ASCIZ  /  ? /
(2) 005672 005015      000    MCRLF:         .ASCIZ  <15><12>
(2) 005675      377 053520 020122 MPFAIL:        .ASCIZ  <377>/PWR FAILED. RESTART AT TEST /
(2) 005733      377 047105 020104 MEFASS:        .ASCIZ  <377>/END PASS DZDME /
(2) 005755      377 000122    MR:          .ASCIZ  <377>/R/
(2) 005760 047377 020117 042504 MERR2:         .ASCIZ  <377>/NO DEVICES PRESENT./
(2) 006005      377 047111 052523 MERR3:         .ASCIZ  <377>/INSUFFICIENT DATA!/
(2) 006031      377 042524 052123 MTSTPC:         .ASCIZ  <377>/TEST PC-/
(2) 006043      377 047514 045503 MLOCK:         .ASCIZ  <377>/LOCK ON SELECTED TEST/
(2) 006072 051503 035122 000040 MCSRX:         .ASCIZ  /CSR: /
(2) 006100 042526 035103 000040 MVECX:         .ASCIZ  /VEC: /
(2) 006106 040520 051523 051505 MPASSX:        .ASCIZ  /PASSES: /
(2) 006117      105 051122 051117 MERRX:         .ASCIZ  /ERRORS: /
(2) 006130 042524 052123 047040 MTSTN:         .ASCIZ  /TEST NO: /
(2) 006142 000052    MASTEK:         .ASCIZ  /*/
(2) 006144 051777 052105 051440 MNEW:          .ASCIZ  <377>/SET SWITCH REG TO DMC11'S DESIRED ACTIVE./
(2) 006217      120 035103 000040 MERRPC:        .ASCIZ  /PC: /
(2) 006224 020212 020040 020040 XHEAD:         .ASCIZ  <212>/
(2) 006263      377 020040 020040          .ASCIZ  <377>/
(2) 006322 020212 050040 020103          .ASCIZ  <212>/ PC      CSR      STAT1      STAT2      STAT3.
(2) 006374 026777 026455 026455          .ASCIZ  <377>/-----
(2) 006450 044377 053517 046440 NUM:          .ASCIZ  <377>/HOW MANY DMC11'S TO BE TESTED?/
(2) 006510 041777 051123 040440 CSR:          .ASCIZ  <377>/CSR ADDRESS?/
(2) 006526 053377 041505 047524 VEC:          .ASCIZ  <377>/VECTOR ADDRESS?/
(2) 006547      377 051102 050040 PRIO:         .ASCIZ  <377>/BR PRIORITY LEVEL? (4,5,6,7)?/
(2) 006606 044777 020106 046504 CRAM:         .ASCIZ  <377>/IF DMC HAS CRAM (M8204) TYPE "Y", IF CROM (M8200) TYPE "N"
(2) 006704 053777 044510 044103 MODU:         .ASCIZ  <377>/WHICH LINE UNIT? IF NONE TYPE "N", i: M8201 TYPE "I", IF "
(2) 007016 051777 044527 041524 LINE:         .ASCIZ  <377>/SWITCH PAC#1 (DDCMP LINE #)?/
(2) 007054 051777 044527 041524 BM:          .ASCIZ  <377>/SWITCH PAC#2 (BM873 BOOT ADD)?/
(2) 007114 044777 020123 044124 CONN:         .ASCIZ  <377>/IS THE LOOP BACK CONNECTOR ON?/
(2) 007154 047377 020117 042504 NOACT:        .ASCIZ  <377>/NO DEVICES ARE SELECTED/
(2) 007205      377 051412 051127 SWMES:         .ASCIZ  <377><12>/SWR= /
(2) 007215      116 053505 020077 SWMES1:        .ASCIZ  /NEW? /
(2) 007223      377 042377 041515 CONERR:       .ASCIZ  <377><377>/DMC11 FOUND AT NON-STANDARD ADDRESS PC: .
(2) 007277      377 054105 042520 CNERR:        .ASCIZ  <377>/EXPECTED FOUND/
(2) 007320 024040 046504 024503 DMCM:         .ASCIZ  / (DMC) .
(2) 007330 024040 046513 024503 KMCM:         .ASCIZ  / (KMC) .
(2) 007340 042377 041515 030461 SPEED:        .ASCIZ  <377>/DMC11-AR(REMOTE,LOW SPEED) OR DMC11-AL(LOCAL,HIGH SPEED) *
(2)          .EVEN
(2) 007454 000005    XSTATQ: 5
1.88 007456      006      003      .BYTE      6.3
1.89 007460 001246    TEMP!

```

1190	007462	006	003		.BYTE	6.3	
1191	007464	001250			TEMP2		
1192	007466	006	003		.BYTE	6.3	
1193	007470	001252			TEMP3		
1194	007472	006	003		.BYTE	6.3	
1195	007474	001254			TEMP4		
1196	007476	006	002		.BYTE	6.2	
1197	007500	001256			TEMP5		
1198					.EVEN		
1199							
1200					:BUFFERS FOR INPUT-OUTPUT		
1201							
1202	007502	000000			INBUF:	0	
1203		007544			.=.+40		
1204	007544	000000			MDATA:	0	
1205		007606			.=.+40		
1206							
1207							
1208					:ROUTINE USED TO CHANGE SOFTWARE SWITCH		
1209					:REGISTER USING THE CONSOLE TERMINAL		
1210					-----		
1211							
1212	007606	022737	000176	001202	CKSWR:	CMP	#SWREG,SWR
1213	007614	001077				BNE	CKSWR5
1214	007616	105777	171362			TSTB	#TKCSR
1215	007622	100003				BPL	2\$
1216	007624	012737	177777	003734		MOV	#-1,DONE
1217	007632	022777	000007	171346	2\$:	CMP	#7,#TKDBR
1218	007640	001404				BEQ	1\$
1219	007642	022777	000207	171336		CMP	#207,#TKDBR
1220	007650	001061				BNE	CKSWR5
1221	007652	010246			1\$:	MOV	R2,-(SP)
1222	007654	010346				MOV	R3,-(SP)
1223	007656	010446				MOV	R4,-(SP)
1224	007660	012737	177777	010016		MOV	#-1,SWFLG
1225	007666	005002			CKSWR1:	CLR	R2
1226	007670	012704	177777			MOV	#-1,R4
1227	007674	104402	007205			TYPE	,SWMES
1228	007700	104411			CKSWR2:	CNVRT	
1229	007702	010052				SOFTSW	
1230	007704	104402	007215		CKSWR3:	TYPE	SWMES1
1231	007710	004737	010020		CKSWR4:	JSR	PC,INCHAR
1232	007714	022703	000015			CMP	#15,R3
1233	007720	001424				BEQ	5\$
1234	007722	022703	000012			CMP	#12,R3
1235	007726	001416				BEQ	4\$
1236	007730	022703	000025			CMP	#25,R3
1237	007734	001754				BEQ	CKSWR1
1238	007736	022703	000007			CMP	#7,R3
1239	007742	001762				BEQ	CKSWR4
1240	007744	005004				CLR	R4
1241	007746	042703	177770			BIC	#177770,R3
1242	007752	006302				ASL	R2
1243	007754	006302				ASL	R2
1244	007756	006302				ASL	R2
1245	007760	050302				BIS	R3,R2

:IS THE SOFT SWR BEING USED?
 :BR IF NO
 :IS DONE SET?
 :GO ON IF NOT SET
 :IF DONE SET SET FLAG
 :WAS CTRL G TYPED? 7 BIT ASCII:
 :BR IF YES
 :WAS CTRL G TYPED? 8 BIT ASCII:
 :BR IF NO
 :STORE R2
 :STORE R3
 :STORE R4
 :SET SOFT TYPE OUT FLAG
 :CLEAR NEW SWR CONTENTS
 :SET FLAG TO ALL ONES
 :TYPE "SWR=" "
 :TYPE OUT PRESENT CONTENTS
 :OF SOFT SWITCH REGISTER
 :TYPE "NEW" "
 :GET RESPONSE
 :WAS IT A CR?
 :BR IF YES
 :WAS IT A LF?
 :BR IF YES
 :WAS IT CTRL U?
 :BR IF YES(START OVER)
 :IF CNTL G GET NEXT CHAR
 :IT MUST BE A DIGIT SO CLR FLAG
 :ONLY 0-7 ARE LEGAL SO MASK OFF BITS
 :SHIFT R2 3 TIMES
 :ADD LAST DIGIT

1246	007762	000752			BR	CKSWR4	:GET NEXT CHARACTER
1247	007764	012766	002002	000006	4\$: MOV	\$.START,6(SP)	:LF WAS TYPED SO GO TO START
1248	007772	005704			5\$: TST	R4	:IS FLAG CLEAR?
1249	007774	001002				6\$	:IF NOT DON'T CHANGE SOFT SWP
1250	007776	010277	171200		MOV	R2,2SWR	:IF YES THEN WRITE NEW CONTENTS TO SCF* SWP
1251	010002	005037	010016		6\$: CLR	SWFLG	:CLEAR TYPEOUT FLAG
1252	010006	012604			MOV	(SP)+,R4	:RESTORE R4
1253	010010	012603			MOV	(SP)+,R3	:RESTORE R3
1254	010012	012602			MOV	(SP)+,R2	:RESTORE R2
1255	010014	000207			CKSWR5: RTS	PC	:RETURN
1256							
1257	010016	000000			SWFLG: 0		
1258							
1259	010020	105777	171160		INCHAR: TSTB	2TKCSR	
1260	010024	100375			BPL	.-4	
1261	010026	017703	171154		MOV	2TKDBR,R3	
1262	010032	105777	171152		TSTB	2TPCSR	
1263	010036	100375			BPL	.-4	
1264	010040	010377	171146		MOV	R3,2TPDBR	
1265	010044	042703	000200		BIC	2BIT7,R3	
1266	010050	000207			RTS	PC	
1267							
1268	010052	000001			SOFTSW: 1		
1269	010054	006	002		.BYTE	6.2	
1270	010056	000176			SWREG		

```

1271
1272
1273
1274
1275
1276
1277
1278
1279
1280 010060 005737 001306
1281 010064 001004
1282 010066 104402 007154
1283 010072 000000
1284 010074 000776
1285 010076 000241
1286 010100 006137 001316
1287 010104 005537 001316
1288 010110 062737 000004 001322
1289 010116 062737 000010 001320
1290 010124 022737 001700 001320
1291 010132 001006
1292 010134 012737 001500 001320
1293 010142 012737 001702 001322
1294 010150 033737 001316 001306
1295 010156 001747
1296 010160 013700 001320
1297 010164 013702 001322
1298 010170 012037 001404
1299 010174 011037 001374
1300 010200 042737 177000 001374
1301 010206 012037 001366
1302 010212 012037 001370
1303 010216 012037 001372
1304 010222 012237 001230
1305 010226 012237 001232
1306 010232 012700 000002
1307 010236 013737 001404 001406
1308 010244 005237 001406
1309 010250 013737 001406 001410
1310 010256 005237 001410
1311 010262 013737 001410 001412
1312 010270 060037 001412
1313 010274 013737 001412 001414
1314 010302 060037 001414
1315
1316 010306 013737 001374 001376
1317 010314 060037 001376
1318 010320 013737 001376 001400
1319 010326 060037 001400
1320 010332 013737 001400 001402
1321 010340 060037 001402
1322
1323 010344 032737 000002 001236
1324 010352 001450
1325 010354
1326 010354 005737 000042

```

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

```

CYCLE: TST DMACTV ;ARE ANY DMC11'S TO BE TESTED?
        BNE 1$ ;BR IF OK.
        TYPE ,NOACT ;NO DMC11'S SELECTED!!
        HALT ;STOP THE SHOW.
        BR -2 ;DISQUALIFY CONT. SW.
1$: CLC ;CLEAR PROC. CARRY BIT.
    ROL RUN ;UPDATE POINTER
    ADC RUN ;CATCH CARRY FROM RUN
    ADD #4,MILK ;UPDATE POINTER
    ADD #10,CREAM ;UPDATE ADDRESS POINTER.
    CMP #DM.MAP+200,CREAM
    BNE 2$ ;KEEP GOING; NOT ALL TESTED FOR.
    MOV #DM.MAP,CREAM ;RESET ADDRESS POINTER.
    MOV #CNT.MAP,MILK ;RESET PASS COUNT POINTER
2$: BIT RUN,DMACTV ;IS THIS ONE ACTIVE?
    BEQ 1$ ;BR IF NO
    MOV CREAM,RO ;GET ADDRESS POINTER
    MOV MILK,R2 ;GET PASS COUNT POINTER
    MOV (RO)+,DMCSR ;LOAD SYSTEM CTRL. REG
    MOV (RO),DMRVEC ;LOAD VECTOR
    BIC #177000,DMRVEC ;CLEAR UNWANTED BITS
    MOV (RO)+,STAT1 ;LOAD STAT1
    MOV (RO)+,STAT2 ;LOAD STAT2
    MOV (RO)+,STAT3 ;LOAD STAT3
    MOV (R2)+,PASCNT ;LOAD PASS COUNT
    MOV (R2)+,ERRCNT ;LOAD ERROR COUNT
    MOV #2,RO ;SAVE CORE THIS WAY!
    MOV DMCSR,DMCSRH
    INC DMCSRH
    MOV DMCSRH,DMCTL
    INC DMCTL
    MOV DMCTL,DMP04
    ADD RO,DMP04
    MOV DMP04,DMP06
    ADD RO,DMP06
    MOV DMRVEC,DMRLVL ;PTY LVL
    ADD RO,DMRLVL
    MOV DMRLVL,DMTVEC ;TX VEC
    ADD RO,DMTVEC
    MOV DMTVEC,DMTLVL ;TX LVL
    ADD RO,DMTLVL
4$: BIT #SW01,STRTSW ;IS TEST NO. SELECTED
    BEQ 7$ ;BR IF NO
    TST #42 ;RUNNING IN AUTO MODE?

```

1327	010360	001045			BNE	7\$	:BR IF YES
1328	010362	104402	005672		TYPE	,MCRLF	
1329	010366	104403			INSTR		:GET TEST NO.
1330	010370	006130			MTSTN		
1331	010372	104405			PARAM		
1332	010374	000001			1		
1333	010376	001000			1000		
1334	010400	001226			TSTNO		
1335	010402	000			0		
1336	010403	001			.BYTE		
1337	010404	012700	012320		.BYTE		
1338	010410	022710			MOV	#TST1,RO	
1339	010412	012737			5\$: CMP	(PC)+,(RO)	:CMP FIRST WORD TO 12737
1340	010414	001020			MOV	(PC)+,2(PC)+	
1341	010416	023760	001226	000002	BNE	6\$	:BR IF NOT SAME
1342	010424	001014			CMP	TSTNO,2(RO)	:DOES TSTNO MATCH?
1343	010426	022760	001226	000004	BNE	6\$	:BR IF NO
1344	010434	001010			CMP	#TSTNO,4(RO)	:IS LAST WORD OK?
1345	010436	010037	001214		BNE	6\$	:BR IF NO
1346	010442	104402	005755		MOV	RO,RETURN	:IT IS A LEGAL TEST SO DO IT
1347	010446	042737	000002	001236	TYPE	,MR	
1348	010454	000412			BIC	#SW01,STRTSW	
1349	010456	005720			BR	8\$	
1350	010460	020027	026010		6\$: TST	(RO)+	:POP RO
1351	010464	001351			CMP	RO,#TLAST+10	:AT END YET?
1352	010466	104402	005666		BNE	5\$	:BR IF NO
1353	010472	000730			TYPE	,MQM	:YES ILLEGAL TEST NO.
1354					BR	4\$	:TRY AGAIN
1355	010474	012737	012320	001214	7\$: MOV	#TST1,RETURN	:PREPARE RETURN ADDRESS
1356	010502	013701	001404		8\$: MOV	DMCSR,R1	:R1 = BASE DMC11 ADDRESS
1357	010506	000177	170502		JMP	2RETURN	:GO START TESTING.
1358							
1359							
1360							
1361							
1362							
1363							
1364							
1365							
1366							
1367							
1368	010512						
1369	010512	000005			AUTO.SIZE:		
1370	010514	012702	001500		RESET		:INSURE A BUS INIT.
1371	010520	005022			CSRMAP: MOV	#DM.MAP,R2	:LOAD MAP POINTER.
1372	010522	022702	001700		1\$: CLR	(R2)+	:ZERO ENTIRE MAP
1373	010526	001374			CMP	#DM.END,R2	:ALL DONE?
1374	010530	005037	001310		BNE	1\$	:BR IF NO
1375	010534	012702	001500		CLR	DMNUM	:SET OCTAL NUMBER OF DMC11'S TO 0
1376	010540	005037	001306		MOV	#DM.MAP,R2	:R2 POINTS TO DMC MAP
1377	010544	032737	000001	001236	CLR	DMACTV	:CLEAR ACTIVE
1378	010552	001002			BIT	#SW00,STRTSW	:QUESTIONS?
1379	010554	000137	011252		BNE	+6	:BR IF YES
1380	010560	012737	000001	001256	JMP	7\$	:IF NO SKIP QUESTIONS
1381	010566	104403			MOV	#1,TEMPS	:START WITH 1
1382	010570	006450			INSTR		
					NUM		

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

1383	010572	104405			PARAM		
1384	010574	000001			16.		
1385	010576	000020			TEMP3		
1386	010600	001252			.BYTE	0	
1387	010602	000			.BYTE	1	
1388	010603	001			MOV	TEMP3, DMNUM	; DMNUM = HOW MANY
1389	010604	013737	001252	001310	TYPE	, MCRLF	
1390	010612	104402	005672		CONVRT		; TYPE WHICH DMC IS BEING DONE
1391	010616	104410			WHICH		; TEMPS IS WHICH DMC
1392	010620	012002			INC	TEMPS	
1393	010622	005237	001256		INSTR		
1394	010626	104403			CSR		
1395	010630	006510			PARAM		
1396	010632	104405			160000		
1397	010634	160000			164000		
1398	010636	164000			TEMP4		
1399	010640	001254			.BYTE	0	
1400	010642	000			.BYTE	1	
1401	010643	001			MOV	TEMP4, (R2)+	; STORE CSR IN MAP
1402	010644	013722	001254		INSTR		
1403	010650	104403			VEC		
1404	010652	006526			PARAM		
1405	010654	104405			0		
1406	010656	000000			776		
1407	010660	000776			TEMP4		
1408	010662	001254			.BYTE	0	
1409	010664	000			.BYTE	1	
1410	010665	001			MOV	TEMP4, (R2)	; STORE VECTOR IN MAP
1411	010666	013712	001254		TYPE		
1412	010672	104402			PRI0		; ASK WHAT BR LEVEL
1413	010674	006547			JSR	PC, INTTY	; GET RESPONSE
1414	010676	004737	012266		CMP	#24, R3	
1415	010702	022703	000024		BHI	50\$	; BR IF LESS THAN 4
1416	010706	101014			CMP	#27, R3	
1417	010710	022703	000027		BLO	50\$	; BR IF GREATER THAN 7
1418	010714	103411			MOV	#11, R4	; R4 = NUMBER OF SHIFTS
1419	010716	012704	000011		ASL	R3	; SHIFT R3 LEFT
1420	010722	006303			DEC	R4	; DEC SHIFT COUNT
1421	010724	005304			BNE	.-4	; BR IF NOT DONE
1422	010726	001375			BIC	#170777, R3	; BIC UNWANTED BITS
1423	010730	042703	170777		BIS	R3, (R2)	; PUT BR LEVEL IN STATUS MAP
1424	010734	050312			BR	8\$	; CONTINUE
1425	010736	000403			TYPE		
1426	010740	104402			MQM		; RESPONSE IS OUT OF LIMITS
1427	010742	005666			BR	10\$	; TRY AGAIN
1428	010744	000752			TYPE		
1429	010746	104402			CRAM		; DOES DMC HAVE CRAM?
1430	010750	006606			JSR	PC, INTTY	; GET REPLY
1431	010752	004737	012266		CMP	#131, R3	
1432	010756	022703	000131		BEQ	9\$	; YES
1433	010762	001427			CMP	#116, R3	; NO
1434	010764	022703	000116		BEQ	40\$	; NOT A Y OR N
1435	010770	001403			TYPE		
1436	010772	104402			MQM		; TYPE "?"
1437	010774	005666			BR	8\$	; ASK AGAIN
1438	010776	000763					

1439	011000	104402		40\$:	TYPE		
1440	011002	007340			SPEED		;DMC11-AR OR DMC11-AL?
1441	011004	004737	012266		JSR	PC,INTTY	;GET RESPONSE
1442	011010	022703	000122		CMP	#122,R3	;IS IT R
1443	011014	001414			BEQ	16\$	;BR IF REMOTE
1444	011016	022703	000114		CMP	#114,R3	;IS IT L
1445	011022	001403			BEQ	41\$	;BR IF LOCAL
1446	011024	104402			TYPE		
1447	011026	005666			MQM		
1448	011030	000763			BR	40\$	;TRY AGAIN
1449	011032	052762	000002 000004	41\$:	BIS	#BIT1,4(R2)	;SET BIT1 IN STAT3
1450	011040	000402			BR	16\$	;CONTINUE
1451	011042	052712	100000	9\$:	BIS	#BIT15,(R2)	;SET BIT 15 IF CRAM
1452	011046	104402		16\$:	TYPE		
1453	011050	006704			MODU		;ASK WHICH LINE UNIT
1454	011052	004737	012266		JSR	PC,INTTY	;GET REPLY
1455	011056	022703	000021		CMP	#21,R3	; "1"
1456	011062	001417			BEQ	30\$	
1457	011064	022703	000022		CMP	#22,R3	; "2"
1458	011070	001412			BEQ	31\$	
1459	011072	022703	000116		CMP	#116,R3	; "N"
1460	011076	001403			BEQ	32\$	
1461	011100	104402			TYPE		
1462	011102	005666			MQM		;IF NOT A 1,2 OR N TYPE "?"
1463	011104	000760			BR	16\$	;TRY AGAIN
1464	011106	052722	010000	32\$:	BIS	#BIT12,(R2)+	;SET BIT 12 IN STAT2 IF NO LU
1465	011112	022222			CMP	(R2)+,(R2)+	;POP OVER STAT2 AND STAT3
1466	011114	000447			BR	33\$	
1467	011116	052712	020000	31\$:	BIS	#BIT13,(R2)	;SET BIT 13 IN STAT2 IF M8202
1468	011122	104402		30\$:	TYPE		
1469	011124	007114			CONN		;ASK IF LOOP-BACK IS ON
1470	011126	004737	012266		JSR	PC,INTTY	;GET REPLY
1471	011132	022703	000131		CMP	#131,R3	;Y
1472	011136	001406			BEQ	17\$	
1473	011140	022703	000116		CMP	#116,R3	;N
1474	011144	001406			BEQ	18\$	
1475	011146	104402			TYPE		
1476	011150	005666			MQM		;IF NOT Y OR N TYPE ""
1477	011152	000763			BR	30\$	;TRY AGAIN
1478	011154	052722	040000	17\$:	BIS	#BIT14,(R2)+	;TURNAROUND IS CONNECTED
1479	011160	000402			BR	19\$	
1480	011162	042722	040000	18\$:	BIC	#BIT14,(R2)+	;NO TURNAROUND
1481	011166			19\$:			
1482	011166	104403			INSTR		
1483	011170	007016			LINE		
1484	011172	104405			PARAM		
1485	011174	000000			0		
1486	011176	000377			377		
1487	011200	001254			TEMP4		
1488	011202	000			.BYTE	0	
1489	011203	001			.BYTE	1	
1490	011204	113722	001254		MOVB	TEMP4,(R2)+	;STORE SWITCH PAC IN MAP
1491	011210	104403			INSTR		
1492	011212	007054			BM		
1493	011214	104405			PARAM		
1494	011216	000000			0		

1495	011220	000377		377		
1496	011222	001254		TEMP4		
1497	011224	000		.BYTE	0	
1498	011225	001		.BYTE	1	
1499	011226	113722	001254	MOV8	TEMP4, (R2)+	; STORE SWITCH PAC IN MAP
1500	011232	005722		TST	(R2)+	; POP OVER STAT3
1501	011234	005337	001252	33\$: DEC	TEMP3	; DEC DMC COUNT
1502	011240	001402		BEQ	34\$	; BR IF DONE
1503	011242	000137	010612	JMP	12\$	; JUMP IF NOT
1504	011246	000137	011702	34\$: JMP	13\$	; CONTINUE
1505	011252	012701	160000	7\$: MOV	#160000, R1	; SET FOR FIRST ADDRESS TO BE TESTED
1506	011256	012737	011774 000004	MOV	#6\$, 2#4	; SET FOR NON-EXISTANT DEVICE TIME OUT
1507	011264	005011		2\$: CLR	(R1)	; CLEAR SEL0
1508	011266	005711		TST	(R1)	; IF DMC11 DMC5R S/B 0
1509	011270	001172		BNE	3\$	; IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO DMC1
1510	011272	005061	000006	CLR	6(R1)	; CLEAR SEL6
1511	011276	005761	000006	TST	6(R1)	; IF DMC11 THEN DMRIC S/B =0!
1512	011302	001165		BNE	3\$	; BR IF NOT DMC11
1513	011304	012711	002000	MOV	#BIT10, (R1)	; SET ROM0
1514	011310	005061	000004	CLR	4, R1)	; CLEAR SEL4
1515	011314	012761	125252 000006	MOV	#125252, 6(R1)	; WRITE THIS TO SEL6
1516	011322	052711	020000	BIS	#BIT13, (R1)	; WRITE IT!
1517	011326	022761	125252 000004	CMP	#125252, 4(R1)	; WAS IT WRITTEN?
1518	011334	001004		BNE	21\$	; IF NO IT IS NOT CRAM
1519	011336	052762	100000 000002	BIS	#BIT15, 2(R2)	; SET BIT15 IF CRAM
1520	011344	000431		BR	22\$	
1521	011346	012711	001000	21\$: MOV	#BIT9, (R1)	; SET ROM1
1522	011352	012761	100417 000006	MOV	#100417, 6(R1)	; PUT INSTRUCTION IN SEL6
1523	011360	012711	001400	MOV	#BIT9!BIT8, (R1)	; CLOCK INSTRUCTION (MICRO PROC PC TO 0)
1524	011364	012711	002000	MOV	#BIT10, (R1)	; SET ROM0
1525	011370	022761	000626 000006	CMP	#626, 6(R1)	; IS IT LOCAL CROM
1526	011376	001411		BEQ	23\$	; BR IF YES
1527	011400	022761	016520 000006	CMP	#16520, 6(R1)	; IS IT REMOTE CROM?
1528	011406	001410		BEQ	22\$	; BR IF YES
1529	011410	022761	177777 000006	CMP	#-1, 6(R1)	; NO CROM?
1530	011416	001404		BEQ	22\$	; BR IF YES
1531	011420	000516		BR	3\$	; NOT A DMC
1532	011422	052762	000002 000006	23\$: BIS	#BIT1, 6(R2)	; SET BIT 1 IN STAT3
1533				AT THIS POINT IT IS ASSUMED THAT	R1 HOLDS A DMC11 CSR ADDRESS.	
1534	011430	010122		22\$: MOV	R1, (R2)+	; STORE CSR IN CORE TABLE.
1535	011432	012711	001000	15\$: MOV	#BIT9, (R1)	; CLEAR LINE UNIT LOOP
1536	011436	005061	000004	CLR	4(R1)	; CLEAR PORT4
1537	011442	012761	122113 000006	MOV	#122113, 6(R1)	; LOAD INSTRUCTION (CLR DTR)
1538	011450	052711	000400	BIS	#BIT8, (R1)	; CLOCK INSTRUCTION
1539	011454	012761	021264 000006	MOV	#021264, 6(R1)	; LOAD INSTRUCTION
1540	011462	052711	000400	BIS	#BIT8, (R1)	; CLOCK INSTRUCTION
1541	011466	122761	000377 000004	CMP8	#377, 4(R1)	; IS IT ALL ONES?
1542	011474	001003		BNE	.+10	; BR IF NO
1543	011476	052712	010000	BIS	#BIT12, (R2)	; IF YES, NO LINE UNIT. SET STATUS BIT
1544	011502	000436		BR	20\$	
1545	011504	032761	000002 000004	BIT	#BIT1, 4(R1)	; IS SWITCH A ONE?
1546	011512	001403		BEQ	.+10	; BR IF M8201
1547	011514	052712	060000	BIS	#BIT13!BIT14, (R2)	; M8202 ASSUME CONNECTOR
1548	011520	000427		BR	20\$	; CONNECTOR ON)
1549	011522	032761	000010 000004	BIT	#BIT3, 4(R1)	; IS MRDY SET
1550	011530	001023		BNE	20\$	; BR IF M8201 NO CONNECTOR (ON LINE)

1551	011532	012761	000100	000004	MOV	#BIT6,4(R1)	;LOAD PORT4
1552	011540	012761	122113	000006	MOV	#122113,6(R1)	;LOAD INSTRUCTION
1553	011546	052711	000400		BIS	#BIT8,(R1)	;CLOCK INSTRUCTION(SET DTR)
1554	011552	012761	021264	000006	MOV	#021264,6(R1)	;LOAD INSTRUCTION
1555	011560	052711	000400		BIS	#BIT8,(R1)	;CLOCK INSTRUCTION(READ MODEM REG)
1556	011564	032761	000010	000004	BIT	#BIT3,4(R1)	;IS MRDY SET NOW?
1557	011572	001402			BEQ	20\$	;BR IF NO CONNECTOR
1558	011574	052712	040000		BIS	#BIT14,(R2)	;SET STATUS BIT FOR CONNECTOR
1559	011600	005722			20\$: TST	(R2)+	;POP POINTER
1560	011602	012761	021324	000006	MOV	#021324,6(R1)	;PUT INSTRUCTION IN PORT6
1561	011610	012711	001400		MOV	#BIT9!BIT8,(R1)	;PORT4+LU 15
1562	011614	156122	000004		BISB	4(R1),(R2)+	;STORE DDCMP LINE # IN TABLE
1563	011620	012761	021344	000006	MOV	#021344,6(R1)	;PORT6+INSTRUCTION
1564	011626	012711	001400		MOV	#BIT8!BIT9,(R1)	;CLOCK INSTR.
1565	011632	156122	000004		BISB	4(R1),(R2)+	;STORE BM873 ADD IN TABLE
1566	011636	005722			TST	(R2)+	;POP OVER STAT3
1567	011640	005011			CLR	(R1)	;CLEAR ROMI
1568	011642	005237	001310		INC	DMNUM	;UPDATE DEVICE COUNTER
1569	011646	022737	000020	001310	CMP	#20,DMNUM	;ARE MAX. NO. OF DEV FOUND?
1570	011654	001412			BEQ	13\$	;YES DON'T LOOK FOR ANY MORE.
1571	011656	005011			3\$: CLR	(R1)	;CLEAR BIT 10
1572	011660	005061	000006		CLR	6(R1)	;CLEAR SEL 6
1573	011664	062701	000010		14\$: ADD	#10,R1	;UPDATE CSR POINTER ADDRESS
1574	011670	022701	164000		CMP	#164000,R1	
1575	011674	001402			BEQ	13\$	;BR IF DONE
1576	011676	000137	011264		JMP	2\$	;JUMP IF NOT
1577	011702	005037	001306		13\$: CLR	DMACTV	
1578	011706	005737	001310		TST	DMNUM	;WERE ANY DMC11'S FOUND AT ALL?
1579	011712	001423			BEQ	5\$	;ERROR AUTO SIZER FOUND NO DMC11'S IN THIS SYS.
1580	011714	013701	001310		MOV	DMNUM,R1	
1581	011720	010137	001314		MOV	R1,SAVNUM	;SAVE NUMBER OF DEVICES
1582	011724	000241			4\$: CLC		
1583	011726	006137	001306		ROL	DMACTV	;GENERATE ACTIVE REGISTER OF DEVICES.
1584	011732	005237	001306		INC	DMACTV	;SET THE BIT
1585	011736	005301			DEC	R1	
1586	011740	001371			BNE	4\$	;BR IF MORE TO GENERATE
1587	011742	012737	000006	000004	MOV	#6,2#4	;RESTORE TRAP VECTOR
1588	011750	013737	001306	001312	MOV	DMACTV,SAVACT	;SAVE ACTIVE REGISTER
1589	011756	000137	012010		JMP	VECMAP	;GO FIND THE VECTOR NOW.
1590	011762	104402	005760		5\$: TYPE	MERR2	;NOTIFY OPR THAT NO DMC11'S FOUND.
1591	011766	005000			CLR	RO	;MAKE DATA LIGHTS ZERO
1592	011770	000000			HALT		;STOP THE SHOW
1593	011772	000776			BR	.-2	;DISABLE CONT. SW.
1594	011774	012716	011664		6\$: MOV	#14\$, (SP)	;ENTERED BY NON-EXISTANT TIME-OUT.
1595	012000	000002			RTI		;RETURN TO MAINSTREAM
1596							
1597	012002	000001			WHICH: 1		
1598	012004	002	002		.BYTE	2,2	
1599	012006	001256			TEMPS		
1600							
1601	012010	032737	000001	001236	VECMAP: BIT	#SW00,STATSW	
1602	012016	001114			BNE	5\$	
1603	012020	012737	000340	000022	MOV	#340,2#22	;SET IOT TRAP PRIO TO 7
1604	012026	012737	012202	000020	MOV	#4\$,2#20	;SET IOT TRAP VECTOR
1605	012034	012702	001500		MOV	#DM.MAP,R2	;SET SOFTWARE POINTER
1606	012040	012700	000300		MOV	#300,RO	;FLOATING VECTORS START HERE.

1607	012044	012701	000302		MOV	#302,R1	;PC OF IOT INSTR.
1608	012050	010120		1\$:	MOV	R1,(R0)+	;START FILLING VECTOR AREA
1609	012052	012721	000004		MOV	#4,(R1)+	;WITH .+2; IOT
1610	012056	022021			CMP	(R0)+(R1)+	;ADD 2 TO R0 +R1
1611	012060	020127	001000		CMP	R1,#1000	
1612	012064	101771			BLOS	1\$	;BR IF MORE TO FILL
1613	012066	013737	001306	001246	MOV	DMACTV,TEMP1	;STORE TEMPORALLY
1614	012074	006037	001246	2\$:	ROR	TEMP1	;BRING OUT A BIT
1615	012100	103063			BCC	5\$	;BR IF ALL DONE
1616	012102	012704	000012		MOV	#12,R4	;R4 IS INDEX REGISTER
1617	012106	016437	012252	177776	MOV	BRLVL(R4),PS	;SET PS TO 7
1618	012114	011201			MOV	(R2),R1	
1619	012116	012761	000200	000004	MOV	#200,4(R1)	
1620	012124	012711	001000		MOV	#BIT9,(R1)	;SET ROMI
1621	012130	012761	121111	000006	MOV	#121111,6(R1)	;PUT INSTRUCTION IN PORTE
1622	012136	012711	001400		MOV	#BIT9:BIT8,(R1)	;FORCE AN INTERRUPT
1623	012142	105200		7\$:	INCB	R0	;STALL
1624	012144	001376			BNE	.-2	;FOR TIME TO INTERRUPT
1625	012146	162704	000002		SUB	#2,R4	;GET NEXT LOWEST PS LEVEL
1626	012152	001404			BEQ	6\$	;BR IF R4 = 0
1627	012154	016437	012252	177776	MOV	BRLVL(R4),PS	;MOVE NEXT LOWER LEVEL IN PS
1628	012162	000767			BR	7\$	;BR TO DELAY
1629	012164	052762	005300	000002	6\$:	BIS	#5300,2(R2)
1630	012172	005011		3\$:	CLR	(R1)	;NO INTERRUPT ASSUME 300 AT LEVEL 5 AND FIX DMC11
1631	012174	062702	000010		ADD	#10,R2	;CLEAR ROMI
1632	012200	000735			BR	2\$	;POP SOFTWARE POINTER
1633	012202	051662	000002		4\$:	BIS	(SP),2(R2)
1634	012206	042762	000007	000002	BIC	#7,2(R2)	;KEEP GOING
1635	012214	016405	012254		MOV	BRLVL+2(R4),R5	;GET VECTOR ADDRESS
1636	012220	006305			ASL	R5	;CLEAR JUNK
1637	012222	006305			ASL	R5	;GET BR LEVEL OF DMC11
1638	012224	006305			ASL	R5	;SHIFT LEVEL 4 PLACES
1639	012226	006305			ASL	R5	;TO THE LEFT FOR THE
1640	012230	042705	170777		BIC	#170777,R5	;STATUS TABLE
1641	012234	050562	000002		BIS	R5,2(R2)	;CLEAR UNWANTED BITS
1642	012240	022626			CMP	(SP)+(SP)+	;PUT BR LEVEL IN STATUS TABLE
1643	012242	012716	012172		MOV	#3\$,(SP)	;POP IOT JUNK OFF STACK
1644	012246	000002			RTI		;SET FOR RETURN
1645	012250	000207		5\$:	RTS	PC	;ALL DONE WITH "AUTO SIZING"
1646							
1647	012252	000000		BRLVL:	0	;LEVEL 0	
1648	012254	000000			0	;LEVEL 0	
1649	012256	000200			200	;LEVEL 4	
1650	012260	000240			240	;LEVEL 5	
1651	012262	000300			300	;LEVEL 6	
1652	012264	000340			340	;LEVEL 7	
1653							
1654							
1655	012266	105777	166712	INTTY:	TSTB	@TKCSR	;WAIT FOR DONE
1656	012272	100375			BPL	.-4	
1657	012274	017703	166706		MOV	@TKDBR,R3	;PUT CHAR IN R3
1658	012300	105777	166704		TSTB	@TPCSR	;WAIT UNTIL PRINTER IS READY
1659	012304	100375			BPL	.-4	
1660	012306	010377	166700		MOV	R3,@TPDBR	;ECHO CHAR
1661	012312	042703	000240		BIC	#BIT7:BITS,R3	;MASK OFF LOWER CASE
1662	012316	000207			RTS	PC	;RETURN

1663
 1664
 1665
 1666
 1667
 1668
 1669
 1670
 1671
 1672
 1673
 1674
 1675
 1676
 1677
 1678
 1679
 1680
 1681
 1682
 1683
 1684
 1685
 1686
 1687
 1688
 1689
 1690
 1691
 1692
 1693
 1694
 1695
 1696
 1697
 1698
 1699
 1700
 1701
 1702
 1703
 1704
 1705
 1706
 1707
 1708
 1709
 1710
 1711
 1712
 1713
 1714
 1715
 1716
 1717
 1718

02100

```

***** 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 SELD
      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
      1$:      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
      1$:      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
*****

```

```

1719
1720
1721
1722 012442 012737 000003 001226
1723 012450 012737 012514 001216
1724
1725 012456 104412
1726 012460 012702 000013
1727 012464 104414
1728 012466 021264
1729 012470 016104 000004
1730 012474 042704 000213
1731 012500 012705 000100
1732 012504 120504
1733 012506 001401
1734 012510 104002
1735 012512 104400
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746 012514 012737 000004 001226
1747 012522 012737 012616 001216
1748
1749 012530 104412
1750 012532 012702 000017
1751 012536 104414
1752 012540 021364
1753 012542 016104 000004
1754 012546 042704 000206
1755 012552 012705 000051
1756 012556 032737 020000 001366
1757 012564 001004
1758 012566 032737 040000 001366
1759 012574 001004
1760 012576 042704 000040
1761 012602 042705 000040
1762 012606
1763 012606 120504
1764 012610 001401
1765 012612 104002
1766 012614 104400
1767
1768
1769
1770
1771
1772
1773
1774

```

```

; TEST 3
-----
TST3: MOV #3,TSTNO
      MOV #TST4,NEXT

      MSTCLR
      MOV #13,R2
      ROMCLK
      021004!<20*13>
      MOV 4(R1),R4
      BIC #213,R4
      MOV #100,R5
      CMPB R5,R4
      BEQ 1$
      HLT 2
1$: SCOPE

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SAVE R2 FOR TYPEOUT
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4+LINE UNIT REG 13
;PUT "FOUND" IN R4
;CLEAR UNKNOWN BITS
;PUT "EXPECTED" IN R5
;ARE RING, DTR, AND MODEM READY SET?
;BR IF YES
;ERROR IN LU 13
;SCOPE THIS TEST

;***** TEST 4 *****
;MAINTENANCE REGISTER READ/ONLY TEST
;DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
;BITS ARE IN THE CORRECT STATE
;*****

; TEST 4
-----
TST4: MOV #4,TSTNO
      MOV #TST5,NEXT

      MSTCLR
      MOV #17,R2
      ROMCLK
      021004!<20*17>
      MOV 4(R1),R4
      BIC #206,R4
      MOV #51,R5
      BIT #BIT13,STAT1
      BNE 2$
      BIT #BIT14,STAT1
      BNE 3$
      BIC #40,R4
      BIC #BIT5,R5
      CMPB R5,R4
      BEQ 1$
      HLT 2
2$:
3$:
1$: SCOPE

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SAVE R2 FOR TYPEOUT
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4+LINE UNIT REG 17
;PUT "FOUND" IN R4
;CLEAR UNKNOWN BITS
;PUT "EXPECTED" IN R5
;IS LU AN M8202 OR M8201?
;BR IF M8202
;CONNECTOR???
;BR IF M8201 WITH CONNECTOR
;MASK OFF SI BIT IF M8202 OR M8201, NO CONNECTOR
;SI BIT IS UNKNOWN
;ARE SI AND ICIR SET?
;BR IF YES
;ERROR IN LU 17
;SCOPE THIS TEST

;***** TEST 5 *****
;LINE UNIT REGISTER WRITE/READ TEST
;SET BITS IN LU REGISTER 12, VERIFY IT IS SET
;CLEAR BITS IN LU REGISTER 12, VERIFY IT IS CLEAR
;*****

```

```

1775      ; TEST 5
1776      ;-----
1777      012616 012737 000005 001226      TST5:  MOV      #5,TSTNO
1778      012624 012737 012756 001216      MOV      #TST6,NEXT
1779      012632 012737 012646 001220      MOV      #1$,LOCK
1780      ;R1 CONTAINS BASE DMC11 ADDRESS
1781      012640 104412      MSTCLR      ;MASTER CLEAR DMC11
1782      012642 012702 000012      MOV      #12,R2      ;SAVE REGISTER ADDRESS FOR TYPEOUT
1783      012646 012761 000040 000004      1$:  MOV      #40,4(R1)      ;LOAD PORT4
1784      012654 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1785      012656 122112      122112      ;SET BITS IN LU-12
1786      012660 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1787      012662 021245      021245      ;READ LU-12
1788      012664 012705 000040      MOV      #40,R5      ;PUT "EXPECTED" IN R5
1789      012670 116104 000005      MOV      5(R1),R4      ;PUT "FOUND" IN R4
1790      012674 042704 000337      BIC      #337,R4      ;CLEAR UNWANTED BITS
1791      012700 120504      CMPB      R5,R4      ;IS BITS SET?
1792      012702 001401      BEQ      2$      ;BR IF YES
1793      012704 104003      HLT      3      ;ERROR, BIT 5 IS NOT SET
1794      012706 104401      2$:  SCOP1      ;SCOPE SUBTEST (SW09=1)
1795      012710 012737 012716 001220      MOV      #3$,LOCK      ;NEW SCOP1
1796      012716 005061 000004      3$:  CLR      4(R1)      ;LOAD PORT4
1797      012722 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1798      012724 122112      122112      ;CLEAR BIT 5 IN LU-12
1799      012726 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1800      012730 021245      021245      ;READ LU-12
1801      012732 005005      CLR      R5      ;PUT "EXPECTED" IN R5
1802      012734 116104 000005      MOV      5(R1),R4      ;PUT "FOUND" IN R4
1803      012740 042704 000337      BIC      #337,R4      ;CLEAR UNWANTED BITS
1804      012744 120504      CMPB      R5,R4      ;IS BITS CLEAR?
1805      012746 001401      BEQ      4$      ;BR IF YES
1806      012750 104003      HLT      3      ;ERROR, BITS IS NOT CLEAR
1807      012752 104401      4$:  SCOP1      ;SCOPE SUBTEST (SW09=1)
1808      012754 104400      SCOPE      ;SCOPE THIS TEST
1809
1810      ;***** TEST 6 *****
1811      ;*LINE UNIT REGISTER WRITE/READ TEST
1812      ;*SET BIT1 IN LU REGISTER 17, VERIFY IT IS SET
1813      ;*CLEAR BIT1 IN LU REGISTER 17, VERIFY IT IS CLEAR
1814      ;*****
1815
1816      ; TEST 6
1817      ;-----
1818
1819      012756 012737 000006 001226      TST6:  MOV      #6,TSTNO
1820      012764 012737 013116 001216      MOV      #TST7,NEXT
1821      012772 012737 013006 001220      MOV      #1$,LOCK
1822      ;R1 CONTAINS BASE DMC11 ADDRESS
1823      013000 104412      MSTCLR      ;MASTER CLEAR DMC11
1824      013002 012702 000017      MOV      #17,R2      ;SAVE REGISTER ADDRESS FOR TYPEOUT
1825      013006 012761 000001 000004      1$:  MOV      #1,4(R1)      ;LOAD PORT4
1826      013014 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1827      013016 122117      122117      ;SET BIT1 IN LU-17
1828      013020 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1829      013022 021365      021365      ;READ LU-17
1830      013024 012705 000001      MOV      #1,R5      ;PUT "EXPECTED" IN R5

```

1831	013030	116104	000005		MOV B	5(R1), R4	: PUT "FOUND" IN R4
1832	013034	042704	000376		BIC	#376, R4	: CLEAR UNWANTED BITS
1833	013040	120504			CMP B	R5, R4	: IS BIT1 SET?
1834	013042	001401			BEQ	2\$	: BR IF YES
1835	013044	104003			HLT	3	: ERROR, BIT 1 IS NOT SET
1836	013046	104401			SCOP1		: SCOPE SUBTEST (SW09=1)
1837	013050	012737	013056	001220	MOV	#3\$, LOCK	: NEW SCOPE
1838	013056	005061	000004		CLR	4(R1)	: LOAD PORT4
1839	013062	104414			ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1840	013064	122117			122117		: CLEAR BIT 1 IN LU-17
1841	013066	104414			ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1842	013070	021365			021365		: READ LU-17
1843	013072	005005			CLR	R5	: PUT "EXPECTED" IN R5
1844	013074	116104	000005		MOV B	5(R1), R4	: PUT "FOUND" IN R4
1845	013100	042704	000376		BIC	#376, R4	: CLEAR UNWANTED BITS
1846	013104	120504			CMP B	R5, R4	: IS BIT1 CLEAR?
1847	013106	001401			BEQ	4\$	: BR IF YES
1848	013110	104003			HLT	3	: ERROR, BIT1 IS NOT CLEAR
1849	013112	104401			SCOP1		: SCOPE SUBTEST (SW09=1)
1850	013114	104400			SCOPE		: SCOPE THIS TEST
1851							
1852							
1853							
1854							
1855							
1856							
1857							
1858							
1859							
1860							
1861	013116	012737	000007	001226	TST7:	MOV	#7, TSTNO
1862	013124	012737	013326	001216		MOV	#TST10, NEXT
1863	013132	012737	013152	001220		MOV	#64\$, LOCK
1864							
1865	013140	104412			MSTCLR		: R1 CONTAINS BASE DMC11 ADDRESS
1866	013142	012702	000013		MOV	#13, R2	: MASTER CLEAR DMC11
1867	013146	012700	000001		MOV	#1, R0	: SAVE REGISTER ADDRESS FOR TYPEOUT
1868	013152				64\$:		: START WITH BIT 0
1869	013152	010061	000004		MOV	R0, 4(R1)	: PUT PATTERN INTO PORT4
1870	013156	042761	000257	000004	BIC	#257, 4(R1)	: CLEAR UNWANTED BITS
1871	013164	104414			ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1872	013166	122113			122100:13		: MOV DATA TO IBUS REGISTER 13
1873	013170	104414			ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1874	013172	021265			21005:(13*20)		: READ FROM IBUS REGISTER 13
1875	013174	010005			MOV	R0, R5	: PUT EXPECTED IN R5
1876	013176	042705	000257		BIC	#257, R5	: CLEAR UNWANTED BITS
1877	013202	116104	000005		MOV B	5(R1), R4	: PUT "FOUND" INTO R4
1878	013206	042704	000257		BIC	#257, R4	: CLEAR UNWANTED BITS
1879	013212	120504			CMP B	R5, R4	: DATA CORRECT?
1880	013214	001401			BEQ	65\$	: BR IF YES
1881	013216	104003			HLT	3	: ERROR
1882	013220	104401			SCOP1		: SW09=1?
1883	013222	000241			CLC		: CLEAR CARRY
1884	013224	106100			ROL B	R0	: SHIFT BIT IN R0
1885	013226	001351			BNE	64\$	: IF R0=0 THEN DONE
1886	013230	012737	013244	001220	MOV	#67\$, LOCK	: NEW SCOPE

***** 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

Address	Hex Address	Hex Data	Hex Data	Hex Data	Assembly	Comments
1887	013236	012700	000001		MOV	#1,R0 ;START WITH BIT 0
1888	013242	005100			COM	R0 ;CHANGE TO FLOATING ZERO
1889	013244					
1890	013244	010061	000004		MOV	R0,4(R1) ;PUT PATTERN INTO PORT4
1891	013250	042761	000257	000004	BIC	#257,4(R1) ;CLEAR UNWANTED BITS
1892	013256	104414			ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1893	013260	122113			122100!13	;MOV DATA TO IBUS REGISTER 13
1894	013262	104414			ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1895	013264	021265			21005!<13*20>	;READ FROM IBUS REGISTER 13
1896	013266	010005			MOV	R0,R5 ;PUT EXPECTED IN R5
1897	013270	042705	000257		BIC	#257,R5 ;CLEAR UNWANTED BITS
1898	013274	116104	000005		MOVB	5(R1),R4 ;PUT "FOUND" INTO R4
1899	013300	042704	000257		BIC	#257,R4 ;CLEAR UNWANTED BITS
1900	013304	120504			CMPB	R5,R4 ;DATA CORRECT?
1901	013306	001401			BEQ	68\$;BR IF YES
1902	013310	104003			HLT	3 ;ERROR
1903	013312	104401			68\$: SCOP1	;SW09=1?
1904	013314	005100			COM	R0 ;CHANGE TO FLOATING 1
1905	013316	000241			CLC	;CLEAR CARRY
1906	013320	106100			ROLB	R0 ;SHIFT BIT IN R0
1907	013322	001347			BNE	69\$;IF R0=0 THEN DONE
1908	013324	104400			SCOPE	;SCOPE THIS TEST
1909						
1910						
1911						
1912						
1913						
1914						
1915						
1916						
1917						
1918						
1919	013326	012737	000010	001226	TST10: MOV	#10,TSTNO
1920	013334	012737	013502	001216	MOV	#TST11,NEXT
1921	013342	012737	013362	001220	MOV	#64\$,LOCK
1922						
1923	013350	104412			MSTCLR	;R1 CONTAINS BASE DMC11 ADDRESS
1924	013352	012702	000014		MOV	#14,R2 ;MASTER CLEAR DMC11
1925	013356	012700	000001		MOV	#1,R0 ;SAVE REGISTER ADDRESS FOR TYPEOUT
1926	013362					;START WITH BIT 0
1927	013362	010061	000004		MOV	R0,4(R1) ;PUT PATTERN INTO PORT4
1928	013366	104414			ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1929	013370	122114			122100!14	;MOV DATA TO IBUS REGISTER 14
1930	013372	104414			ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1931	013374	021305			21005!<14*20>	;READ FROM IBUS REGISTER 14
1932	013376	010005			MOV	R0,R5 ;PUT EXPECTED IN R5
1933	013400	116104	000005		MOVB	5(R1),R4 ;PUT "FOUND" INTO R4
1934	013404	120504			CMPB	R5,R4 ;DATA CORRECT?
1935	013406	001401			BEQ	65\$;BR IF YES
1936	013410	104003			HLT	3 ;ERROR
1937	013412	104401			65\$: SCOP1	;SW09=1?
1938	013414	000241			CLC	;CLEAR CARRY
1939	013416	106100			RCLB	R0 ;SHIFT BIT IN R0
1940	013420	001360			BNE	64\$;IF R0=0 THEN DONE
1941	013422	012737	013436	001220	MOV	#67\$,LOCK ;NEW SCOP1
1942	013430	012700	000001		MOV	#1,R0 ;START WITH BIT 0

DZOME MACY11 30(1046) 11-JUL-77 11:40 PAGE 39
 DZOME.P11 12-MAY-77 14:18 LINE UNIT 1

000ME.P11 12-MAY-77 14:18

LINE UNIT WRITE/READ TESTS

Line	Address	Instruction	Comment
1943	013434	005100	
1944	013436		
1945	013436	010061	000004
1946	013442	104414	
1947	013444	122114	
1948	013446	104414	
1949	013450	021305	
1950	013452	010005	
1951	013454	116104	000005
1952	013460	120504	
1953	013462	001401	
1954	013464	104003	
1955	013466	104401	
1956	013470	005100	
1957	013472	000241	
1958	013474	106100	
1959	013476	001356	
1960	013500	104400	
1961			
1962			
1963			
1964			
1965			
1966			
1967			
1968			
1969			
1970			
1971	013502	012737	000011 001226
1972	013510	012737	013544 001216
1973			
1974	013516	104412	
1975	013520	104414	
1976	013522	021324	
1977	013524	016104	000004
1978	013530	113705	001370
1979	013534	120504	
1980	013536	001401	
1981	013540	104031	
1982	013542	104400	
1983			
1984			
1985			
1986			
1987			
1988			
1989			
1990			
1991			
1992			
1993	013544	012737	000012 001226
1994	013552	012737	013606 001216
1995			
1996	013560	104412	
1997	013562	104414	
1998	013564	021344	

LINE UNIT WRITE/READ TESTS

```

1999 013566 016104 000004      MOV      4(R1),R4      ;PUT "FOUND" IN R4
2000 013572 113705 001371      MOVB     STAT2+1,R5    ;PUT "EXPECTED" IN R5
2001 013576 120504      CMPB     R5,R4      ;SW OK?
2002 013600 001401      BEQ      1$          ;BR IF YES
2003 013602 104031      HLT      31          ;ERROR, SWITCH PAC READ ERROR
2004 013604 104400      1$:      SCOPE      ;SCOPE THIS TEST
2005
2006
2007      ;***** TEST 13 *****
2008      ;*LINE UNIT CLOCK TEST
2009      ;*THIS TEST VERIFYS THAT THE LU INTERNAL CLOCK
2010      ;*(BIT 1 IN LU-17) IS WORKING
2011      ;*****
2012
2013      : TEST 13
2014      :-----
2015 013606 012737 000013 001226  TST13: MOV      #13,TSTNO
2016 013614 012737 013706 001216      MOV      #TST14,NEXT
2017      ;R1 CONTAINS BASE DMC11 ADDRESS
2018 013622 104412      MSTCLR      ;MASTER CLEAR DMC11
2019 013624 005037 001416      CLR      TEMP      ;PREPARE FOR DELAY
2020 013630      1$:
2021 013630 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION. ROMCLK PC=5304
2022 013632 021364 021364      021364      ;PORT4+LU-17
2023 013634 032761 000002 000004      BIT      #2,4(R1)      ;IS CLOCK BIT SET?
2024 013642 001004      BNE      2$          ;BR IF YES
2025 013644 005237 001416      INC      TEMP      ;DELAY
2026 013650 001367      BNE      1$          ;DELAY FINISHED?
2027 013652 104004      HLT      4          ;ERROR BIT IS STUCK CLEAR
2028 013654 005037 001416      2$:      CLR      TEMP      ;PREPARE FOR DELAY
2029 013660      3$:
2030 013660 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION. ROMCLK PC=5304
2031 013662 021364 021364      021364      ;PORT4+LU-17
2032 013664 032761 000002 000004      BIT      #2,4(R1)      ;IS CLOCK BIT CLEAR?
2033 013672 001404      BEQ      4$          ;BR IF YES
2034 013674 005237 001416      INC      TEMP      ;DELAY
2035 013700 001367      BNE      3$          ;BR IF DELAY NOT DONE
2036 013702 104004      HLT      4          ;ERROR BIT IS STUCK SET
2037 013704 104400      4$:      SCOPE
2038
2039
2040      ;***** TEST 14 *****
2041      ;*OUT DATA SILO TEST
2042      ;*SET SOM AND LOAD OUT DATA SILO
2043      ;*VERIFY THAT OCOR SET, INDICATING THAT THE
2044      ;*CHARACTER IS AT THE BOTTOM OF THE OUT SILO
2045      ;*****
2046
2047      : TEST 14
2048      :-----
2049 013706 012737 000014 001226  TST14: MOV      #14,TSTNO
2050 013714 012737 014006 001216      MOV      #TST15,NEXT
2051      ;R1 CONTAINS BASE DMC11 ADDRESS
2052 013722 104412      MSTCLR      ;MASTER CLEAR DMC11
2053 013724 012711 004000      MOV      #BIT11,(R1)      ;SET LINE UNIT LOOP
2054 013730 012761 000001 000304      MOV      #1,4,R1      ;LOAD PORT4 WITH BITC

```

15:

```

ROMCLK      122111
ROMCLK      122110
TIMER,      2
MOV         #17,R2
ROMCLK      021364
MOV         4(R1),R4
BIC        #357,R4
MOV         #20,R5
CMPB       R5,R4
BEQ        15
HLT         5

```

```

: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
: SET SOM
: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
: LOAD OUT DATA SILO
: WAIT FOR OCOR
: SAVE ADDRESS FOR TYPEOUT
: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
: PORT4←LU 17
: PUT "FOUND" IN R4
: CLEAR UNWANTED BITS
: PUT "EXPECTED" IN R5
: IS OCOR SET?
: BR IF YES

```

SCOPE

:SCOPE THIS TEST

```
;***** TEST 15 *****
;*DDCMP TEST OF RTS AND OUT ACTIVE
;*SET SOM AND LOAD OUT DATA SILO
;*SINGLE STEP 2 DATA CLOCKS, VERIFY
;*THAT RTS AND ACTIVE ARE SET
;*****
```

: TEST 15

TST15:

```
MOV     #15,TSTNO
MOV     #TST16,NEXT
```

```

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SET LINE UNIT LOOP
;LOAD PORT4 WITH BIT0
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET SOM
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD OUT DATA SILO
;WAIT FOR OCOR
;CLOCK DATA FOUR TIMES
;SAVE ADDRESS FOR TYPEOUT
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4+LU 11
;PUT "FOUND" IN R4
;CLEAR UNWANTED BITS
;PUT "EXPECTED" IN R5
;IS ACTIVE SET?
;BR IF YES

```

15:

```
MOV      #13,R2
POMCLK
021264
MOV      4(R1),R4
BIC      #337,R4
MOV      #BITS,R5
CMPB     R5,R4
```

```
;SAVE ADDRESS FOR TYPEOUT
;NEXT WORD IS INSTRUCTION. R3MCLK PC=5304
;PORT4+LU 13
;PUT EXPECTED IN R4
;CLEAR UNWANTED BITS
;PUT "EXPECTED" IN R5. RTS SHOULD BE SET
;IS RTS OK?
```

2111	014136	001401			BEQ	2\$		:BB IF YES
2112	014140	104005			HLT	5		:RTS ERROR
2113	014142							
2114	014142	104400						:SCOPE THIS TEST
2115								
2116								
2117								:***** TEST 16 *****
2118								:*TEST OF OUT CLEAR
2119								:*SET SOM AND LOAD OUT DATA SILO
2120								:*SINGLE STEP DATA CLOCK, SET OUT CLEAR
2121								:*VERIFY THAT OCOR, RTS, AND ACTIVE ARE CLEARED
2122								:*****
2123								
2124								: TEST 16
2125								
2126	014144	012737	000016	001226	TST16:	MOV	#16,TSTNO	
2127	014152	012737	014342	001216		MOV	#TST17,NEXT	
2128								:R1 CONTAINS BASE DMC11 ADDRESS
2129	014160	104412				MSTCLR		:MASTER CLEAR DMC11
2130	014162	012711	004000			MOV	#BIT11,(R1)	:SET LINE UNIT LOOP
2131	014166	012761	000001	000004		MOV	#1,4(R1)	:LOAD PORT4 WITH BIT0
2132	014174	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2133	014176	122111				122111		:SET SOM
2134	014200	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2135	014202	122110				122110		:LOAD OUT DATA SILO
2136	014204	004737	026350			JSR	PC,OCOR	:WAIT FOR OCOR
2137	014210	104415	000002			DATACLK,	2	:CLOCK DATA FOUR TIMES
2138	014214	012761	000200	000004		MOV	#BIT7,4(R1)	:SET BIT7 IN PORT4
2139	014222	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2140	014224	122111				122111		:SET OUT CLEAR
2141	014226	104415	000001			DATACLK,	1	:GIVE A TICK TO CLEAR RTS
2142	014232	012702	000017			MOV	#17,R2	:SAVE ADDRESS FOR TYPEOUT
2143	014236	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2144	014240	021364				021364		:PORT4+LU 17
2145	014242	016104	000004			MOV	4(R1),R4	:PUT "FOUND" IN R4
2146	014246	042704	000357			BIC	#357,R4	:CLEAR UNWANTED BITS
2147	014252	005005				CLR	R5	:PUT "EXPECTED" IN R5
2148	014254	120504				CMPB	R5,R4	:IS OCOR CLEARED?
2149	014256	001401				BEQ	1\$	:BR IF YES
2150	014260	104005				HLT	5	
2151	014262				1\$:			
2152	014262	012702	000013			MOV	#13,R2	:SAVE ADDRESS FOR TYPEOUT
2153	014266	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2154	014270	021264				021264		:PORT4+LU 13
2155	014272	016104	000004			MOV	4(R1),R4	:PUT EXPECTED IN R4
2156	014276	042704	000337			BIC	#337,R4	:CLEAR UNWANTED BITS
2157	014302	005005				CLR	R5	:PUT "EXPECTED" IN R5, RTS SHOULD BE CLEARED
2158	014304	120504				CMPB	R5,R4	:IS RTS OK?
2159	014306	001401				BEQ	2\$	:BR IF YES
2160	014310	104005				HLT	5	:RTS ERROR
2161	014312				2\$:			
2162	014312	012702	000011			MOV	#11,R2	:SAVE ADDRESS FOR TYPEOUT
2163	014316	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2164	014320	021224				021224		:PORT4+LU11
2165	014322	016104	000004			MOV	4(R1),R4	:PUT "FOUND" IN R4
2166	014326	012705	000020			MOV	#BIT4,R5	:ONLY OUT READY SHOULD BE SET

2167	014332	120504				CMPB	R5,R4		;IS ACTIVE CLEAR?
2168	014334	001401				BEQ	3\$		;BR IF YES
2169	014336	104005				HLT	5		;ERROR ACTIVE NOT CLEARED
2170	014340				3\$:				
2171	014340	104400				SCOPE			;SCOPE THIS TEST
2172									
2173									
2174									
2175									
2176									
2177									
2178									
2179									
2180									
2181									
2182									
2183									
2184	014342	012737	000017	001226					
2185	014350	012737	014524	001216	TST17:	MOV	#17,TSTNO		
2186						MOV	#TST20,NEXT		
2187	014356	104412				MSTCLR			;R1 CONTAINS BASE DMC11 ADDRESS
2188	014360	012711	004000			MOV	#BIT11,(R1)		;MASTER CLEAR DMC11
2189	014364	004737	026502			JSR	PC,OUTRDY		;SET LINE UNIT LOOP
2190	014370	012761	000001	000004		MOV	#1,4(R1)		;WAIT FOR OUT-READY
2191	014376	104414				ROMCLK			;SET BIT0 IN PORT4
2192	014400	122111				122111			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2193	014402	012705	000000			MOV	#0,R5		;SET SOM!
2194	014406	004737	026502			JSR	PC,OUTRDY		;LOAD CHARACTER IN R5 FOR TYPEOUT
2195	014412	010561	000004			MOV	R5,4(R1)		;WAIT FOR OUT-READY
2196	014416	104414				ROMCLK			;LOAD PORT4 WITH CHARACTER
2197	014420	122110				122110			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2198	014422	004737	026350			JSR	PC,OCOR		;LOAD OUT DATA
2199	014426	005003				CLR	R3		;WAIT FOR OCOR TO SET
2200	014430	010502				MOV	R5,R2		;CLEAR BIT COUNTER
2201	014432	104415	000002			DATACLK,		2	;LOAD CHARACTER IN R2
2202	014436	104415	000001		1\$:	DATACLK,		1	;2 TICKS TO SET UP TRANSMITTER
2203	014442	106002				RORB	R2		;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2204	014444	103005				BCC	2\$		;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2205	014446	004737	026316			JSR	PC,GETSI		;BR IF CARRY CLEAR
2206	014452	103406				BCS	3\$		;GET THE WINDOW
2207	014454	104006				HLT	6		;BR IF BIT IS A MARK
2208	014456	000404				BR	3\$		;ERROR BIT WAS A SPACE
2209	014460	004737	026316		2\$:	JSR	PC,GETSI		;CONTINUE WITH TEST
2210	014464	103001				BCC	3\$		;GET THE WINDOW
2211	014466	104006				HLT	6		;BR IF BIT IS A SPACE
2212	014470				3\$:				;ERROR BIT WAS A MARK
2213	014470	005203				INC	R3		;NEXT BIT
2214	014472	022703	000010			CMP	#10,R3		;DONE YET?
2215	014476	001357				BNE	1\$		;BR IF NO
2216	014500	104415	000014			DATACLK,		14	;CLOCK TRANSMITTER 14 MORE TICKS
2217	014504	104414				ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2218	014506	021264				021264			;PORT4+LU-13
2219	014510	032761	000040	000004		BIT	#BIT5,4(R1)		;RTS SHOULD BE CLEAR NOW
2220	014516	001401				BEQ	4\$		;BR IF YES
2221	014520	104034				HLT	34		;ERROR, RTS NOT CLEAR
2222	014522	104400			4\$:	SCOPE			;SCOPE THIS TEST

```

2223
2224
2225      ;***** TEST 20 *****
2226      ;*DDCMP TRANSMITTER TEST
2227      ;*SINGLE CLOCK THE CHARACTER 125
2228      ;*VERIFY EACH BIT POSITION AS IT
2229      ;*PASSES THE BIT WINDOW (SI BIT)
2230      ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2231      ;*****
2232
2233      ; TEST 20
2234      -----
2235      TST20:  MOV     #20,TSTNO
2236              MOV     #TST21,NEXT
2237
2238              MSTCLR   ;R1 CONTAINS BASE DMC11 ADDRESS
2239              MOV     #BIT11,(R1) ;MASTER CLEAR DMC11
2240              JSR     PC,OUTRDY ;SET LINE UNIT LOOP
2241              MOV     #1,4(R1) ;WAIT FOR OUT-READY
2242              ROMCLK  ;SET BIT0 IN PORT4
2243              122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2244              MOV     #125,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
2245              JSR     PC,OUTRDY ;WAIT FOR OUT-READY
2246              MOV     R5,4(R1) ;LOAD PORT4 WITH CHARACTER
2247              ROMCLK  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2248              122110 ;LOAD OUT DATA
2249              JSR     PC,OCOR ;WAIT FOR OCOR TO SET
2250              CLR     R3 ;CLEAR BIT COUNTER
2251              MOV     R5,R2 ;LOAD CHARACTER IN R2
2252              DATACLK, 2 ;2 TICKS TO SET UP TRANSMITTER
2253              DATACLK, 1 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2254              RORB     R2 ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2255              BCC     2$ ;BR IF CARRY CLEAR
2256              JSR     PC,GETSI ;GET THE WINDOW
2257              BCS     3$ ;BR IF BIT IS A MARK
2258              HLT     6 ;ERROR BIT WAS A SPACE
2259              BR      3$ ;CONTINUE WITH TEST
2260              JSR     PC,GETSI ;GET THE WINDOW
2261              BCC     3$ ;BR IF BIT IS A SPACE
2262              HLT     6 ;ERROR BIT WAS A MARK
2263
2264              3$: INC     R3 ;NEXT BIT
2265              CMP     #10,R3 ;DONE YET?
2266              BNE     1$ ;BR IF NO
2267              DATACLK, 14 ;CLOCK TRANSMITTER 14 MORE TICKS
2268              ROMCLK  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2269              021264 ;PORT4+LU-13
2270              BIT     #BIT5,4(R1) ;RTS SHOULD BE CLEAR NOW
2271              BEQ     4$ ;BR IF YES
2272              HLT     34 ;ERROR, RTS NOT CLEAR
2273              4$: SCOPE ;SCOPE THIS TEST
2274
2275      ;***** TEST 21 *****
2276      ;*DDCMP TRANSMITTER TEST
2277      ;*SINGLE CLOCK THE CHARACTER 252
2278

```

```

2279 ;*VERIFY EACH BIT POSITION AS IT
2280 ;*PASSES THE BIT WINDOW (SI BIT)
2281 ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2282 ;*****
2283
2284 ; TEST 21
2285 -----
2286 014706 012737 000021 001226 TST21: MOV #21,TSTNO ;R1 CONTAINS BASE DMC11 ADDRESS
2287 014714 012737 015070 001216 MOV #TST22,NEXT ;MASTER CLEAR DMC11
2288 ;
2289 014722 104412 MSTCLR ;SET LINE UNIT LOOP
2290 014724 012711 004000 MOV #BIT11,(R1) ;WAIT FOR OUT-READY
2291 014730 004737 026502 JSR PC,OUTRDY ;SET BIT0 IN PORT4
2292 014734 012761 000001 000004 MOV #1,4(R1) ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2293 014742 104414 ROMCLK ;SET SOM!
2294 014744 122111 122111 MOV #252,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
2295 014746 012705 000252 JSR PC,OUTRDY ;WAIT FOR OUT-READY
2296 014752 004737 026502 MOV R5,4(R1) ;LOAD PORT4 WITH CHARACTER
2297 014756 010561 000004 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2298 014762 104414 122110 ;LOAD OUT DATA
2299 014764 122110 JSR PC,OCOR ;WAIT FOR OCOR TO SET
2300 014766 004737 026350 CLR R3 ;CLEAR BIT COUNTER
2301 014772 005003 MOV R5,R2 ;LOAD CHARACTER IN R2
2302 014774 010502 DATACLK, 2 ;2 TICKS TO SET UP TRANSMITTER
2303 014776 104415 000002 DATACLK, 1 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2304 015002 104415 000001 RORB R2 ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2305 015006 106002 BCC 2$ ;BR IF CARRY CLEAR
2306 015010 103005 JSR PC,GETSI ;GET THE WINDOW
2307 015012 004737 026316 BCS 3$ ;BR IF BIT IS A MARK
2308 015016 103406 HLT 6 ;ERROR BIT WAS A SPACE
2309 015020 104006 BR 3$ ;CONTINUE WITH TEST
2310 015022 000404 JSR PC,GETSI ;GET THE WINDOW
2311 015024 004737 026316 BCC 3$ ;BR IF BIT IS A SPACE
2312 015030 103001 HLT 6 ;ERROR BIT WAS A MARK
2313 015032 104006
2314 015034
2315 015034 005203 INC R3 ;NEXT BIT
2316 015036 022703 000010 CMP #10,R3 ;DONE YET?
2317 015042 001357 BNE 1$ ;BR IF NO
2318 015044 104415 000014 DATACLK, 14 ;CLOCK TRANSMITTER 14 MORE TICKS
2319 015050 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2320 015052 021264 021264 ;PORT4+LU-13
2321 015054 032761 000040 000004 BIT #BITS,4(R1) ;RTS SHOULD BE CLEAR NOW
2322 015062 001401 BEQ 4$ ;BR IF YES
2323 015064 104034 HLT 34 ;ERROR, RTS NOT CLEAR
2324 015066 104400 4$: SCOPE ;SCOPE THIS TEST
2325
2326 ;***** TEST 22 *****
2327 ;*DDCMP TRANSMITTER TEST
2328 ;*SINGLE CLOCK THE CHARACTER 377
2329 ;*VERIFY EACH BIT POSITION AS IT
2330 ;*PASSES THE BIT WINDOW (SI BIT)
2331 ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2332 ;*****
2333
2334

```

```

2335 ; TEST 22
2336 -----
2337 015070 012737 000022 001226 TST22: MOV #22,TSTNO
2338 015076 012737 015252 001216 MOV #TST23,NEXT
2339 ; R1 CONTAINS BASE DMC11 ADDRESS
2340 015104 104412 MSTCLR ; MASTER CLEAR DMC11
2341 015106 012711 004000 MOV #BIT11,(R1) ; SET LINE UNIT LOOP
2342 015112 004737 026502 JSR PC,OUTRDY ; WAIT FOR OUT-READY
2343 015116 012761 000001 000004 MOV #1,4(R1) ; SET BIT0 IN PORT4
2344 015124 104414 ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2345 015126 122111 122111 ; SET SOM!
2346 015130 012705 000377 MOV #377,R5 ; LOAD CHARACTER IN R5 FOR TYPEOUT
2347 015134 004737 026502 JSR PC,OUTRDY ; WAIT FOR OUT-READY
2348 015140 010561 000004 MOV R5,4(R1) ; LOAD PORT4 WITH CHARACTER
2349 015144 104414 ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2350 015146 122110 122110 ; LOAD OUT DATA
2351 015150 004737 026350 JSR PC,OCOR ; WAIT FOR OCOR TO SET
2352 015154 005003 CLR R3 ; CLEAR BIT COUNTER
2353 015156 010502 MOV R5,R2 ; LOAD CHARACTER IN R2
2354 015160 104415 000002 DATACLK, 2 ; 2 TICKS TO SET UP TRANSMITTER
2355 015164 104415 000001 1$ DATACLK, 1 ; SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2356 015170 106002 RORB R2 ; SHIFT NEXT SOFTWARE BIT IN TO CARRY
2357 015172 103005 BCC 2$ ; BR IF CARRY CLEAR
2358 015174 004737 026316 JSR PC,GETSI ; GET THE WINDOW
2359 015200 103406 BCS 3$ ; BR IF BIT IS A MARK
2360 015202 104006 HLT 6 ; ERROR BIT WAS A SPACE
2361 015204 000404 BR 3$ ; CONTINUE WITH TEST
2362 015206 004737 026316 JSR PC,GETSI ; GET THE WINDOW
2363 015212 103001 BCC 3$ ; BR IF BIT IS A SPACE
2364 015214 104006 HLT 6 ; ERROR BIT WAS A MARK
2365 015216 3$ INC R3 ; NEXT BIT
2366 015216 005203 CMP #10,R3 ; DONE YET?
2367 015220 022703 000010 BNE 1$ ; BR IF NO
2368 015224 001357 DATACLK, 14 ; CLOCK TRANSMITTER 14 MORE TICKS
2369 015226 104415 000014 ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2370 015232 104414 021264 ; PORT4+LU-13
2371 015234 021264 BIT #BITS,4(R1) ; RTS SHOULD BE CLEAR NOW
2372 015236 032761 000040 000004 BEQ 4$ ; BR IF YES
2373 015244 001401 HLT 34 ; ERROR, RTS NOT CLEAR
2374 015246 104034 4$ SCOPE ; SCOPE THIS TEST
2375 015250 104400
2376
2377 ;***** TEST 23 *****
2378 ;*DDCMP TRANSMITTER TEST
2379 ;*SINGLE CLOCK A BINARY COUNT PATTERN
2380 ;*VERIFY EACH BIT POSITION AS IT
2381 ;*PASSES THE BIT WINDOW (SI BIT)
2382 ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2383 ;*AND R5 CONTAINS THE CHARACTER THAT FAILED
2384 ;*****
2385
2386 ; TEST 23
2387 -----
2388
2389 015252 012737 000023 001226 TST23: MOV #23,TSTNO
2390 015260 012737 015460 001216 MOV #TST24,NEXT

```

```

MSTCLR
MOV      #BIT11,(R1)
CLR      R3
CLR      R4
CLR      R5
JSR      PC,OUTRDY
MOV      #1,4(R1)
ROMCLK
122111
JSR      PC,OUTRDY
MOV      R4,4(R1)
ROMCLK
122110
INC      R4
JSR      PC,OUTRDY
MOV      R4,4(R1)
ROMCLK
122110
JSR      PC,OCOR
45:      DATACLK, 2
CLR      R3
MOV      R5,R2
15:      DATACLK, 1
RORB     R2
BCC      25
JSR      PC,GETSI
BCS      35
HLT      6
25:      BR      35
JSR      PC,GETSI
BCC      35
35:      HLT      6
INC      R3
CMP      #10,R3
BNE      15
INC      R4
JSR      PC,OUTRDY
MOV      R4,4(R1)
ROMCLK
122110
INC      R5
CMP      #400,R5
BNE      45
55:      SCOPE

```

```

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SET LINE UNIT LOOP
;R3 CONTAINS BIT COUNT
;R4 CONTAINS CHAR TO BE LOADED IN SILO
;R5 CONTAINS CHARACTER CURRENTLY BEING SHIFTED C
;WAIT FOR OUT-READY
;SET BIT0 IN PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET SOM!
;WAIT FOR OUT-READY
;LOAD PORT4 WITH CHARACTER
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD OUT DATA
;INCREMENT TO NEXT CHARACTER
;WAIT FOR OUT-READY
;LOAD PORT4 WITH CHARACTER
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD OUT DATA
;WAIT FOR OCOR TO SET
;2 TICKS TO SET UP TRANSMITTER
;CLEAR BIT COUNTER
;LOAD CHARACTER IN R2
;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
;SHIFT NEXT SOFTWARE BIT IN TO CARRY
;BR IF CARRY CLEAR
;GET THE WINDOW
;BR IF BIT IS A MARK
;ERROR BIT WAS A SPACE
;CONTINUE WITH TEST
;GET THE WINDOW
;BR IF BIT IS A SPACE
;ERROR BIT WAS A MARK

;NEXT BIT
;DONE YET?
;BR IF NO
;NEXT CHARACTER
;WAIT FOR OUT-READY
;LOAD PORT4 WITH CHARACTER
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD OUT DATA
;NEXT CHARACTER
;DONE YET?
;BR IF NO
;SCOPE THIS TEST

```

```

***** TEST 24 *****
; *DDCMP STRIP SYNC TEST
; *SET LU LOOP, SINGLE STEP 5 SYNCs
; *VERIFY THAT IN ACTIVE DOES NOT SET
; *****

```

: TEST 24
:-----

BASIC RECEIVER TESTS

```

TST24:      MOV      #24,TSTNO
             MOV      #TST25,NEXT

             MSTCLR
             MOV      #BIT11,(R1)
             MOV      #12,R2
             JSR      PC,SYNC
             S
             DATACLK,          54
             ROMCLK
             021244
             MOV      4(R1),R4
             BIC      #277,R4
             CLR      R5
             CMPB     R5,R4
             BEQ      1$
             HLT      40
1$:          SCOPE

```

```
;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SET LU LOOP
;SAVE LU REG FOR TYPEOUT
;SINGLE CLOCK 5 SYNC CHARACTERS

;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4←LUI2
;PUT "FOUND" IN R4
;CLEAR UNWANTED BITS
;PUT "EXPECTED" IN R5
;IS ACTIVE CLEAR?
;BR IF YES
;ERROR ACTIVE IS NOT CLEAR
;SCOPE THIS TEST
```

```

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

```

: TEST 25

```

TST25:  MOV      #25, TSTNO
        MOV      #TST26, NEXT

        MSTCLR
        MOV      #BIT11, (R1)
        MOV      #12, R2
        JSR      PC, SYNC
        S
        DATA CLK,          64
        ROM CLK
        021244
        MOV      4(R1), R4
        BIC      #277, R4
        MOV      #BIT6, R5
        CMPB     R5, R4
        BEQ      1$
        HLT      40
1$:      SCOPE

```

```

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SET LU LOOP
;SAVE LU REG FOR TYPEOUT
;SINGLE CLOCK 5 SYNC CHARACTERS

;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4+LU12
;PUT "FOUND" IN R4
;CLEAR UNWANTED BITS
;PUT "EXPECTED" IN R5
;IS ACTIVE SET?
;BR IF YES
;ERROR ACTIVE IS NOT SET
;SCOPE THIS TEST

```

```

***** 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

.....

```

2503 015636 012737 000026 001226 TST26: MOV #26,TSTNO
2504 015644 012737 015724 001216 MOV #TST27,NEXT
2505 ;R1 CONTAINS BASE DMC11 ADDRESS
2506 015652 104412 MSTCLR ;MASTER CLEAR DMC11
2507 015654 012711 004000 MOV #BIT11,(R1) ;SET LU LOOP
2508 015660 012702 000012 MOV #12,R2 ;SAVE LU REG FOR TYPEOUT
2509 015664 004737 026366 JSR PC,SYNC ;SINGLE CLOCK 1 SYNC CHARACTERS
2510 015670 000001 1 DATACLK, 24
2511 015672 104415 000024 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2512 015676 104414 021244 ;PORT4+LUI2
2513 015700 021244 MOV 4(R1),R4 ;PUT "FOUND" IN R4
2514 015702 016104 000004 BIC #277,R4 ;CLEAR UNWANTED BITS
2515 015706 042704 000277 CLR R5 ;PUT "EXPECTED" IN R5
2516 015712 005005 CMPB R5,R4 ;IS ACTIVE CLEAR?
2517 015714 120504 BEQ 1$ ;BR IF YES
2518 015716 001401 HLT 40 ;ERROR ACTIVE IS NOT CLEAR
2519 015720 104040 1$: SCOPE ;SCOPE THIS TEST
2520 015722 104400
2521
2522
2523 ;***** TEST 27 *****
2524 ;*DDCMP IN ACTIVE TEST
2525 ;*SET LU LOOP, SINGLE STEP 2 SYNCs AND A NON-SYNC (301)
2526 ;*VERIFY THAT IN ACTIVE IS SET
2527 ;*****
2528
2529 : TEST 27
2530 -----
2531 015724 012737 000027 001226 TST27: MOV #27,TSTNO
2532 015732 012737 016014 001216 MOV #TST30,NEXT
2533 ;R1 CONTAINS BASE DMC11 ADDRESS
2534 015740 104412 MSTCLR ;MASTER CLEAR DMC11
2535 015742 012711 004000 MOV #BIT11,(R1) ;SET LU LOOP
2536 015746 012702 000012 MOV #12,R2 ;SAVE LU REG FOR TYPEOUT
2537 015752 004737 026366 JSR PC,SYNC ;SINGLE CLOCK 2 SYNC CHARACTERS
2538 015756 000002 2 DATACLK, 34
2539 015760 104415 000034 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2540 015764 104414 021244 ;PORT4+LUI2
2541 015766 021244 MOV 4(R1),R4 ;PUT "FOUND" IN R4
2542 015770 016104 000004 BIC #277,R4 ;CLEAR UNWANTED BITS
2543 015774 042704 000277 MOV #BIT6,R5 ;PUT "EXPECTED" IN R5
2544 016000 012705 000100 CMPB R5,R4 ;IS ACTIVE SET?
2545 016004 120504 BEQ 1$ ;BR IF YES
2546 016006 001401 HLT 40 ;ERROR ACTIVE IS NOT SET
2547 016010 104040 1$: SCOPE ;SCOPE THIS TEST
2548 016012 104400
2549
2550
2551 ;***** TEST 30 *****
2552 ;*IN CLEAR TEST
2553 ;*SYNC UP RECEIVER AND TRANSMIT A CHARACTER
2554 ;*WAIT FOR IN RDY, THEN SET IN CLEAR
2555 ;*VERIFY THAT IN ACTIVE AND IN RDY ARE CLEARED
2556 ;*****
2557
2558 : TEST 30

```

```

2559
2560 016014 012737 000030 001226      TST30:  MOV      #30,TSTNO
2561 016022 012737 016166 001216      MOV      #TST31,NEXT
2562
2563 016030 104412                     MSTCLR
2564 016032 012702 000012             MOV      #12,R2
2565 016036 012711 004000             MOV      #BIT11,(R1)
2566 016042 004737 026534             JSR      PC,CHAR
2567 016046 000301                     301
2568 016050 104415 000053             DATACLK, 53
2569 016054 104416 000002             TIMER, 2
2570 016060 104414                     ROMCLK
2571 016062 021244                     021244
2572 016064 016104 000004             MOV      4(R1),R4
2573 016070 042704 000357             BIC      #357,R4
2574 016074 012705 000020             MOV      #BIT4,R5
2575 016100 120504                     CMPB     R5,R4
2576 016102 001401                     BEQ      1$
2577 016104 104040                     HLT      40
2578 016106
2579 016106 012761 000200 000004      1$:  MOV      #BIT7,4(R1)
2580 016114 104414                     ROMCLK
2581 016116 122112                     122112
2582 016120 104414                     ROMCLK
2583 016122 021244                     021244
2584 016124 016104 000004             MOV      4(R1),R4
2585 016130 042704 000277             BIC      #277,R4
2586 016134 005005                     CLR      R5
2587 016136 120504                     CMPB     R5,R4
2588 016140 001401                     BEQ      2$
2589 016142 104040                     HLT      40
2590 016144
2591 016144 016104 000004             2$:  MOV      4(R1),R4
2592 016150 042704 000357             BIC      #357,R4
2593 016154 005005                     CLR      R5
2594 016156 120504                     CMPB     R5,R4
2595 016160 001401                     BEQ      3$
2596 016162 104040                     HLT      40
2597 016164 104400                     3$:  SCOPE
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608 016166 012737 000031 001226      TST31:  MOV      #31,TSTNO
2609 016174 012737 016302 001216      MOV      #TST32,NEXT
2610
2611 016202 104412                     MSTCLR
2612 016204 012702 000012             MOV      #12,R2
2613 016210 012711 004000             MOV      #BIT11,(R1)
2614 016214 004737 026534             JSR      PC,CHAR

```

;R1 CONTAINS BASE DMC11 ADDRESS
 ;MASTER CLEAR DMC11
 ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
 ;SET LINE UNIT LOOP
 ;LOAD SILO WITH 3 SYNC
 ;AND A NON-SYNC (301)
 ;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
 ;LOAD PORT4
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;SET IN CLEAR
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;PORT4+LU 12
 ;PUT "FOUND" IN R4
 ;CLEAR UNWANTED BITS
 ;PUT "EXPECTED" IN R5
 ;IS IN ACTIVE CLEAR?
 ;ERROR, IN ACTIVE IS NOT CLEAR
 ;PUT "FOUND" IN R4
 ;CLEAR UNWANTED BITS
 ;PUT "EXPECTED" IN R5
 ;IS INRDY CLEARED?
 ;ERROR, INRDY IS NOT CLEARED
 ;SCOPE THIS TEST

```

;***** TEST 31 *****
;DDCMP BASIC RECEIVER TEST
;SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 0
;VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
;*****
; TEST 31
;-----
TST31:  MOV      #31,TSTNO
        MOV      #TST32,NEXT

        MSTCLR
        MOV      #12,R2
        MOV      #BIT11,(R1)
        JSR      PC,CHAR

```

;R1 CONTAINS BASE DMC11 ADDRESS
 ;MASTER CLEAR DMC11
 ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
 ;SET LINE UNIT LOOP
 ;LOAD SILO WITH 3 SYNC

Address	Hex	Dec	Label	Instruction	Comment
2615	016220	000000			:AND THE CHARACTER 0
2616	016222	104415	000053	DATACLK, 53	:SINGLE CLOCK THE DATA
2617	016226	104416	000002	TIMER, 2	:WAIT FOR INRDY
2618	016232	104414		ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2619	016234	021244		021244	:PORT4←LU 12
2620	016236	016104	000004	MOV 4(R1),R4	:PUT "FOUND" IN R4
2621	016242	042704	000357	BIC #357,R4	:CLEAR UNWANTED BITS
2622	016246	012705	000020	MOV #BIT4,R5	:PUT "EXPECTED" IN R5
2623	016252	120504		CMPB R5,R4	:IS INRDY SET?
2624	016254	001401		BEQ 1\$	
2625	016256	104040		HLT 40	:ERROR, INRDY IS NOT SET
2626	016260		1\$:		
2627	016260	104414		ROMCLK	:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2628	016262	021204		021204	:PORT4←IN DATA
2629	016264	016104	000004	MOV 4(R1),R4	:PUT "FOUND" IN R4
2630	016270	005005		CLR R5	:PUT "EXPECTED" IN R5
2631	016272	120504		CMPB R5,R4	:WAS A 0 RECEIVED?
2632	016274	001401		BEQ 2\$	
2633	016276	104010		HLT 10	:ERROR, RECEIVED DATA IS WRONG
2634	016300	104400	2\$:	SCOPE	:SCOPE THIS TEST
2635					
2636					
2637					
2638					:***** TEST 32 *****
2639					:*DDCMP BASIC RECEICER TEST
2640					:*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 125
2641					:*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
2642					:*****
2643					
2644					: TEST 32
2645	016302	012737	000032	001226	TST32: MOV #32,TSTNO
2646	016310	012737	016420	001216	MOV #TST33,NEXT
2647					
2648	016316	104412			:R1 CONTAINS BASE DMC11 ADDRESS
2649	016320	012702	000012		:MASTER CLEAR DMC11
2650	016324	012711	004000		:SAVE REG ADDRESS IN R2 FOR TYPEOUT
2651	016330	004737	026534		:SET LINE UNIT LOOP
2652	016334	000125			:LOAD SILO WITH 3 SYNC
2653	016336	104415	000053		:AND THE CHARACTER 125
2654	016342	104416	000002		:SINGLE CLOCK THE DATA
2655	016346	104414			:WAIT FOR INRDY
2656	016350	021244			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2657	016352	016104	000004		:PORT4←LU 12
2658	016356	042704	000357		:PUT "FOUND" IN R4
2659	016362	012705	000020		:CLEAR UNWANTED BITS
2660	016366	120504			:PUT "EXPECTED" IN R5
2661	016370	001401			:IS INRDY SET?
2662	016372	104040			
2663	016374		1\$:		:ERROR, INRDY IS NOT SET
2664	016374	104414			
2665	016376	021204			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2666	016400	016104	000004		:PORT4←IN DATA
2667	016404	012705	000125		:PUT "FOUND" IN R4
2668	016410	120504			:PUT "EXPECTED" IN R5
2669	016412	001401			:WAS A 125 RECEIVED?
2670	016414	104010			:ERROR, RECEIVED DATA IS WRONG

```

2671 016416 104400          25: SCOPE          ;SCOPE THIS TEST
2672
2673
2674          ;***** TEST 33 *****
2675          ;DDCMP BASIC RECEICER TEST
2676          ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 252
2677          ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
2678          ;*****
2679
2680          : TEST 33
2681          :-----
2682 016420 012737 000033 001226      TST33: MOV      #33,TSTNO
2683 016426 012737 016536 001216      MOV      #TST34,NEXT
2684
2685          MSTCLR                      ;R1 CONTAINS BASE DMC11 ADDRESS
2686 016436 012702 000012              MOV      #12,R2          ;MASTER CLEAR DMC11
2687 016442 012711 004000              MOV      #BIT11,(R1)      ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
2688 016446 004737 026534              JSR      PC,CHAR        ;SET LINE UNIT LOOP
2689 016452 000252                      252          ;LOAD SILO WITH 3 SYNCs
2690 016454 104415 000053              DATACLK, 53          ;AND THE CHARACTER 252
2691 016460 104416 000002              TIMER, 2             ;SINGLE CLOCK THE DATA
2692 016464 104414                      ROMCLK              ;WAIT FOR INRDY
2693 016466 021244 021244              021244          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2694 016470 016104 000004              MOV      4(R1),R4        ;PORT4*LU 12
2695 016474 042704 000357              BIC      #357,R4        ;PUT "FOUND" IN R4
2696 016500 012705 000020              MOV      #BIT4,R5        ;CLEAR UNWANTED BITS
2697 016504 120504                      CMPB      R5,R4          ;PUT "EXPECTED" IN R5
2698 016506 001401                      BEQ       1$            ;IS INRDY SET?
2699 016510 104040                      HLT       40            ;ERROR, INRDY IS NOT SET
2700 016512
2701 016512 104414          1$: ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2702 016514 021204          021204          ;PORT4*IN DATA
2703 016516 016104 000004              MOV      4(R1),R4        ;PUT "FOUND" IN R4
2704 016522 012705 000252              MOV      #252,R5        ;PUT "EXPECTED" IN R5
2705 016526 120504                      CMPB      R5,R4          ;WAS A 252 RECEIVED?
2706 016530 001401                      BEQ       2$            ;
2707 016532 104010                      HLT       10            ;ERROR, RECEIVED DATA IS WRONG
2708 016534 104400          2$: SCOPE          ;SCOPE THIS TEST
2709
2710
2711          ;***** TEST 34 *****
2712          ;DDCMP BASIC RECEICER TEST
2713          ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 377
2714          ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
2715          ;*****
2716
2717          : TEST 34
2718          :-----
2719 016536 012737 000034 001226      TST34: MOV      #34,TSTNO
2720 016544 012737 016654 001216      MOV      #TST35,NEXT
2721
2722          MSTCLR                      ;R1 CONTAINS BASE DMC11 ADDRESS
2723 016554 012702 000012              MOV      #12,R2          ;MASTER CLEAR DMC11
2724 016560 012711 004000              MOV      #BIT11,(R1)      ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
2725 016564 004737 026534              JSR      PC,CHAR        ;SET LINE UNIT LOOP
2726 016570 000377                      377          ;LOAD SILO WITH 3 SYNCs
                ;AND THE CHARACTER 377

```

Address	Hex	Dec	Label	Instruction	Comment
2727	016572	104415	000053		
2728	016576	104416	000002		
2729	016602	104414			
2730	016604	021244			
2731	016606	016104	000004		
2732	016612	042704	000357		
2733	016616	012705	000020		
2734	016622	120504			
2735	016624	001401			
2736	016626	104040			
2737	016630				
2738	016630	104414			
2739	016632	021204			
2740	016634	016104	000004		
2741	016640	012705	000377		
2742	016644	120504			
2743	016646	001401			
2744	016650	104010			
2745	016652	104400			
2746					
2747					
2748					
2749					
2750					
2751					
2752					
2753					
2754					
2755					
2756	016654	012737	000035	001226	
2757	016662	012737	017004	001216	
2758					
2759	016670	104412			
2760	016672	005037	027152		
2761	016676	005037	027154		
2762	016702	005002			
2763	016704	012703	000073		
2764	016710	012711	004000		
2765	016714	004737	026712		
2766	016720	104415	000043		
2767	016724	104415	000730		
2768	016730	004737	027156		
2769	016734	104414			
2770	016736	021204			
2771	016740	016104	000004		
2772	016744	010205			
2773	016746	120504			
2774	016750	001401			
2775	016752	104010			
2776	016754	005202			
2777	016756	022702	000400		
2778	016762	001407			
2779	016764	005303			
2780	016766	001360			
2781	016770	004737	026712		
2782	016774	012703	000073		

```

2783 017000 000751          BR      1$          ;CONTINUE
2784 017002 104400          3$:    SCOPE          ;SCOPE THIS TEST
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797 017004 012737 000036 001226          TST36:  MOV      #36,TSTNO
2798 017012 012737 017144 001216          MOV      #TST37,NEXT
2799
2800 017020 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
2801 017022 005037 027152          CLR      SCHAR          ;MASTER CLEAR DMC11
2802 017026 005037 027154          CLR      STUFLG          ;START BINARY COUNT AT ZERO
2803 017032 005002          CLR      R2          ;CLEAR BITSTUFF FLAG
2804 017034 012703 000073          MOV      #73,R3          ;R2 IS "EXPECTED" DATA
2805 017040 005011          CLR      (R1)          ;R3 IS CHARACTER COUNT
2806 017042 012761 000040 000004          MOV      #BIT5,4(R1) ;CLEAR LU LOOP IN MAINT REG
2807 017050 104414          ROMCLK          ;LOAD PORT4
2808 017052 122112          122112          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2809 017054 004737 026712          JSR      PC,SILOLD          ;SET LU LOOP IN LU REG 12
2810 017060 104415 000043          DATACLK, 43          ;LOAD SILO WITH COUNT PATTERN
2811 017064 104415 000730          DATACLK, 730          ;SYNC RECEIVER AND GET IT ACTIVE
2812 017070 004737 027156          4$:    JSR      PC,INRDY          ;CLOCK IN 73 CHARACTERS
2813 017074 104414          ROMCLK          ;WAIT FOR INRDY
2814 017076 021204          021204          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2815 017100 016104 000004          MOV      4(R1),R4          ;PORT4+IN DATA
2816 017104 010205          MOV      R2,R5          ;PUT "FOUND" IN R4
2817 017106 120504          CMPB      R5,R4          ;PUT "EXPECTED" IN R5
2818 017110 001401          BEQ      2$          ;IS DATA CORRECT?
2819 017112 104010          HLT      10          ;BR IF YES
2820 017114 005202          2$:    INC      R2          ;DATA ERROR
2821 017116 022702 J00400          CMP      #400,R2          ;NEXT CHARACTER
2822 017122 001407          BEQ      3$          ;ALL DONE?
2823 017124 005303          DEC      R3          ;BR IF YES
2824 017126 001360          BNE      4$          ;DECREMENT CHARACTER COUNT
2825 017130 004737 026712          JSR      PC,SILOLD          ;BR IF SILO NOT EMPTY
2826 017134 012703 000073          MOV      #73,R3          ;LOAD SILO WITH MORE OF COUNT PATTERN
2827 017140 000751          BR      1$          ;RELOAD CHARACTER COUNT
2828 017142 104400          3$:    SCOPE          ;CONTINUE
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838

```

***** TEST 36 *****

*DDCMP DATA TEST

*THIS TEST SINGLE STEPS A BINARY COUNT PATTERN

*CHECKING EACH CHARACTER AS IT IS RECEIVED

*THIS TEST IS EXACTLY THE SAME AS THE LAST TEST.

*EXCEPT LINE UNIT LOOP IS SET IN LU REGISTER 12

TEST 36

***** TEST 37 *****

*TRANSMITTER MARK TEST

*SINGLE CLOCK 3 SYNCs AND A 301 AND 20 EXTRA

*CLOCK TICKS, VERIFY THAT A 301, A 377 AND A 377

*WERE RECEIVED INDICATING THAT THE TRANSMITTER WENT

*TO A MARK STATE FOR 16 BITS WHEN OUT SILO WAS EMPTY

```

2839          : TEST 37
2840          :-----
2841 017144 012737 000037 001226      TST37: MOV      #37,TSTNO
2842 017152 012737 017304 001216      MOV      #TST40,NEXT
2843          :
2844 017160 104412          MSTCLR          :R1 CONTAINS BASE DMC11 ADDRESS
2845 017162 012711 004000      MOV      #BIT11,(R1) :MASTER CLEAR DMC11
2846 017166 004737 026534      JSR      PC,CHAR :SET LINE UNIT LOOP
2847 017172 000301          301          :LOAD SILO WITH 3 SYNCs
2848 017174 104415 000073      DATACLK, 73 :AND A 301
2849 017200 004737 027156      JSR      PC,INRDY :CLOCK THE 301 IN AND 20 EXTRA TICKS
2850 017204 104414          ROMCLK          :WAIT FOR INRDY
2851 017206 021204          021204      :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2852 017210 016104 000004      MOV      4(R1),R4 :PORT4<IN DATA
2853 017214 012705 000301      MOV      #301,R5 :PUT "FOUND" IN R4
2854 017220 120504          CMPB      R5,R4 :PUT "EXPECTED" IN R5
2855 017222 001401          BEQ      1$ :WAS A 301 RECEIVED?
2856 017224 104010          HLT      10 :
2857 017226 004737 027156      1$: JSR      PC,INRDY :ERROR FIRST CHARACTER INCORRECT
2858 017232 104414          ROMCLK          :WAIT FOR INRDY
2859 017234 021204          021204      :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2860 017236 016104 000004      MOV      4(R1),R4 :PORT4<IN DATA
2861 017242 012705 000377      MOV      #377,R5 :PUT "FOUND" IN R4
2862 017246 120504          CMPB      R5,R4 :PUT "EXPECTED" IN R5
2863 017250 001401          BEQ      2$ :WAS A 377 RECEIVED?
2864 017252 104010          HLT      10 :
2865 017254 004737 027156      2$: JSR      PC,INRDY :ERROR, 377 WAS NOT RECEIVED
2866 017260 104414          ROMCLK          :WAIT FOR INRDY
2867 017262 021204          021204      :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2868 017264 016104 000004      MOV      4(R1),R4 :PORT4<IN DATA
2869 017270 012705 000377      MOV      #377,R5 :PUT "FOUND" IN R4
2870 017274 120504          CMPB      R5,R4 :PUT "EXPECTED" IN R5
2871 017276 001401          BEQ      3$ :WAS A 377 RECEIVED?
2872 017300 104010          HLT      10 :
2873 017302 104400          3$: SCOPE :ERROR, 377 WAS NOT RECEIVED
2874          :SCOPE THIS TEST
2875
2876          :***** TEST 40 *****
2877          :*CABLE TURNAROUND TEST
2878          :*CLEAR LINE UNIT LOOP, SET DTR
2879          :*VERIFY THAT MODEM READY IS SET
2880          :*CLEAR DTR, VERIFY THAT MRDY IS CLEARED
2881          :*****
2882
2883          : TEST 40
2884          :-----
2885 017304 012737 000040 001226      TST40: MOV      #40,TSTNO
2886 017312 012737 017466 001216      MOV      #TST41,NEXT
2887          :
2888 017320 104412          MSTCLR          :R1 CONTAINS BASE DMC11 ADDRESS
2889 017322 032737 020000      BIT      #BIT13,STAT1 :MASTER CLEAR DMC11
2890 017330 001004          BNE      .+12 :IS LINE UNIT M8202?
2891 017332 032737 040000      BIT      #BIT14,STAT1 :IS LINE UNIT M8202?
2892 017340 001451          BEQ      2$ :BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
2893 017342 005011          CLR      (R1) :IS TURNAROUND CONNECTOR ON?
2894 017344 012761 000100 000004      MOV      #100,4(R1) :SKIP TEST IF NO
2895          :CLEAR LINE UNIT LOOP
2896          :LOAD PORT4

```

2895	017352	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2896	017354	122113			122113		;SET DTR
2897	017356	104416	000002		TIMER,	2	;WAIT
2898	017362	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2899	017364	021264			021264		;PORT4+LU13
2900	017366	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
2901	017372	042704	000223		BIC	#223,R4	;CLEAR UNWANTED BITS
2902	017376	012705	000110		MOV	#110,R5	;PUT "EXPECTED" IN R5
2903	017402	120504			CMPB	R5,R4	;IS MRDY SET?
2904	017404	001401			BEQ	1\$	
2905	017406	104011			HLT	11	;ERROR, MRDY NOT SET
2906	017410	005061	000004		CLR	4(R1)	;CLEAR PORT4
2907	017414	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2908	017416	122113			122113		;CLEAR DTR
2909	017420	104416	000002		TIMER,	2	
2910	017424	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2911	017426	021264			021264		;PORT4+LU13
2912	017430	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
2913	017434	042704	000223		BIC	#223,R4	;CLEAR UNWANTED BITS
2914	017440	005005			CLR	R5	;PUT "EXPECTED" IN R5
2915	017442	032737	020000	001366	BIT	#BIT13,STAT1	;IS LINE UNIT M8202?
2916	017450	001402			BEQ	+.6	;BR IF NO
2917	017452	052705	000010		BIS	#BIT3,R5	;MRDY SET ON M8202
2918	017456	120504			CMPB	R5,R4	;IS MRDY CLEAR?
2919	017460	001401			BEQ	2\$	
2920	017462	104011			HLT	11	;ERROR, MRDY NOT CLEAR
2921	017464	104400			SCOPE		;SCOPE THIS TEST
2922							
2923							
2924							
2925							
2926							
2927							
2928							
2929							
2930							
2931							
2932	017466	012737	000041	001226	TST41: MOV	#41,TSTNO	
2933	017474	012737	017632	001216	MOV	#TST42,NEXT	
2934							
2935	017502	104412			MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
2936	017504	032737	020000	001366	BIT	#BIT13,STAT1	;MASTER CLEAR DMC11
2937	017512	001004			BNE	+.12	;IS LINE UNIT M8202?
2938	017514	032737	040000	001366	BIT	#BIT14,STAT1	;BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
2939	017522	001442			BEQ	1\$	;IS TURNAROUND CONNECTOR ON?
2940	017524	012711	004000		MOV	#BIT11,(R1)	;SKIP TEST IF NO
2941	017530	012761	000100	000004	MOV	#100, 4(R1)	;SET LINE UNIT LOOP
2942	017536	104414			ROMCLK		;LOAD PORT4
2943	017540	122113			122113		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2944	017542	104416	000002		TIMER,	2	;CLEAR ALL MODEM SIGNALS, EXCEPT DTR
2945	017546	012761	000001	000004	MOV	#1,4(R1)	;WAIT
2946	017554	104414			ROMCLK		;LOAD PORT4
2947	017556	122111			122111		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2948	017560	004537	027620		JSR	R5,MESLD	;SET SOM
2949	017564	030102			MESDAT		;FILL OUT DATA SILO
2950	017566	000100			64.		;WITH 64 CHARACTERS

***** TEST 41 *****
;CABLE TURNAROUND TEST
;CLEAR LINE UNIT LOOP, LOAD OUT DATA SILO
;VERIFY THAT ALL MODEM SIGNALS ARE SET

; TEST 41

2951	017570	012700	000050		MOV	#50,R0	;PREPARE FOR DELAY
2952	017574	005011			CLR	(R1)	;CLEAR LINE UNIT LOOP
2953	017576			2\$:			
2954	017576	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2955	017600	021264			021264		;PORT4+LU13
2956	017602	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
2957	017606	042704	000223		BIC	#223,R4	;CLEAR UNWANTED BITS
2958	017612	012705	000154		MOV	#154,R5	;PUT "EXPECTED" IN R5
2959	017616	120504			CMPB	R5,R4	;COMPARE EXPECTED AND FOUND
2960	017620	001403			BEQ	1\$	;BR IF OK
2961	017622	005300			DEC	R0	;DEC DELAY COUNT
2962	017624	001364			BNE	2\$	;BR IF NOT ZERO
2963	017626	104011			HLT	11	;ERROR, ALL SIGNALS ARE NOT SET
2964	017630	104400		1\$:	SCOPE		;SCOPE THIS TEST
2965							
2966							
2967							
2968							
2969							
2970							
2971							
2972							
2973							
2974							
2975							
2976	017632	012737	000042	001226	TST42:	MOV	#42,TSTNO
2977	017640	012737	020146	001216		MOV	#TST43,NEXT
2978	017646	012737	017662	001220		MOV	#64\$,\$LOCK
2979							
2980	017654	104412				MSTCLR	
2981	017656	012711	004000			MOV	#BIT11,(R1)
2982	017662	004737	027662		64\$:	JSR	PC,CLR10
2983	017666	005000				CLR	R0
2984	017670	012737	120001	027316		MOV	#CRC16,XPOLY
2985	017676	012737	000000	017736		MOV	#0,66\$
2986	017704	005037	017740			CLR	67\$
2987	017710	004737	027322			JSR	PC,BCCLD
2988	017714	000000				0	
2989	017716	104415	000021			DATACLK,	21
2990	017722	104415	000001		65\$:	DATACLK,	1
2991	017726	005200				INC	R0
2992	017730	004537	027212			JSR	R5,SIMBCC
2993	017734	000001				1	
2994	017736	000000			66\$:	0	
2995	017740	000000			67\$:	0	
2996	017742	103405				BCS	68\$
2997	017744	004737	027434			JSR	PC,GETQ0
2998	017750	103006				BCC	69\$
2999	017752	104012				HLT	12
3000	017754	000404				BR	69\$
3001	017756	004737	027434		68\$:	JSR	PC,GETQ0
3002	017762	103401				BCS	69\$
3003	017764	104016				HLT	16
3004	017766				69\$:		
3005	017766	006037	017736			ROR	66\$
3006	017772	013737	027320	017740		MOV	CALBCC,67\$

3007	020000	022700	000010		CMP	#10,R0	; DONE YET?
3008	020004	001346			BNE	65\$	; BR IF NOT DONE
3009	020006	104401			SCOPE1		; SCOPE SUBTEST (SW09=1)
3010	020010	012737	020016	001220	MOV	#71\$,LOCK	; NEW SCOPE1
3011	020016	004737	027662		JSR	PC,CLR10	; CLEAR BCC REGISTERS
3012	020022	005000			CLR	R0	; START SHIFT COUNTER AT ZERO
3013	020024	012737	120001	027316	MOV	#CRC16,XPOLY	; LOAD POLYNOMIAL FOR SOFTWARE BCC
3014	020032	012737	000000	020072	MOV	#0,73\$	; LOAD CHAR FOR SOFTWARE BCC
3015	020040	005037	020074		CLR	74\$	; CLEAR OLD SOFTWARE BCC
3016	020044	004737	027322		JSR	PC,BCCLD	; LOAD OUT SILO WITH 2 SYNC
3017	020050	000000					; AND THE CHARACTER 0
3018	020052	104415	000032		DATACLK,	32	; GET RECEIVER ACTIVE
3019	020056	104415	000001		DATACLK,	1	; SHIFT BCC ONCE
3020	020062	005200			INC	R0	; BUMP SHIFT COUNT
3021	020064	004537	027212		JSR	R5,SIMBCC	; CALCULATE SOFTWARE BCC LSB
3022	020070	000001					; ONE SHIFT
3023	020072	000000					; DATA CHARACTER
3024	020074	000000					; OLD BCC
3025	020076	103405			BCS	75\$	; BR IF SOFT BCC LSB IS SET
3026	020100	004737	027446		JSR	PC,GETQI	; GET HARDWARE RECEIVER BCC LSB
3027	020104	103006			BCC	76\$	; BR IF HARD BCC LSB IS CLEAR
3028	020106	104013			HLT	13	; ERROR, BCC LSB IS SET
3029	020110	000404			BR	76\$	; CONTINUE
3030	020112	004737	027446		JSR	PC,GETQI	; GET HARDWARE RECEIVER BCC LSB
3031	020116	103401			BCS	76\$	; BR IF HARD BCC LSB IS SET
3032	020120	104017			HLT	17	; ERROR, BCC LSB IS CLEAR
3033	020122						
3034	020122	006037	020072		ROR	73\$	; SHIFT SOFT DATA
3035	020126	013737	027320	020074	MOV	CALBCC,7-5	; LOAD OLD SOFT BCC
3036	020134	022700	000010		CMP	#10,R0	; DONE YET?
3037	020140	001346			BNE	72\$	; BR IF NOT DONE
3038	020142	104401			SCOPE1		; SCOPE SUBTEST (SW09=1)
3039	020144	104400			SCOPE		; SCOPE THIS TEST
3040							
3041							
3042							
3043							
3044							
3045							
3046							
3047							
3048							
3049							
3050							
3051	020146	012737	000043	001226	TST43:	MOV	#43,TSTNO
3052	020154	012737	020462	001216		MOV	#TST44,NEXT
3053	020162	012737	020176	001220		MOV	#64\$,LOCK
3054							
3055	020170	104412			MSTCLR		; R1 CONTAINS BASE DMC11 ADDRESS
3056	020172	012711	004000		MOV	#BIT11,(R1)	; MASTER CLEAR DMC11
3057	020176	004737	027662		JSR	PC,CLR10	; SET LU LOOP
3058	020202	005000			CLR	R0	; CLEAR BCC REGISTERS
3059	020204	012737	120001	027316	MOV	#CRC16,XPOLY	; START SHIFT COUNTER AT ZERO
3060	020212	012737	000377	020252	MOV	#377,66\$;	; LOAD POLYNOMIAL FOR SOFTWARE BCC
3061	020220	005037	020254		CLR	67\$	; LOAD CHAR FOR SOFTWARE BCC
3062	020224	004737	027322		JSR	PC,BCCLD	; CLEAR OLD SOFTWARE BCC
							; LOAD OUT SILO WITH 2 SYNC

***** 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

3063	020230	000377			377		: AND THE CHARACTER 377
3064	020232	104415	000021		DATACLK,	21	: GET TRANSMITTER ACTIVE
3065	020236	104415	000001	65\$:	DATACLK,	1	: SHIFT BCC ONCE
3066	020242	005200			INC	R0	: BUMP SHIFT COUNT
3067	020244	004537	027212		JSR	R5,SIMBCC	: CALCULATE SOFTWARE BCC LSB
3068	020250	000001			1		: ONE SHIFT
3069	020252	000000		66\$:	0		: DATA CHARACTER
3070	020254	000000		67\$:	0		: OLD BCC
3071	020256	103405			BCS	68\$	: BR IF SOFT BCC LSB IS SET
3072	020260	004737	027434		JSR	PC,GETQ0	: GET HARDWARE TRANSMITTER BCC LSB
3073	020264	103006			BCC	69\$	: BR IF HARD BCC LSB IS CLEAR
3074	020266	104012			HLT	12	: ERROR, BCC LSB IS SET
3075	020270	000404			BR	69\$	: CONTINUE
3076	020272	004737	027434	68\$:	JSR	PC,GETQ0	: GET HARDWARE TRANSMITTER BCC LSB
3077	020276	103401			BCS	69\$	: BR IF HARD BCC LSB IS SET
3078	020300	104016			HLT	16	: ERROR, HARD BCC LSB IS CLEAR
3079	020302			69\$:			
3080	020302	006037	020252		ROR	66\$	: SHIFT SOFT DATA
3081	020306	013737	027320	020254	MOV	CALBCC,67\$	: LOAD OLD SOFT BCC
3082	020314	022700	000010	.	CMP	#10,R0	: DONE YET?
3083	020320	001346			BNE	65\$	: BR IF NOT DONE
3084	020322	104401			SCOPE1		: SCOPE SUBTEST (SW09=1)
3085	020324	012737	020332	001220	MOV	#71\$,LOCK	: NEW SCOPE1
3086	020332	004737	027662	71\$: .	JSR	PC,CLRIO	: CLEAR BCC REGISTERS
3087	020336	005000			CLR	R0	: START SHIFT COUNTER AT ZERO
3088	020340	012737	120001	027316	MOV	#CRC16,XPOLY	: LOAD POLYNOMIAL FOR SOFTWARE BCC
3089	020346	012737	000377	020406	MOV	#377,73\$;	: LOAD CHAR FOR SOFTWARE BCC
3090	020354	005037	020410		CLR	74\$	: CLEAR OLD SOFTWARE BCC
3091	020360	004737	027322		JSR	PC,BCCLD	: LOAD OUT SILO WITH 2 SYNCS
3092	020364	000377			377		: AND THE CHARACTER 377
3093	020366	104415	000032		DATACLK,	32	: GET RECEIVER ACTIVE
3094	020372	104415	000001	72\$:	DATACLK,	1	: SHIFT BCC ONCE
3095	020376	005200			INC	R0	: BUMP SHIFT COUNT
3096	020400	004537	027212		JSR	R5,SIMBCC	: CALCULATE SOFTWARE BCC LSB
3097	020404	000001			1		: ONE SHIFT
3098	020406	000000		73\$:	0		: DATA CHARACTER
3099	020410	000000		74\$:	0		: OLD BCC
3100	020412	103405			BCS	75\$	: BR IF SOFT BCC LSB IS SET
3101	020414	004737	027446		JSR	PC,GETQI	: GET HARDWARE RECEIVER BCC LSB
3102	020420	103006			BCC	76\$	: BR IF HARD BCC LSB IS CLEAR
3103	020422	104013			HLT	13	: ERROR, BCC LSB IS SET
3104	020424	000404			BR	76\$	: CONTINUE
3105	020426	004737	027446	75\$:	JSR	PC,GETQI	: GET HARDWARE RECEIVER BCC LSB
3106	020432	103401			BCS	76\$	: BR IF HARD BCC LSB IS SET
3107	020434	104017			HLT	17	: ERROR, BCC LSB IS CLEAR
3108	020436			76\$:			
3109	020436	006037	020406		ROR	73\$	: SHIFT SOFT DATA
3110	020442	013737	027320	020410	MOV	CALBCC,74\$	: LOAD OLD SOFT BCC
3111	020450	022700	000010		CMP	#10,R0	: DONE YET?
3112	020454	001346			BNE	72\$	: BR IF NOT DONE
3113	020456	104401			SCOPE1		: SCOPE SUBTEST (SW09=1)
3114	020460	104400		77\$:	SCOPE		: SCOPE THIS TEST
3115							
3116							
3117							
3118							

***** TEST 44 *****
 : *TEST OF CRC OPERATION

```

***** TEST 44 *****
: *TEST OF CRC OPERATION

```

```

3119                                     : *USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3120                                     : *125, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3121                                     : *TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3122                                     : *****
3123                                     :
3124                                     : TEST 44
3125                                     : -----
3126 020462 012737 000044 001226      TST44: MOV      #44,TSTNO
3127 020470 012737 020776 001216      MOV      #TST45,NEXT
3128 020476 012737 020512 001220      MOV      #64$,LOCK
3129                                     :
3130 020504 104412                                     MSTCLR
3131 020506 012711 004000                                     MOV      #BIT11,(R1)
3132 020512 004737 027662      64$: JSR      PC,CLR10
3133 020516 005000                                     CLR      R0
3134 020520 012737 120001 027316      MOV      #CRC16,XPOLY
3135 020526 012737 000125 020566      MOV      #125,66$;
3136 020534 005037 020570                                     CLR      67$
3137 020540 004737 027322      JSR      PC,BCCLD
3138 020544 000125 125                                     :
3139 020546 104415 000021      DATACLK, 21
3140 020552 104415 000001      65$: DATACLK, 1
3141 020556 005200                                     INC      R0
3142 020560 004537 027212      JSR      R5,SIMBCC
3143 020564 000001                                     :
3144 020566 000000      66$: 0
3145 020570 000000      67$: 0
3146 020572 103405                                     BCS      68$
3147 020574 004737 027434      JSR      PC,GETQ0
3148 020600 103006                                     BCC      69$
3149 020602 104012                                     HLT      12
3150 020604 000404                                     BR      69$
3151 020606 004737 027434      68$: JSR      PC,GETQ0
3152 020612 103401                                     BCS      69$
3153 020614 104016                                     HLT      16
3154 020616                                     :
3155 020616 006037 020566      69$: ROR      66$
3156 020622 013737 027320 020570      MOV      CALBCC,67$
3157 020630 022700 000010      CMP      #10,R0
3158 020634 001346                                     BNE      65$
3159 020636 104401                                     SCOPE1
3160 020640 012737 020646 001220      MOV      #71$,LOCK
3161 020646 004737 027662      71$: JSR      PC,CLR10
3162 020652 005000                                     CLR      R0
3163 020654 012737 120001 027316      MOV      #CRC16,XPOLY
3164 020662 012737 000125 020722      MOV      #125,73$;
3165 020670 005037 020724                                     CLR      74$
3166 020674 004737 027322      JSR      PC,BCCLD
3167 020700 000125 125                                     :
3168 020702 104415 000032      DATACLK, 32
3169 020706 104415 000001      72$: DATACLK, 1
3170 020712 005200                                     INC      R0
3171 020714 004537 027212      JSR      R5,SIMBCC
3172 020720 000001                                     :
3173 020722 000000      73$: 0
3174 020724 000000      74$: 0

```

```

: R1 CONTAINS BASE DMC11 ADDRESS
: MASTER CLEAR DMC11
: SET LU LOOP
: CLEAR BCC REGISTERS
: START SHIFT COUNTER AT ZERO
: LOAD POLYNOMIAL FOR SOFTWARE BCC
: LOAD CHAR FOR SOFTWARE BCC
: CLEAR OLD SOFTWARE BCC
: LOAD OUT SILO WITH 2 SYNCs
: AND THE CHARACTER 125
: GET TRANSMITTER ACTIVE
: SHIFT BCC ONCE
: BUMP SHIFT COUNT
: CALCULATE SOFTWARE BCC LSB
: ONE SHIFT
: DATA CHARACTER
: OLD BCC
: BR IF SOFT BCC LSB IS SET
: GET HARDWARE TRANSMITTER BCC LSB
: BR IF HARD BCC LSB IS CLEAR
: ERROR, BCC LSB IS SET
: CONTINUE
: GET HARDWARE TRANSMITTER BCC LSB
: BR IF HARD BCC LSB IS SET
: ERROR, HARD BCC LSB IS CLEAR
:
: SHIFT SOFT DATA
: LOAD OLD SOFT BCC
: DONE YET?
: BR IF NOT DONE
: SCOPE SUBTEST (SW09=1)
: NEW SCOPE1
: CLEAR BCC REGISTERS
: START SHIFT COUNTER AT ZERO
: LOAD POLYNOMIAL FOR SOFTWARE BCC
: LOAD CHAR FOR SOFTWARE BCC
: CLEAR OLD SOFTWARE BCC
: LOAD OUT SILO WITH 2 SYNCs
: AND THE CHARACTER 125
: GET RECEIVER ACTIVE
: SHIFT BCC ONCE
: BUMP SHIFT COUNT
: CALCULATE SOFTWARE BCC LSB
: ONE SHIFT
: DATA CHARACTER
: OLD BCC

```

3175	020726	103405			BCC	75\$	:BR IF SOFT BCC LSB IS SET
3176	020730	004737	027446		JSR	PC,GETQI	:GET HARDWARE RECEIVER BCC LSB
3177	020734	103006			BCC	76\$	:BR IF HARD BCC LSB IS CLEAR
3178	020736	104013			HLT	13	:ERROR, BCC LSB IS SET
3179	020740	000404			BR	76\$	:CONTINUE
3180	020742	004737	027446	75\$:	JSR	PC,GETQI	:GET HARDWARE RECEIVER BCC LSB
3181	020746	103401			BCC	76\$	:BR IF HARD BCC LSB IS SET
3182	020750	104017			HLT	17	:ERROR, BCC LSB IS CLEAR
3183	020752			76\$:			
3184	020752	006037	020722		ROR	73\$	:SHIFT SOFT DATA
3185	020756	013737	027320	020724	MOV	CALBCC,74\$	:LOAD OLD SOFT BCC
3186	020764	022700	000010		CMP	#10,R0	:DONE YET?
3187	020770	001346			BNE	72\$	:BR IF NOT DONE
3188	020772	104401			SCOPI		:SCOPE SUBTEST (SW09=1)
3189	020774	104400			SCOPE		:SCOPE THIS TEST
3190							
3191							
3192							:***** TEST 45 *****
3193							:*TEST OF CRC OPERATION
3194							:*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3195							:*252, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3196							:*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3197							:*****
3198							
3199							: TEST 45
3200							:-----
3201	020776	012737	000045	001226	TST45:	MOV	#45,TSTNO
3202	021004	012737	021312	001216		MOV	#TST46,NEXT
3203	021012	012737	021026	001220		MOV	#64\$,LOCK
3204							:R1 CONTAINS BASE DMC11 ADDRESS
3205	021020	104412			MSTCLR		:MASTER CLEAR DMC11
3206	021022	012711	004000		MOV	#BIT11,(R1)	:SET LU LOOP
3207	021026	004737	027662		JSR	PC,CLR10	:CLEAR BCC REGISTERS
3208	021032	005000			CLR	R0	:START SHIFT COUNTER AT ZERO
3209	021034	012737	120001	027316	MOV	#CRC16,XPOLY	:LOAD POLYNOMIAL FOR SOFTWARE BCC
3210	021042	012737	000252	021102	MOV	#252,66\$;	:LOAD CHAR FOR SOFTWARE BCC
3211	021050	005037	021104		CLR	67\$	:CLEAR OLD SOFTWARE BCC
3212	021054	004737	027322		JSR	PC,BCCLD	:LOAD OUT SILO WITH 2 SYNCs
3213	021060	000252			252		:AND THE CHARACTER 252
3214	021062	104415	000021		DATACLK,	21	:GET TRANSMITTER ACTIVE
3215	021066	104415	000001		DATACLK,	1	:SHIFT BCC ONCE
3216	021072	005200			INC	R0	:BUMP SHIFT COUNT
3217	021074	004537	027212		JSR	R5,SIMBCC	:CALCULATE SOFTWARE BCC LSB
3218	021100	000001			1		:ONE SHIFT
3219	021102	000000			0		:DATA CHARACTER
3220	021104	000000			0		:OLD BCC
3221	021106	103405			BCC	68\$	:BR IF SOFT BCC LSB IS SET
3222	021110	004737	027434		JSR	PC,GETQ0	:GET HARDWARE TRANSMITTER BCC LSB
3223	021114	103006			BCC	69\$	:BR IF HARD BCC LSB IS CLEAR
3224	021116	104012			HLT	12	:ERROR, BCC LSB IS SET
3225	021120	000404			BR	69\$	:CONTINUE
3226	021122	004737	027434	68\$:	JSR	PC,GETQ0	:GET HARDWARE TRANSMITTER BCC LSB
3227	021126	103401			BCC	69\$	:BR IF HARD BCC LSB IS SET
3228	021130	104016			HLT	16	:ERROR, HARD BCC LSB IS CLEAR
3229	021132						
3230	021132	006037	021102		ROR	66\$	:SHIFT SOFT DATA

3231	021136	013737	027320	021104		MOV	CALBCC,67\$	:LOAD OLD SOFT BCC
3232	021144	022700	000010			CMP	#10,R0	:DONE YET?
3233	021150	001346				BNE	65\$	:BR IF NOT DONE
3234	021152	104401				SCOPI		:SCOPE SUBTEST (SW09=1)
3235	021154	012737	021162	001220		MOV	#71\$,LOCK	:NEW SCOPE1
3236	021162	004737	027662		71\$:	JSR	PC,CLR10	:CLEAR BCC REGISTERS
3237	021166	005000				CLR	R0	:START SHIFT COUNTER AT ZERO
3238	021170	012737	120001	027316		MOV	#CRC16,XPOLY	:LOAD POLYNOMIAL FOR SOFTWARE BCC
3239	021176	012737	000252	021236		MOV	#252,73\$	:LOAD CHAR FOR SOFTWARE BCC
3240	021204	005037	021240			CLR	74\$	:CLEAR OLD SOFTWARE BCC
3241	021210	004737	027322			JSR	PC,BCCLO	:LOAD OUT SILO WITH 2 SYNC
3242	021214	000252				252		:AND THE CHARACTER 252
3243	021216	104415	000032			DATACLK,	32	:GET RECEIVER ACTIVE
3244	021222	104415	000001		72\$:	DATACLK,	1	:SHIFT BCC ONCE
3245	021226	005200				INC	R0	:BUMP SHIFT COUNT
3246	021230	004537	027212			JSR	R5,SIMBCC	:CALCULATE SOFTWARE BCC LSB
3247	021234	000001				1		:ONE SHIFT
3248	021236	000000			73\$:	0		:DATA CHARACTER
3249	021240	000000			74\$:	0		:OLD BCC
3250	021242	103405				BCS	75\$	:BR IF SOFT BCC LSB IS SET
3251	021244	004737	027446			JSR	PC,GETQI	:GET HARDWARE RECEIVER BCC LSB
3252	021250	103006				BCC	76\$	:BR IF HARD BCC LSB IS CLEAR
3253	021252	104013				HLT	13	:ERROR, BCC LSB IS SET
3254	021254	000404				BR	76\$	:CONTINUE
3255	021256	004737	027446		75\$:	JSR	PC,GETQI	:GET HARDWARE RECEIVER BCC LSB
3256	021262	103401				BCS	76\$	:BR IF HARD BCC LSB IS SET
3257	021264	104017				HLT	17	:ERROR, BCC LSB IS CLEAR
3258	021266				76\$:			
3259	021266	006037	021236			ROR	73\$	:SHIFT SOFT DATA
3260	021272	013737	027320	021240		MOV	CALBCC,74\$	:LOAD OLD SOFT BCC
3261	021300	022700	000010			CMP	#10,R0	:DONE YET?
3262	021304	001346				BNE	72\$	:BR IF NOT DONE
3263	021306	104401				SCOPI		:SCOPE SUBTEST (SW09=1)
3264	021310	104400			77\$:	SCOPE		:SCOPE THIS TEST
3265								
3266								
3267								
3268								
3269								
3270								
3271								
3272								
3273								
3274								
3275	021312	012737	000046	001226		TST46:	MOV	#46,TSTNO
3276	021320	012737	021550	001216			MOV	#TST47,NEXT
3277								
3278	021326	104412				MSTCLR		:R1 CONTAINS BASE DMC11 ADDRESS
3279	021330	012711	004000			MOV	#BIT11,(R1)	:MASTER CLEAR DMC11
3280	021334	005003				CLR	R3	:SET LINE UNIT LOOP
3281	021336	005004				CLR	R4	:ZERO BIT COUNT
3282	021340	005005				CLR	R5	:R4 CONTAINS CHAR TO BE LOADED IN SILO
3283	021342	005037	021444			CLR	4\$	:R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
3284	021346	012737	120001	027316		MOV	#CRC16,XPOLY	:CLEAR SOFT BCC
3285	021354	004737	027464			JSR	PC,SYNLO	:LOAD POLYNOMIAL
3286	021360	010461	000004			MOV	R4,4,R1)	:LOAD SILO WITH 2 SYNC, SOM SET
								:PORT4-CHAR

***** TEST 46 *****
 :*TRANSMITTER CRC TEST
 :*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK A BINARY
 :*COUNT PATTERN, VERIFY THE LSB OF THE TRANSMITTER BCC ON EACH SHIFT
 :*****
 ; TEST 46
 ;-----
 TST46: MOV #46,TSTNO
 MOV #TST47,NEXT
 MSTCLR
 MOV #BIT11,(R1)
 CLR R3
 CLR R4
 CLR R5
 CLR 4\$
 MOV #CRC16,XPOLY
 JSR PC,SYNLO
 MOV R4,4,R1)

3287	021364	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3288	021366	122110			122110		:LOAD OUT DATA
3289	021370	005204			INC	R4	:INCREMENT TO NEXT CHARACTER
3290	021372	010461	000004		MOV	R4,4(R1)	:PORT4+CHAR
3291	021376	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3292	021400	122110			122110		:LOAD OUT DATA
3293	021402	005204			INC	R4	:INCREMENT TO NEXT CHARACTER
3294	021404	010461	000004		MOV	R4,4(R1)	:PORT4+CHAR
3295	021410	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3296	021412	122110			122110		:LOAD OUT DATA
3297	021414	004737	026350		JSR	PC,OCOR	:WAIT FOR OCOR
3298	021420	104415	000021		DATACLK,21		:CLOCK DATA
3299	021424	010537	021442	15:	MOV	R5,35	:LOAD CHAR FOR SOFT CRC
3300	021430	104415	000001	25:	DATACLK,1		:SHIFT BCC ONCE
3301	021434	004537	027212		JSR	R5,SIMBCC	:CALCULATE SOFT BCC
3302	021440	000001			1		:SOFT SHIFT COUNT
3303	021442	000000		35:	0		:SOFT CHARACTER
3304	021444	000000		45:	0		:OLD SOFT BCC
3305	021446	103405			BCC	55	:BR IF SOFT BCC LSB IS SET
3306	021450	004737	027434		JSR	PC,GETQC	:GET HARDWARE TRANSMITTER BCC LSB
3307	021454	103006			BCC	65	:BR IF OK (CLEARED)
3308	021456	104020			HLT	20	:ERROR, BCC LSB WAS SET
3309	021460	000404			BR	65	:CONTINUE WITH TEST
3310	021462	004737	027434	55:	JSR	PC,GETQC	:GET HARDWARE TRANSMITTER BCC LSB
3311	021466	103401			BCC	65	:BR IF OK (SET)
3312	021470	104021			HLT	21	:ERROR, BCC LSB WAS CLEAR
3313							
3314	021472			65:	ROR	35	:SHIFT SOFT DATA
3315	021472	006037	021442		MOV	CALBCC,45	:LOAD OLD SOFT BCC
3316	021476	013737	027320	021444	INC	R3	:INCREMENT BIT COUNTER
3317	021504	005203			CMP	#10,R3	:DONE A FULL CHARACTER YET?
3318	021506	022703	000010		BNE	25	:BR IF NO
3319	021512	001346			CLR	R3	:RESTART BIT COUNTER
3320	021514	005003			INC	R4	:INCREMENT DATA FOR SILO
3321	021516	005204			CMP	#400,R4	:DONE BINARY COUNT YET?
3322	021520	022704	000400		BLE	95	:BR IF YES
3323	021524	003404			MOV	R4,4(R1)	:PORT4+DATA
3324	021526	010461	000004		ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3325	021532	104414			122110		:LOAD OUT DATA
3326	021534	122110			INC	R5	:INCREMENT DATA
3327	021536	005205		95:	CMP	#400,R5	:DONE BINARY PATTERN YET?
3328	021540	022705	000400		BNE	15	:BR IF NO
3329	021544	001327		75:	SCOPE		:SCOPE THIS TEST
3330	021546	10440C					
3331							
3332							
3333							:***** TEST 47 *****
3334							:*RECEIVER CRC TEST
3335							:*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK A BINARY
3336							:*COUNT PATTERN, VERIFY THE LSB OF THE RECEIVER BCC ON EACH SHIFT
3337							:*****
3338							
3339							: TEST 47
3340							
3341	021550	012737	000047	001226	TST47:	MOV	#47,TSTNO
3342	021556	012737	022006	001216		MOV	#TST50,NEXT

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
---------	-----	-----	-------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	------

```

3399
3400
3401
3402
3403
3404
3405
3406
3407
3408
3409 022006 012737 000050 001226
3410 022014 012737 022340 001216
3411
3412 022022 104412
3413
3414
3415
3416 022024 012711 004000
3417 022030 012704 030102
3418 022034 005037 022130
3419 022040 012700 000004
3420 022044 004737 027464
3421 022050 004737 026502
3422 022054 004537 027620
3423 022060 030102
3424 022062 000004
3425 022064 004737 027574
3426 022070 004737 026350
3427 022074 005003
3428 022076 104415 000022
3429 022102 112405
3430 022104 010502
3431
3432
3433
3434
3435 022106 012737 120001 027316
3436 022114 010537 022126
3437 022120 004537 027212
3438 022124 000010
3439 022126 000000
3440 022130 000000
3441 022132 013737 027320 022130
3442 022140 104415 000001
3443 022144 106002
3444 022146 103005
3445 022150 004737 026316
3446 022154 103406
3447 022156 104006
3448 022160 000404
3449 022162 004737 026316
3450 022166 103001
3451 022170 104006
3452 022172
3453 022172 005203
3454 022174 022703 000010

```

```

***** TEST 50 *****
*TRANSMITTER DDCMP CRC TEST
*THIS TEST TRANSMITS A FOUR CHARACTER MESSAGE WITH JRC
*BOTH DATA AND THE BCC ARE VERIFIED IN THE BIT
*WINDOW. THE FOUR CHARACTERS ARE 0,125,252,377
*THE TRANSMITTER IS CHECKED FOR GOING TO A MARK STATE AFTER THE BCC
*****

: TEST 50
-----
TST50: MOV #50,TSTNO
      MOV #TST51,NEXT
      MSTCLR
      :LOAD OUT DATA SILO
      MOV #BIT11,(R1)
      MOV #MESDAT,R4
      CLR 10$
      MOV #4,R0
      JSR PC,SYNLD
      JSR PC,OUTRDY
      JSR R5,MESLD
      MESDAT
      4
      JSR PC,EOM
      JSR PC,OCOR
      CLR R3
      DATACLK,22
      12$: MOV# (R4)+,R5
      MOV R5,R2
      :CHECK FIRST FOUR CHARACTER MESSAGE
      :IN THE BIT WINDOW (0,125,252,377)
      MOV #CRC16,XPOLY
      MOV R5,67$
      JSR R5,SIMBCC
      10
      67$: 0
      10$: 0
      MOV CALBCC,10$
      DATACLK,1
      64$: RORB R2
      BCC 65$
      JSR PC,GETSI
      BCS 66$
      HLT 6
      65$: JSR PC,GETSI
      BCC 66$
      HLT 6
      66$: INC R3
      CMP #10,R3
      :SET LINE UNIT LOOP
      :LOAD POINTER TO DATA
      :CLEAR SOFT BCC
      :LOAD CHARACTER COUNT
      :LOAD 2 SYNC IN OUT SILC
      :WAIT FOR OUTRDY
      :LOAD SILO WITH 4 CHAR MESS
      :ADDRESS OF MESSAGE
      :NUMBER OF CHARACTERS
      :LOAD GARBAGE CHARACTER, WITH EOM SET
      :WAIT FOR OCOR
      :CLEAR BIT COUNTER
      :CLOCK DATA
      :LOAD R5 WITH CHAR
      :LOAD R2 WITH CHAR
      :LOAD POLYNOMIAL
      :LOAD SOFT CHAR FOR BCC
      :CALCULATE SOFT BCC
      :SHIFT COUNT
      :CHARACTER
      :OLD BCC
      :LOAD SOFT BCC FOR NEXT SHIFT
      :SHIFT DATA IN TO BIT WINDOW
      :SHIFT SOFT DATA
      :BR IF A SPACE
      :LOOK AT BIT WINDOW
      :BR IF OK (MARK)
      :ERROR, BIT WINDOW WAS A SPACE
      :CONTINUE
      :LOOK AT BIT WINDOW
      :BR IF OK (SPACE)
      :ERROR, BIT WINDOW WAS A MARK
      :BUMP BIT COUNTER
      :DONE FULL 8 BITS YET

```

3455	022200	001357		BNE	64\$		:BR IF NO
3456	022202	005003		CLR	R3		:CLEAR BIT COUNTER
3457	022204	005300		DEC	R0		:DEC CHARACTER COUNT
3458	022206	001335		BNE	12\$		:BR IF NOT DONE YET
3459							
3460							:CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
3461							
3462	022210	013700	027320	MOV	CALBCC,R0		:PUT BCC IN R0
3463	022214	104415	000001	68\$: DATACLK,	1		:SHIFT HARDWARE BCC
3464	022220	006000		ROR	R0		:SHIFT SOFT BCC
3465	022222	103005		BCC	69\$		:BR IF CARRY CLEAR
3466	022224	004737	026316	JSR	PC,GETSI		:LOOK AT BIT WINDOW
3467	022230	103406		BCS	70\$		:BR IF OK (MARK)
3468	022232	104014		HLT	14		:ERROR, CRC WRONG (SPACE)
3469	022234	000404		BR	70\$		:CONTINUE
3470	022236	004737	026316	59\$: JSR	PC,GETSI		:LOOK AT BIT WINDOW
3471	022242	103001		BCC	70\$		:BR IF OK (SPACE)
3472	022244	104014		HLT	14		:ERROR, CRC WRONG (MARK)
3473	022246			70\$: INC	R3		:BUMP BIT COUNTER
3474	022246	005203		CMP	#20,R3		:FINISHED BCC YET?
3475	022250	022703	000020	BNE	68\$		:BR IF NO
3476	022254	001357		CLR	R3		:CLEAR BIT COUNTER
3477	022256	005003					
3478							:CHECK TO SEE IF TRANSMITTER IS MARKING
3479							
3480							
3481	022260	104415	000001	2\$: DATACLK,	1		:CLOCK TRANSMITTER
3482	022264	004737	026316	JSR	PC,GETSI		:LOOK AT WINDOW
3483	022270	103401		BCS	3\$		:IT SHOULD BE MARKING
3484	022272	104024		HLT	24		:ERROR, BIT WAS A SPACE
3485	022274	005203		3\$: INC	R3		:BUMP BIT COUNTER
3486	022276	022703	000007	CMP	#7,R3		:DONE YET
3487	022302	001366		BNE	2\$		:BR IF NO
3488	022304	104415	000010	DATACLK,	10		:GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
3489	022310	005003		CLR	R3		:CLEAR BIT COUNTER
3490	022312	104415	000001	4\$: DATACLK,	1		:SHIFT OUT NEXT BIT
3491	022316	004737	026316	JSR	PC,GETSI		:LOOK AT BIT WINDOW
3492	022322	103401		BCS	4		:BR IF IT IS A MARK
3493	022324	104024		HLT	24		:ERROR, TRANSMITTER IS NOT MARKING
3494	022326	005203		INC	R3		:INC BIT COUNT
3495	022330	022703	000020	CMP	#20,R3		:DONE YET?
3496	022334	001366		BNE	4\$		:BR IF NO
3497	022336	104400		5\$: SCOPE			:SCOPE THIS TEST
3498							
3499							
3500							:***** TEST 51 *****
3501							:*RECEIVER DDCMP CRC TEST
3502							:*THIS TEST CLOCKS A FOUR CHARACTER MESSAGE WITH BCC
3503							:*AND VERIFYS CORRECT DATA RECEPTION AND BCC MATCH
3504							:*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
3505							:*****
3506							
3507							: TEST 51
3508							
3509	022340	012737	000051 001226	TST51: MOV	#51,TSTNO		
3510	022346	012737	022542 001216	MOV	#TST52,NEXT		


```

3567
3568
3569
3570 022542 012737 000052 001226 TST52: MOV #52,TSTNO
3571 022550 012737 023642 001216 MOV #TST53,NEXT
3572
3573 022556 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3574 ;MASTER CLEAR DMC11
3575 ;LOAD OUT DATA SILO
3576
3577 022560 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
3578 022564 012704 030102 MOV #MESDAT,R4 ;LOAD POINTER TO DATA
3579 022570 005037 022700 CLR 10$ ;CLEAR SOFT BCC
3580 022574 012700 000004 MOV #4,R0 ;LOAD CHARACTER COUNT
3581 022600 004737 027464 JSR PC,SYNLD ;LOAD 2 SYNCs IN OUT SILO
3582 022604 004737 026502 JSR PC,OUTRDY ;WAIT FOR OUTRDY
3583 022610 004537 027620 JSR R5,MESLD ;LOAD SILO WITH 4 CHAR MESS
3584 022614 030102 MESDAT ;ADDRESS OF MESSAGE
3585 022616 000004 4 ;NUMBER OF CHARACTERS
3586 022620 004737 027574 JSR PC,EOM ;LOAD GARBAGE CHARACTER, WITH EOM SET
3587 022624 004537 027620 JSR R5,MESLD ;LOAD FOUR MORE CHARACTERS
3588 022630 030102 MESDAT ;ADDRESS OF MESSAGE
3589 022632 000004 4 ;NUMBER OF CHACTERS
3590 022634 004737 027574 JSR PC,EOM ;SET EOM
3591 022640 004737 026350 JSR PC,OCOR ;WAIT FOR OCOR
3592 022644 005003 CLR R3 ;CLEAR BIT COUNTER
3593 022646 104415 000022 DATACLK,22 ;CLOCK DATA
3594 022652 112405 12$: MOVB (R4)+,R5 ;LOAD R5 WITH CHAR
3595 022654 010502 MOV R5,R2 ;LOAD R2 WITH CHAR
3596
3597 ;CHECK FIRST FOUR CHARACTER MESSAGE
3598 ;IN THE BIT WINDOW (0,125,252,377)
3599
3600 022656 012737 120001 027316 MOV #CRC16,XPOLY ;LOAD POLYNOMIAL
3601 022664 010537 022676 MOV R5,67$ ;LOAD SOFT CHAR FOR BCC
3602 022670 004537 027212 JSR R5,SIMBCC ;CALCULATE SOFT BCC
3603 022674 000010 10 ;SHIFT COUNT
3604 022676 000000 67$: 0 ;CHARACTER
3605 022700 000000 10$: 0 ;OLD BCC
3606 022702 013737 027320 022700 MOV CALBCC,10$ ;LOAD SOFT BCC FOR NEXT SHIFT
3607 022710 104415 000001 64$: DATACLK,1 ;SHIFT DATA IN TO BIT WINDOW
3608 022714 106002 RORB R2 ;SHIFT SOFT DATA
3609 022716 103005 BCC 65$ ;BR IF A SPACE
3610 022720 004737 026316 JSR PC,GETSI ;LOOK AT BIT WINDOW
3611 022724 103406 BCS 66$ ;BR IF OK (MARK)
3612 022726 104006 HLT 6 ;ERROR, BIT WINDOW WAS A SPACE
3613 022730 000404 BR 66$ ;CONTINUE
3614 022732 004737 026316 65$: JSR PC,GETSI ;LOOK AT BIT WINDOW
3615 022736 103001 BCC 66$ ;BR IF OK (SPACE)
3616 022740 104006 HLT 6 ;ERROR, BIT WINDOW WAS A MARK
3617 022742 66$:
3618 022742 005203 INC R3 ;BUMP BIT COUNTER
3619 022744 022703 000010 CMP #10,R3 ;DONE FULL 8 BITS YET
3620 022750 001357 BNE 64$ ;BR IF NO
3621 022752 005003 CLR R3 ;CLEAR BIT COUNTER
3622 022754 005300 DEC R0 ;DEC CHARACTER COUNT

```

3623	022756	001335		BNE	12\$	;BR IF NOT DONE YET
3624						
3625						;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
3626						
3627	022760	013700	027320		MOV	CALBCC,RO ;PUT BCC IN RO
3628	022764	104415	000001	68\$:	DATACLK,1 ;SHIFT HARDWARE BCC	
3629	022770	006000			ROR	RO ;SHIFT SOFT BCC
3630	022772	103005			BCC	69\$;BR IF CARRY CLEAR
3631	022774	004737	026316		JSR	PC,GETSI ;LOOK AT BIT WINDOW
3632	023000	103406			BCS	70\$;BR IF OK (MARK)
3633	023002	104014			HLT	14 ;ERROR, CRC WRONG (SPACE)
3634	023004	000404			BR	70\$;CONTINUE
3635	023006	004737	026316	69\$:	JSR	PC,GETSI ;LOOK AT BIT WINDOW
3636	023012	103001			BCC	70\$;BR IF OK (SPACE)
3637	023014	104014			HLT	14 ;ERROR, CRC WRONG (MARK)
3638	023016			70\$:		
3639	023016	005203			INC	R3 ;BUMP BIT COUNTER
3640	023020	022703	000020		CMP	#20,R3 ;FINISHED BCC YET?
3641	023024	001357			BNE	68\$;BR IF NO
3642	023026	005003			CLR	R3 ;CLEAR BIT COUNTER
3643	023030	012700	000004		MOV	#4,RO ;RESET CHARACTER COUNTER
3644	023034	012704	030102		MOV	#MESDAT,R4 ;LOAD MESSAGE POINTER
3645	023040	005037	023072		CLR	11\$;CLR SOFT BCC
3646	023044	112405		13\$:	MOVB	(R4)+,R5 ;LOAD CHAR IN R5
3647	023046	010502			MOV	R5,R2 ;LOAD CHAR IN R2
3648						
3649						;CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
3650						
3651	023050	012737	120001	027316	MOV	#CRC16,XPOLY ;LOAD POLYNOMIAL
3652	023056	010537	023070		MOV	R5,76\$;LOAD SOFT CHAR FOR BCC
3653	023062	004537	027212		JSR	R5,SIMBCC ;CALCULATE SOFT BCC
3654	023066	000010			10	;SHIFT COUNT
3655	023070	000000		76\$:	0	;CHARACTER
3656	023072	000000		11\$:	0	;OLD BCC
3657	023074	013737	027320	023072	MOV	CALBCC,11\$;LOAD SOFT BCC FOR NEXT SHIFT
3658	023102	104415	000001		DATACLK,1 ;SHIFT DATA IN TO BIT WINDOW	
3659	023106	106002		73\$:	ROR	R2 ;SHIFT SOFT DATA
3660	023110	103005			BCC	74\$;BR IF A SPACE
3661	023112	004737	026316		JSR	PC,GETSI ;LOOK AT BIT WINDOW
3662	023116	103406			BCS	75\$;BR IF OK (MARK)
3663	023120	104006			HLT	6 ;ERROR, BIT WINDOW WAS A SPACE
3664	023122	000404			BR	75\$;CONTINUE
3665	023124	004737	026316	74\$:	JSR	PC,GETSI ;LOOK AT BIT WINDOW
3666	023130	103001			BCC	75\$;BR IF OK (SPACE)
3667	023132	104006			HLT	6 ;ERROR, BIT WINDOW WAS A MARK
3668	023134			75\$:		
3669	023134	005203			INC	R3 ;BUMP BIT COUNTER
3670	023136	022703	000010		CMP	#10,R3 ;DONE FULL 8 BITS YET
3671	023142	001357			BNE	73\$;BR IF NO
3672	023144	005003			CLR	R3 ;CLEAR BIT COUNTER
3673	023146	005300			DEC	RO ;DEC CHARACTER COUNT
3674	023150	001335			BNE	13\$;BR IF NOT DONE YET
3675						
3676						;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
3677						
3678	023152	013700	027320		MOV	CALBCC,RO ;PUT BCC IN RO

3679	023156	104415	000001	77\$:	DATACLK, 1	;SHIFT HARDWARE BCC
3680	023162	006000			ROR R0	;SHIFT SOFT BCC
3681	023164	103005			BCC 78\$	;BR IF CARRY CLEAR
3682	023166	004737	026316		JSR PC,GETSI	;LOOK AT BIT WINDOW
3683	023172	103406			BCS 79\$	;BR IF OK (MARK)
3684	023174	104014			HLT 14	;ERROR, CRC WRONG (SPACE)
3685	023176	000404			BR 79\$	;CONTINUE
3686	023200	004737	026316	78\$:	JSR PC,GETSI	;LOOK AT BIT WINDOW
3687	023204	103001			BCC 79\$	;BR IF OK (SPACE)
3688	023206	104014			HLT 14	;ERROR, CRC WRONG (MARK)
3689	023210			79\$:	INC R3	;BUMP BIT COUNTER
3690	023210	005203			CMP #20,R3	;FINISHED BCC YET?
3691	023212	022703	000020		BNE 77\$	;BR IF NO
3692	023216	001357			CLR R3	;CLEAR BIT COUNTER
3693	023220	005003				
3694						
3695						;CHECK TO SEE IF TRANSMITTER IS MARKING
3696						
3697	023222	104415	000001	2\$:	DATACLK, 1	;CLOCK TRANSMITTER
3698	023226	004737	026316		JSR PC,GETSI	;LOOK AT WINDOW
3699	023232	103401			BCS 3\$	;IT SHOULD BE MARKING
3700	023234	104024			HLT 24	;ERROR, BIT WAS A SPACE
3701	023236	005203		3\$:	INC R3	;BUMP BIT COUNTER
3702	023240	022703	000007		CMP #7,R3	;DONE YET
3703	023244	001366			BNE 2\$	;BR IF NO
3704	023246	104415	000010		DATACLK, 10	;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
3705	023252	005003			CLR R3	;CLEAR BIT COUNTER
3706	023254	104415	000001	4\$:	DATACLK, 1	;SHIFT OUT NEXT BIT
3707	023260	004737	026316		JSR PC,GETSI	;LOOK AT BIT WINDOW
3708	023264	103401			BCS +4	;BR IF IT IS A MARK
3709	023266	104024			HLT 24	;ERROR, TRANSMITTER IS NOT MARKING
3710	023270	005203			INC R3	;INC BIT COUNT
3711	023272	022703	000020		CMP #20,R3	;DONE YET
3712	023276	001366			BNE 4\$	;BR IF NO
3713						
3714						;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
3715						;WAS RECEIVED CORRECTLY (0,125,252,377)
3716						
3717	023300	104415	000001		DATACLK, 1	;GET LAST BIT IN RECEIVER
3718	023304	012703	000004		MOV #4,R3	;R3=CHARACTER COUNT
3719	023310	012702	030102		MOV #MESSAGE,R2	;LOAD MESSAGE POINTER IN R2
3720	023314	004737	027156	40\$:	JSR PC,INRDY	;WAIT FOR INRDY
3721	023320	104414			ROMCLK	;NEXT WORD IS INSTRUCTION. ROMCLK PC=5334
3722	023322	021204			021204	
3723	023324	016104	000004		MOV 4(R1),R4	;PUT "FOUND" IN R4
3724	023330	112205			MOVB (R2)+,R5	;PUT "EXPECTED" IN R5
3725	023332	120504			CMPB R5,R4	;IS RECEIVED DATA CORRECT?
3726	023334	001401			BEQ 41\$	;BR IF YES
3727	023336	104010			HLT 10	;RECEIVE DATA ERROR
3728	023340	005303		41\$:	DEC R3	;DEC CHARACTER COUNT
3729	023342	001364			BNE 40\$	;BR IF NOT DONE YET
3730						
3731						
3732						;CHECK TO SEE THAT IN BCC MATCH IS SET
3733						;AND THAT THE BCC WAS RECEIVED CORRECTLY
3734	023344	004737	027156		JSR PC,INRDY	;WAIT FOR INRDY

3735	023350	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3736	023352	021204			021204		;GET FIRST HALF OF CRC
3737	023354	116137	000004	001252	MOVB	4(R1),TEMP3	;PUT IN TEMP3
3738	023362	042737	177400	001252	BIC	#177400,TEMP3	;CLEAR HI BYTE
3739	023370	004737	027156		JSR	PC,INRDY	;WAIT FOR INRDY
3740	023374	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3741	023376	021244			021244		
3742	023400	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
3743	023404	042704	000376		BIC	#376,R4	;CLEAR UNWANTED BITS
3744	023410	012705	000001		MOV	#1,R5	;PUT "EXPECTED" IN R5
3745	023414	120504			CMPB	R5,R4	;IS IN BCC MATCH SET?
3746	023416	001401			BEQ	50\$	
3747	023420	104015			HLT	15	;IN BCC MATCH ERROR
3748	023422						
3749	023422	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3750	023424	021204			021204		;GET LAST HALF
3751	023426	116137	000004	001251	MOVB	4(R1),TEMP2+1	;PUT IN TEMP2
3752	023434	042737	000377	001250	BIC	#377,TEMP2	;CLEAR LO BYTE
3753	023442	053737	001250	001252	BIS	TEMP2,TEMP3	;16 BIT BCC NOW IN TEMP3
3754	023450	023737	027320	001252	CMP	CALBCC,TEMP3	;IS IT CORRECT?
3755	023456	001401			BEQ	42\$	;BR IF OK
3756	023460	104027			HLT	27	
3757							
3758							;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
3759							;WAS RECEIVED CORRECTLY (0,125,252,377)
3760							
3761	023462	012703	000004		42\$: MOV	#4,R3	;R3=CHARACTER COUNT
3762	023466	012702	030102		MOV	#MESDAT,R2	;LOAD MESSAGE POINTER IN R2
3763	023472	004737	027156		43\$: JSR	PC,INRDY	;WAIT FOR INRDY
3764	023476	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3765	023500	021204			021204		
3766	023502	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
3767	023506	112205			MOVB	(R2)+,R5	;PUT "EXPECTED" IN R5
3768	023510	120504			CMPB	R5,R4	;IS RECEIVED DATA CORRECT?
3769	023512	001401			BEQ	44\$	;BR IF YES
3770	023514	104010			HLT	10	;RECEIVE DATA ERROR
3771	023516	005303			44\$: DEC	R3	;DEC CHARACTER COUNT
3772	023520	001364			BNE	43\$	;BR IF NOT DONE YET
3773							
3774							;CHECK TO SEE THAT IN BCC MATCH IS SET
3775							;AND THAT THE BCC WAS RECEIVED CORRECTLY
3776							
3777	023522	004737	027156		JSR	PC,INRDY	;WAIT FOR INRDY
3778	023526	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3779	023530	021204			021204		;GET FIRST HALF OF CRC
3780	023532	116137	000004	001252	MOVB	4(R1),TEMP3	;PUT IN TEMP3
3781	023540	042737	177400	001252	BIC	#177400,TEMP3	;CLEAR HI BYTE
3782	023546	004737	027156		JSR	PC,INRDY	;WAIT FOR INRDY
3783	023552	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3784	023554	021244			021244		
3785	023556	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
3786	023562	042704	000376		BIC	#376,R4	;CLEAR UNWANTED BITS
3787	023566	012705	000001		MOV	#1,R5	;PUT "EXPECTED" IN R5
3788	023572	120504			CMPB	R5,R4	;IS IN BCC MATCH SET?
3789	023574	001401			BEQ	51\$	
3790	023576	104015			HLT	15	;IN BCC MATCH ERROR

3791	023600				51\$:	ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3792	023600	104414				021204		;GET LAST HALF
3793	023602	021204				MOVB	4(R1),TEMP2+1	;PUT IN TEMP2
3794	023604	116137	000004	001251		BIC	#377,TEMP2	;CLEAR LO BYTE
3795	023612	042737	000377	001250		BIS	TEMP2,TEMP3	;16 BIT BCC NOW IN TEMP3
3796	023620	053737	001250	001252		CMP	CALBCC,TEMP3	;IS IT CORRECT?
3797	023626	023737	027320	001252		BEQ	5\$	;BR IF OK
3798	023634	001401				HLT	27	
3799	023636	104027			5\$:	SCOPE		;SCOPE THIS TEST
3800	023640	104400						
3801								
3802								
3803								
3804								
3805								
3806								
3807								
3808								
3809								
3810								
3811								
3812								
3813								
3814								
3815								
3816								
3817	023642	012737	000053	001226	TST53:	MOV	#53,TS1NO	
3818	023650	012737	025042	001216		MOV	#TS154,NEXT	
3819								
3820	023656	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
3821								;MASTER CLEAR DMC11
3822								
3823								
3824	023660	012711	004000			MOV	#BIT11,(R1)	;SET LINE UNIT LOOP
3825	023664	012704	030102			MOV	#MESDAT,R4	;LOAD POINTER TO DATA
3826	023670	005037	024004			CLR	10\$	;CLEAR SOFT BCC
3827	023674	012700	000004			MOV	#4,R0	;LOAD CHARACTER COUNT
3828	023700	004737	027464			JSR	PC,SYNLD	;LOAD 2 SYNC'S IN OUT SILO
3829	023704	004737	026502			JSR	PC,OUTRDY	;WAIT FOR OUTRDY
3830	023710	004537	027620			JSR	R5,MESLD	;LOAD SILO WITH 4 CHAR MESS
3831	023714	030102				MESDAT		;ADDRESS OF MESSAGE
3832	023716	000004				4		;NUMBER OF CHARACTERS
3833	023720	004737	027574			JSR	PC,EOM	;LOAD GARBAGE CHARACTER, WITH EOM SET
3834	023724	004737	027544			JSR	PC,SOM	;LOAD GARBAGE CHAR WITH SOM SET
3835	023730	004537	027620			JSR	R5,MESLD	;LOAD FOUR MORE CHARACTERS
3836	023734	030102				MESDAT		;ADDRESS OF MESSAGE
3837	023736	000004				4		;NUMBER OF CHACTERS
3838	023740	004737	027574			JSR	PC,EOM	;SET EOM
3839	023744	004737	026350			JSR	PC,OCOR	;WAIT FOR OCOR
3840	023750	005003				CLR	R3	;CLEAR BIT COUNTER
3841	023752	104415	000022			DATACLK,22		;CLOCK DATA
3842	023756	112405			12\$:	MOVB	(R4)+,R5	;LOAD R5 WITH CHAR
3843	023760	010502				MOV	R5,R2	;LOAD R2 WITH CHAR
3844								
3845								
3846								

***** TEST 53 *****
;DDCMP EOM FUNCTION TEST
;THIS TEST LOADS OUT SILO WITH: 2 SYNC'S, 4 CHAR MESSAGE, EOM
;SOM, 4 CHAR MESS, EOM. THE DATA STREAM IS CHECKED TO BE
;4 CHAR, BCC 4 CHAR, BCC, MARKS. THIS TEST VERIFYS THAT
;THE CHARACTERS LOADED WITH EOM SET ARE LOST
;ALSO THAT THE CHAR LOADED WITH SOM IS NOT IN THE BCC
;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 53

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;LOAD OUT DATA SILO
;SET LINE UNIT LOOP
;LOAD POINTER TO DATA
;CLEAR SOFT BCC
;LOAD CHARACTER COUNT
;LOAD 2 SYNC'S IN OUT SILO
;WAIT FOR OUTRDY
;LOAD SILO WITH 4 CHAR MESS
;ADDRESS OF MESSAGE
;NUMBER OF CHARACTERS
;LOAD GARBAGE CHARACTER, WITH EOM SET
;LOAD GARBAGE CHAR WITH SOM SET
;LOAD FOUR MORE CHARACTERS
;ADDRESS OF MESSAGE
;NUMBER OF CHACTERS
;SET EOM
;WAIT FOR OCOR
;CLEAR BIT COUNTER
;CLOCK DATA
;LOAD R5 WITH CHAR
;LOAD R2 WITH CHAR
;CHECK FIRST FOUR CHARACTER MESSAGE
;IN THE BIT WINDOW (0,125,252,377)

3847									
3848	023762	012737	120001	027316		MOV	#CRC16,XPOLY		;LOAD POLYNOMIAL
3849	023770	010537	024002			MOV	R5,67\$		;LOAD SOFT CHAR FOR BCC
3850	023774	004537	027212			JSR	R5,SIMBCC		;CALCULATE SOFT BCC
3851	024000	000010				LD			;SHIFT COUNT
3852	024002	000000			67\$:	0			;CHARACTER
3853	024004	000000			10\$:	0			;OLD BCC
3854	024006	013737	027320	024004		MOV	CALBCC,10\$		;LOAD SOFT BCC FOR NEXT SHIFT
3855	024014	104415	000001		64\$:	DATACLK,	1		;SHIFT DATA IN TO BIT WINDOW
3856	024020	106002				RORB	R2		;SHIFT SOFT DATA
3857	024022	103005				BCC	65\$		;BR IF A SPACE
3858	024024	004737	026316			JSR	PC,GETSI		;LOOK AT BIT WINDOW
3859	024030	103406				BCS	66\$		;BR IF OK (MARK)
3860	024032	104006				HLT	6		;ERROR, BIT WINDOW WAS A SPACE
3861	024034	000404				BR	66\$		;CONTINUE
3862	024036	004737	026316		65\$:	JSR	PC,GETSI		;LOOK AT BIT WINDOW
3863	024042	103001				BCC	66\$		;BR IF OK (SPACE)
3864	024044	104006				HLT	6		;ERROR, BIT WINDOW WAS A MARK
3865	024046				66\$:				
3866	024046	005203				INC	R3		;BUMP BIT COUNTER
3867	024050	022703	000010			CMP	#10,R3		;DONE FULL 8 BITS YET
3868	024054	001357				BNE	64\$		;BR IF NO
3869	024056	005003				CLR	R3		;CLEAR BIT COUNTER
3870	024060	005300				DEC	R0		;DEC CHARACTER COUNT
3871	024062	001335				BNE	12\$		;BR IF NOT DONE YET
3872									
3873									;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
3874									
3875	024064	013700	027320			MOV	CALBCC,R0		;PUT BCC IN R0
3876	024070	104415	000001		68\$:	DATACLK,	1		;SHIFT HARDWARE BCC
3877	024074	006000				ROR	R0		;SHIFT SOFT BCC
3878	024076	103005				BCC	69\$		;BR IF CARRY CLEAR
3879	024100	004737	026316			JSR	PC,GETSI		;LOOK AT BIT WINDOW
3880	024104	103406				BCS	70\$		;BR IF OK (MARK)
3881	024106	104014				HLT	14		;ERROR, CRC WRONG (SPACE)
3882	024110	000404				BR	70\$		;CONTINUE
3883	024112	004737	026316		69\$:	JSR	PC,GETSI		;LOOK AT BIT WINDOW
3884	024116	103001				BCC	70\$		;BR IF OK (SPACE)
3885	024120	104014				HLT	14		;ERROR, CRC WRONG (MARK)
3886	024122				70\$:				
3887	024122	005203				INC	R3		;BUMP BIT COUNTER
3888	024124	022703	000020			CMP	#20,R3		;FINISHED BCC YET?
3889	024130	001357				BNE	68\$		;BR IF NO
3890	024132	005003				CLR	R3		;CLEAR BIT COUNTER
3891									
3892									;CHECK CHARACTER LOADED WITH SOM (000), IN THE BIT WINDOW
3893									
3894	024134	005005				CLR	R5		;CHARACTER LOADED WITH SOM
3895	024136	010502				MOV	R5,R2		;LOAD R2 WITH CHAR
3896	024140	104415	000001		32\$:	DATACLK,	1		;CLOCK TRANSMITTER
3897	024144	106002				RORB	R2		;SHIFT SOFT DATA
3898	024146	103005				BCC	30\$		;BR IF SPACE
3899	024150	004737	026316			JSR	PC,GETSI		;LOOK AT BIT WINDOW
3900	024154	103406				BCS	31\$		;BR IF OK (MARK)
3901	024156	104006				HLT	6		;ERROR, BIT WINDOW WAS A SPACE
3902	024160	000404				BR	31\$		;CONTINUE

3903	024162	004737	026316	30\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
3904	024166	103001			BCC	31\$	;BR IF OK (SPACE)
3905	024170	104006			HLT	6	;ERROR, BIT WINDOW WAS A MARK
3906	024172	005203		31\$:	INC	R3	;BUMP BIT COUNTER
3907	024174	022703	000010		CMP	#10,R3	;DONE CHARACTER YET?
3908	024200	001357			BNE	32\$	;BR IF NO
3909	024202	005003			CLR	R3	;RESET BIT COUNTER
3910	024204	012700	000004		MOV	#4,R0	;RESET CHARACTER COUNTER
3911	024210	012704	030102		MOV	#MESDAT,R4	;LOAD MESSAGE POINTER
3912	024214	005037	024246		CLR	11\$	;CLR SOFT BCC
3913	024220	112405		13\$:	MOVB	(R4)+,R5	;LOAD CHAR IN R5
3914	024222	010502			MOV	R5,R2	;LOAD CHAR IN R2
3915							
3916							;CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
3917							
3918	024224	012737	120001	027316	MOV	#CRC16,XPOLY	;LOAD POLYNOMIAL
3919	024232	010537	024244		MOV	R5,76\$	;LOAD SOFT CHAR FOR BCC
3920	024236	004537	027212		JSR	R5,SIMBCC	;CALCULATE SOFT BCC
3921	024242	000010			LD		;SHIFT COUNT
3922	024244	000000		76\$:	D		;CHARACTER
3923	024246	000000		11\$:	D		;OLD BCC
3924	024250	013737	027320	024246	MOV	CALBCC,11\$	;LOAD SOFT BCC FOR NEXT SHIFT
3925	024256	104415	000001	73\$:	DATACLK,	1	;SHIFT DATA IN TO BIT WINDOW
3926	024262	106002			RORB	R2	;SHIFT SOFT DATA
3927	024264	103005			BCC	74\$	;BR IF A SPACE
3928	024266	004737	026316		JSR	PC,GETSI	;LOOK AT BIT WINDOW
3929	024272	103406			BCS	75\$	;BR IF OK (MARK)
3930	024274	104006			HLT	6	;ERROR, BIT WINDOW WAS A SPACE
3931	024276	000404			BR	75\$	;CONTINUE
3932	024300	004737	026316	74\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
3933	024304	103001			BCC	75\$	;BR IF OK (SPACE)
3934	024306	104006			HLT	6	;ERROR, BIT WINDOW WAS A MARK
3935	024310			75\$:			
3936	024310	005203			INC	R3	;BUMP BIT COUNTER
3937	024312	022703	000010		CMP	#10,R3	;DONE FULL 8 BITS YET?
3938	024316	001357			BNE	73\$	;BR IF NO
3939	024320	005003			CLR	R3	;CLEAR BIT COUNTER
3940	024322	005300			DEC	R0	;DEC CHARACTER COUNT
3941	024324	001335			BNE	13\$	;BR IF NOT DONE YET
3942							
3943							;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
3944							
3945	024326	013700	027320		MOV	CALBCC,R0	;PUT BCC IN R0
3946	024332	104415	000001	77\$:	DATACLK,	1	;SHIFT HARDWARE BCC
3947	024336	006000			ROR	R0	;SHIFT SOFT BCC
3948	024340	103005			BCC	78\$	;BR IF CARRY CLEAR
3949	024342	004737	026316		JSR	PC,GETSI	;LOOK AT BIT WINDOW
3950	024346	103406			BCS	79\$	;BR IF OK (MARK)
3951	024350	104014			HLT	14	;ERROR, CRC WRONG (SPACE)
3952	024352	000404			BR	79\$	;CONTINUE
3953	024354	004737	026316	78\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
3954	024360	103001			BCC	79\$	;BR IF OK (SPACE)
3955	024362	104014			HLT	14	;ERROR, CRC WRONG (MARK)
3956	024364			79\$:			
3957	024364	005203			INC	R3	;BUMP BIT COUNTER
3958	024366	022703	000020		CMP	#20,R3	;FINISHED BCC YET?

3959	024372	001357		BNE	77\$		;BR IF NO
3960	024374	005003		CLR	R3		;CLEAR BIT COUNTER
3961							
3962							;CHECK TO SEE IF TRANSMITTER IS MARKING
3963							
3964	024376	104415	000001	2\$:	DATACLK,	1	;CLOCK TRANSMITTER
3965	024402	004737	026316		JSR	PC,GETSI	;LOOK AT WINDOW
3966	024406	103401			BCS	3\$	;IT SHOULD BE MARKING
3967	024410	104024			HLT	24	;ERROR, BIT WAS A SPACE
3968	024412	005203		3\$:	INC	R3	;BUMP BIT COUNTER
3969	024414	022703	000007		CMP	#7,R3	;DONE YET
3970	024420	001366			BNE	2\$	;BR IF NO
3971	024422	104415	000010		DATACLK,	10	;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
3972	024426	005003			CLR	R3	;CLEAR BIT COUNTER
3973	024430	104415	000001	4\$:	DATACLK,	1	;SHIFT OUT NEXT BIT
3974	024434	004737	026316		JSR	PC,GETSI	;LOOK AT BIT WINDOW
3975	024440	103401			BCS	+4	;BR IF IT IS A MARK
3976	024442	104024			HLT	24	;ERROR, TRANSMITTER IS NOT MARKING
3977	024444	005203			INC	R3	;INC BIT COUNT
3978	024446	022703	000020		CMP	#20,R3	;DONE YET?
3979	024452	001366			BNE	4\$	;BR IF NO
3980							
3981							;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
3982							;WAS RECEIVED CORRECTLY (0,125,252,377)
3983							
3984	024454	104415	000001		DATACLK,	1	;GET LAST BIT IN RECEIVER
3985	024460	012703	000004		MOV	#4,R3	;R3=CHARACTER COUNT
3986	024464	012702	030102		MOV	#MESDAT,R2	;LOAD MESSAGE POINTER IN R2
3987	024470	004737	027156	40\$:	JSR	PC,INRDY	;WAIT FOR INRDY
3988	024474	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3989	024476	021204			021204		
3990	024500	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
3991	024504	112205			MOVB	(R2)+,R5	;PUT "EXPECTED" IN R5
3992	024506	120504			CMPB	R5,R4	;IS RECEIVED DATA CORRECT?
3993	024510	001401			BEG	41\$	;BR IF YES
3994	024512	104010			HLT	10	;RECEIVE DATA ERROR
3995	024514	005303		41\$:	DEC	R3	;DEC CHARACTER COUNT
3996	024516	001364			BNE	40\$	;BR IF NOT DONE YET
3997							
3998							;CHECK TO SEE THAT IN BCC MATCH IS SET
3999							;AND THAT THE BCC WAS RECEIVED CORRECTLY
4000							
4001	024520	004737	027156		JSR	PC,INRDY	;WAIT FOR INRDY
4002	024524	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4003	024526	021204			021204		;GET FIRST HALF OF CRC
4004	024530	116137	000004		MOVB	4(R1),TEMP3	;PUT IN TEMP3
4005	024536	042737	177400		BIC	#177400,TEMP3	;CLEAR HI BYTE
4006	024544	004737	027156		JSR	PC,INRDY	;WAIT FOR INRDY
4007	024550	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4008	024552	021244			021244		
4009	024554	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
4010	024560	042704	000376		BIC	#376,R4	;CLEAR UNWANTED BITS
4011	024564	012705	000001		MOV	#1,R5	;PUT "EXPECTED" IN R5
4012	024570	120504			CMPB	R5,R4	;IS IN BCC MATCH SET?
4013	024572	001401			BEG	50\$	
4014	024574	104015			HLT	15	;IN BCC MATCH ERROR

4015	024576			50\$:	ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4016	024576	104414			021204		;GET LAST HALF
4017	024600	021204			MOVB	4(R1),TEMP2+1	;PUT IN TEMP2
4018	024602	116137	000004	001251	BIC	#377,TEMP2	;CLEAR LO BYTE
4019	024610	042737	000377	001250	BIS	TEMP2,TEMP3	;16 BIT BCC NOW IN TEMP3
4020	024616	053737	001250	001252	CMP	CALBCC,TEMP3	;IS IT CORRECT?
4021	024624	023737	027320	001252	BEQ	45\$	;BR IF OK
4022	024632	001401			HLT	27	
4023	024634	104027					
4024							
4025							;CHECK THAT CHARACTER LOADED WITH SOM WAS RECEIVED (000)
4026							
4027	024636	004737	027156	45\$:	JSR	PC,INRDY	;WAIT FOR INRDY
4028	024642	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4029	024644	021204			021204		;GET RECEIVE DATA
4030	024646	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
4031	024652	005005			CLR	R5	;PUT "EXPECTED" IN R5
4032	024654	120504			CMPB	R5,R4	;IS RECEIVED DATA CORRECT?
4033	024656	001401			BEQ	42\$	;BR IF YES
4034	024660	104010			HLT	10	;RECEIVE DATA ERROR
4035							
4036							;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
4037							;WAS RECEIVED CORRECTLY (0,125,252,377)
4038							
4039	024662	012703	000004	42\$:	MOV	#4,R3	;R3=CHARACTER COUNT
4040	024666	012702	030102		MOV	#MESDAT,R2	;LOAD MESSAGE POINTER IN R2
4041	024672	004737	027156	43\$:	JSR	PC,INRDY	;WAIT FOR INRDY
4042	024676	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4043	024700	021204			021204		
4044	024702	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
4045	024706	112205			MOVB	(R2)+,R5	;PUT "EXPECTED" IN R5
4046	024710	120504			CMPB	R5,R4	;IS RECEIVED DATA CORRECT?
4047	024712	001401			BEQ	44\$	;BR IF YES
4048	024714	104010			HLT	10	;RECEIVE DATA ERROR
4049	024716	005303		44\$:	DEC	R3	;DEC CHARACTER COUNT
4050	024720	001364			BNE	43\$	;BR IF NOT DONE YET
4051							
4052							;CHECK TO SEE THAT IN BCC MATCH IS SET
4053							;AND THAT THE BCC WAS RECEIVED CORRECTLY
4054							
4055	024722	004737	027156		JSR	PC,INRDY	;WAIT FOR INRDY
4056	024726	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4057	024730	021204			021204		;GET FIRST HALF OF CRC
4058	024732	116137	000004	001252	MOVB	4(R1),TEMP3	;PUT IN TEMP3
4059	024740	042737	177400	001252	BIC	#177400,TEMP3	;CLEAR HI BYTE
4060	024746	004737	027156		JSR	PC,INRDY	;WAIT FOR INRDY
4061	024752	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4062	024754	021244			021244		
4063	024756	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
4064	024762	042704	000376		BIC	#376,R4	;CLEAR UNWANTED BITS
4065	024766	012705	000001		MOV	#1,R5	;PUT "EXPECTED" IN R5
4066	024772	120504			CMPB	R5,R4	;IS IN BCC MATCH SET?
4067	024774	001401			BEQ	51\$	
4068	024776	104015			HLT	15	;IN BCC MATCH ERROR
4069	025000			51\$:			
4070	025000	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```

4071 025002 021204          021204          ;GET LAST HALF
4072 025004 116137 000004 001251      MOV      4(R1),TEMP2+1      ;PUT IN TEMP2
4073 025012 042737 000377 001250      BIC      #377,TEMP2      ;CLEAR LO BYTE
4074 025020 053737 001250 001252      BIS      TEMP2,TEMP3      ;16 BIT BCC NOW IN TEMP3
4075 025026 023737 027320 001252      CMP      CALBCC,TEMP3      ;IS IT CORRECT?
4076 025034 001401          001252      BEQ      5$              ;BR IF OK
4077 025036 104027          001252      HLT      27
4078 025040          5$: SCOPE          ;SCOPE THIS TEST
4079
4080
4081
4082
4083
4084
4085
4086
4087
4088
4089
4090
4091 025042 012737 000054 001226      TST54: MOV      #54,TSTNO
4092 025050 012737 025274 001216      MOV      #TST55,NEXT
4093
4094 025056 104412          001226      MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
4095 025060 012711 004000          001226      MOV      #BIT11,(R1)      ;MASTER CLEAR DMC11
4096 025064 012702 030102          001226      MOV      #MESDAT,R2      ;SET LINE UNIT LOOP
4097 025070 012700 000004          001226      MOV      #4,R0      ;R2 POINTS TO MESSAGE
4098 025074 004737 027464          001226      JSR      PC,SYNLD      ;R0 = CHAR COUNT
4099 025100 004737 026502          001226      JSR      PC,OUTRDY      ;LOAD SILO WITH TWO SYNCS
4100 025104 004537 027620          001226      JSR      R5,MESLD      ;WAIT FOR OUTRDY
4101 025110 030102          001226      MESDAT          ;LOAD MESSAGE IN SILO
4102 025112 000004          001226      4              ;START OF MESSAGE
4103 025114 004737 026350          001226      JSR      PC,OCOR      ;CHARACTER COUNT
4104 025120 104415 000065          001226      DATACLK, 65      ;WAIT FOR OCOR
4105 025124 004537 027620          001226      JSR      R5,MESLD      ;CLOCK DATA (EMPTY SILO)
4106 025130 030102          001226      MESDAT          ;PUT MORE CHARACTERS IN SILO
4107 025132 000004          001226      4
4108 025134 004737 026350          001226      JSR      PC,OCOR
4109 025140 104415 000005          001226      DATACLK, 5
4110 025144 104414          001226      ROMCLK
4111 025146 021264          001226      021264
4112 025150 032761 000040 000004      BIT      #BITS,4,(R1)
4113 025156 001401          000004      BEQ      5$
4114 025160 104034          000004      HLT      34
4115 025162 104415 000041          000004      5$: DATACLK, 41
4116 025166 004737 027156          000004      1$: JSR      PC,INRDY
4117 025172 104414          000004      ROMCLK
4118 025174 021204          000004      021204
4119 025176 016104 000004          000004      MOV      4(R1),R4
4120 025202 112205          000004      MOV      (R2)+,R5
4121 025204 120504          000004      CMP      R5,R4
4122 025206 001401          000004      BEQ      2$
4123 025210 104010          000004      HLT      10
4124 025212 005300          000004      2$: DEC      R0
4125 025214 001364          000004      BNE      1$
4126 025216 004737 027156          000004      3$: JSR      PC,INRDY

```

```

*****TEST 54*****
;EMPTY SILO TEST
;LOAD SILO WITH 2 SYNCS, 4 CHAR MESSAGE, SINGLE CLOCK
;UNTIL THE SILO IS EMPTY, LOAD 4 MORE CHARACTERS IN THE
;SILO. GIVE MORE TICKS, AND VERIFY THAT ONLY THE FIRST
;4 CHARACTER MESSAGE WAS RECEIVED AND THAT RTS IS CLEAR
*****

```

```

; TEST 54
;-----

```

```

MOV      #54,TSTNO
MOV      #TST55,NEXT

MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
MOV      #BIT11,(R1)      ;MASTER CLEAR DMC11
MOV      #MESDAT,R2      ;SET LINE UNIT LOOP
MOV      #4,R0      ;R2 POINTS TO MESSAGE
JSR      PC,SYNLD      ;R0 = CHAR COUNT
JSR      PC,OUTRDY      ;LOAD SILO WITH TWO SYNCS
JSR      R5,MESLD      ;WAIT FOR OUTRDY
MESDAT          ;LOAD MESSAGE IN SILO
4              ;START OF MESSAGE
JSR      PC,OCOR      ;CHARACTER COUNT
DATACLK, 65      ;WAIT FOR OCOR
JSR      R5,MESLD      ;CLOCK DATA (EMPTY SILO)
MESDAT          ;PUT MORE CHARACTERS IN SILO
4

JSR      PC,OCOR
DATACLK, 5
ROMCLK
021264
BIT      #BITS,4,(R1)
BEQ      5$
HLT      34
5$: DATACLK, 41
1$: JSR      PC,INRDY
ROMCLK
021204
MOV      4(R1),R4
MOV      (R2)+,R5
CMP      R5,R4
BEQ      2$
HLT      10
2$: DEC      R0
BNE      1$
3$: JSR      PC,INRDY

```

```

4127 025222 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4128 025224 021204 021204 ;GET RECEIVE DATA
4129 025226 016104 000004 MOV 4(R1),R4 ;PUT IT IN "FOUND"
4130 025232 012705 000377 MOV #377,R5 ;R5 = "EXPECTED"
4131 025236 120504 CMPB R5,R4 ;SHOULD SEE 377
4132 025240 001401 BEQ 4$ ;BR IF OK
4133 025242 104010 HLT 10 ;ERROR, TRANSMITTER DID NOT ABORT
4134 025244 004737 027156 4$: JSR PC,INRDY ;WAIT FOR INRDY
4135 025250 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4136 025252 021204 021204 ;GET RECEIVE DATA
4137 025254 016104 000004 MOV 4(R1),R4 ;PUT IT IN "FOUND"
4138 025260 012705 000377 MOV #377,R5 ;R5 = "EXPECTED"
4139 025264 120504 CMPB R5,R4 ;SHOULD SEE 377
4140 025266 001401 BEQ 10$ ;BR IF OK
4141 025270 104010 HLT 10 ;ERROR, TRANSMITTER DID NOT ABORT
4142 025272 10$: ;SCOPE THIS TEST
4143 025272 104400 SCOPE
4144
4145
4146 ;***** TEST 55 *****
4147 ;*HALF DUPLEX TEST
4148 ;*SET LINE UNIT LOOP AND HALF DUPLEX, SEND SYNCs AND A
4149 ;*MESSAGE. VERIFY THAT IN-ACTIVE AND IN-READY ARE CLEAR
4150 ;*****
4151
4152 : TEST 55
4153 :-----
4154 025274 012737 000055 001226 TST55: MOV #55,TSTNO
4155 025302 012737 025412 001216 MOV #TST56,NEXT
4156
4157 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
4158 025310 104412 ;MASTER CLEAR DMC1:
4159 025312 012702 000012 MOV #12,R2 ;SAVE R2 FOR TYPEOUT
4160 025316 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
4161 025322 012761 000020 000004 MOV #BIT4,4(R1) ;LOAD PORT4
4162 025332 122113 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4163 025334 004737 027464 122113 JSR PC,SYNLD ;SET H/D BIT
4164 025340 004737 026502 JSR PC,OUTRDY ;LOAD 2 SYNCs
4165 025344 004537 027620 JSR R5,MESLD ;WAIT FOR OUTRDY
4166 025350 030102 MESDAT ;LOAD 4 CHAR MESSAGE
4167 025352 000004 4 ;ADDRESS OF MESSAGE
4168 025354 004737 026350 JSR PC,OCOR ;CHARACTER COUNT
4169 025360 104415 000073 DATACLK, 73 ;WAIT FOR OCOR
4170 025364 104414 ROMCLK ;SEND MESSAGE
4171 025366 021244 021244 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4172 025370 016104 000004 MOV 4(R1),R4 ;READ LU-12
4173 025374 042704 000257 BIC #257,R4 ;PUT "FOUND" IN R4
4174 025400 005005 CLR R5 ;CLEAR UNWANTED BITS
4175 025402 120504 CMPB R5,R4 ;R5 = "EXPECTED"
4176 025404 001401 BEQ 1$ ;IN-ACTIVE AND IN-RDY SHOULD BE CLEAR
4177 025406 104035 HLT 35 ;BR IF OK
4178 025410 104400 1$: ;ERROR BOTH ARE NOT CLEAR
4179 SCOPE
4180
4181
4182 ;***** TEST 56 *****
4183 ;*DDCMP CABLE DATA TEST

```

```

4183 ;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
4184 ;*4 SYNCs, 16 CHAR, EOM, 16 CHAR, EOM, 16 CHAR, EOM
4185 ;*THE 16 CHARACTERS INCLUDE A FLOATING ONE AND ZERO
4186 ;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
4187 ;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
4188 ;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
4189 ;*****
4190
4191 ; TEST 56
4192 ;-----
4193 025412 012737 000056 001226 TST56: MOV #56, TSTNO
4194 025420 012737 026000 001216 MOV #TST57, NEXT
4195
4196 025426 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
4197 025430 032737 040000 001366 BIT #BIT14, STAT1 ;MASTER CLEAR DMC11
4198 025436 001557 BEQ 3$ ;SKIP TEST IF NO
4199 025440 012711 004000 MOV #BIT11, (R1) ;LOOPBACK CONNECTOR ON
4200 025444 004737 027464 JSR PC, SYNLD ;SET LINE UNIT LOOP
4201 025450 004737 027464 JSR PC, SYNLD ;LOAD 2 SYNCs
4202 025454 012737 120001 027316 MOV #CRC16, XPOLY ;LOAD 2 MORE SYNCs
4203 025462 005037 025512 CLR 6$ ;LOAD POLYNOMIAL FOR SOFT CRC CALC
4204 025466 012703 000020 MOV #16, R3 ;CLEAR OLD BCC
4205 025472 012702 030106 MOV #FLDAT, R2 ;CHARACTER COUNT
4206 025476 112237 025510 7$: MOVB (R2)+, 5$ ;R2= POINTER
4207 025502 004537 027212 JSR R5, SIMBCC ;LOAD CHAR FOR SOFT BCC CALC.
4208 025506 000010 10 JSR R5, SIMBCC ;CALC SOFT BCC
4209 025510 000000 0 ;SHIFT COUNT
4210 025512 000000 0 ;CHARACTER
4211 025514 013737 027320 025512 6$: MOV CALBCC, 6$ ;OLD BCC
4212 025522 005303 DEC R3 ;LOAD OLD BCC
4213 025524 001364 BNE 7$ ;DEC COUNT
4214 025526 004537 027620 JSR R5, MESLD ;BR IF NOT DONE YET
4215 025532 030106 FLTDAT ;LOAD SILO
4216 025534 000020 16. ;MESSAGE ADDRESS
4217 025536 004737 027574 JSR PC, EOM ;CHARACTER COUNT
4218 025542 004537 027620 JSR R5, MESLD ;LOAD AN EOM
4219 025546 030106 FLTDAT ;LOAD SILO
4220 025550 000020 16. ;MESSAGE ADDRESS
4221 025552 004737 027574 JSR PC, EOM ;CHARACTER COUNT
4222 025556 004537 027620 JSR R5, MESLD ;LOAD AN EOM
4223 025562 030106 FLTDAT ;LOAD SILO
4224 025564 000020 16. ;MESSAGE ADDRESS
4225 025566 004737 027574 JSR PC, EOM ;CHARACTER COUNT
4226 025572 004737 026350 JSR PC, OCOR ;LOAD AN EOM
4227 025576 005011 CLR (R1) ;WAIT FOR OCOR
4228 025600 012700 000003 MOV #3, R0 ;CLEAR LINE UNIT LOOP
4229 025604 012703 000020 MOV #16, R3 ;R0 = MESSAGE COUNT
4230 025610 012702 030106 MOV #FLDAT, R2 ;R3= CHARACTER COUNT
4231 025614 004737 027156 1$: JSR PC, INRDY ;LOAD MESSAGE POINTER IN R2
4232 025620 104414 ROMCLK ;WAIT FOR INRDY
4233 025622 021204 021204 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4234 025624 016104 000004 MOV 4(R1), R4 ;GET DATA FROM IN SILO
4235 025630 112205 MOVB (R2)+, R5 ;PUT CHARACTER IN "FOUND"
4236 025632 120504 CMPB R5, R4 ;PUT "EXPECTED" IN R5
4237 025634 001401 BEQ 2$ ;IS RECEIVED DATA CORRECT
4238 025636 104025 HLT 2$ ;BR IF OK
;DATA ERROR

```

```

4239 025640          25:      DEC      R3          ;DEC CHARACTER COUNT
4240 025640 005303      BNE      1$          ;BR IF NOT DONE THIS MESSAGE
4241 025642 001364      MOV      #16.,R3      ;RESET CHARACTER COUNT
4242 025644 012703 000020
4243
4244          ;CHECK TO SEE THAT IN BCC MATCH IS SET
4245          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
4246
4247 025650 004737 027156      JSR      PC,INRDY      ;WAIT FOR INRDY
4248 025654 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4249 025656 021204      021204      ;GET FIRST HALF OF CRC
4250 025660 116137 000004 001252      MOVB      4(R1),TEMP3      ;PUT IN TEMP3
4251 025666 042737 177400 001252      BIC      #177400,TEMP3      ;CLEAR HI BYTE
4252 025674 004737 027156      JSR      PC,INRDY      ;WAIT FOR INRDY
4253 025700 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4254 025702 021244      021244
4255 025704 016104 000004      MOV      4(R1),R4      ;PUT "FOUND" IN R4
4256 025710 042704 000376      BIC      #376,R4      ;CLEAR UNWANTED BITS
4257 025714 012705 000001      MOV      #1,R5      ;PUT "EXPECTED" IN R5
4258 025720 120504      CMPB      R5,R4      ;IS IN BCC MATCH SET?
4259 025722 001401      BEQ      25$
4260 025724 104015      HLT      15      ;IN BCC MATCH ERROR
4261 025726          25$:
4262 025726 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4263 025730 021204      021204      ;GET LAST HALF
4264 025732 116137 000004 001251      MOVB      4(R1),TEMP2+1      ;PUT IN TEMP2
4265 025740 042737 000377 001250      BIC      #377,TEMP2      ;CLEAR LO BYTE
4266 025746 053737 001250 001252      BIS      TEMP2,TEMP3      ;16 BIT BCC NOW IN TEMP3
4267 025754 023737 027320 001252      CMP      CALBCC,TEMP3      ;IS IT CORRECT?
4268 025762 001401      BEQ      4$      ;BR IF OK
4269 025764 104027      HLT      27
4270 025766 012702 030106      4$:      MOV      #FLTDAT,R2      ;RESET MESSAGE POINTER
4271 025772 005300      DEC      R0      ;DECREMENT COUNTER
4272 025774 001307      BNE      1$      ;BR IF NOT DONE
4273 025776 104400      3$:      SCOPE      ;SCOPE THIS TEST
4274
4275
4276          ;***** TEST 57 *****
4277          ;*DDCMP CABLE DATA TEST
4278          ;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
4279          ;*4 SYNCs, 59 DATA CHARACTERS, EOM WITH GARBAGE CHARACTER
4280          ;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
4281          ;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
4282          ;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
4283          ;*****
4284
4285          : TEST 57
4286          : -----
4287 026000 012737 000057 001226      TST57:      MOV      #57,TSTNO
4288 026006 012737 003364 001216      MOV      #.EOP,NEXT
4289
4290 026014 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
4291 026016 032737 040000 001366      BIT      #BIT14,STAT1      ;MASTER CLEAR DMC11
4292 026024 001533      BEQ      3$      ;SKIP TEST IF NO
4293 026026 012711 004000      MOV      #BIT11,(R1)      ;LOOPBACK CONNECTOR CN
4294 026032 004737 027464      JSR      PC,SYNLD      ;SET LINE UNIT LOOP
                     ;LOAD 2 SYNCs

```

4295	026036	004737	027464		JSR	PC,SYNLD	:LOAD 2 MORE SYNC
4296	026042	012737	120001	027316	MOV	#CRC16,XPOLY	:LOAD POLYNOMIAL FOR SOFT CRC CALC
4297	026050	005037	026100		CLR	6\$	:CLEAR OLD BCC
4298	026054	012703	000073		MOV	#59,R3	:CHARACTER COUNT
4299	026060	012702	030102		MOV	#MESDAT,R2	:R2= POINTER
4300	026064	112237	026076		7\$: MOVB	(R2)+,5\$	:LOAD CHAR FOR SOFT BCC CALC.
4301	026070	004537	027212		JSR	R5,SIMBCC	:CALC SOFT BCC
4302	026074	000010			IO		:SHIFT COUNT
4303	026076	000000			5\$: 0		:CHARACTER
4304	026100	000000			6\$: 0		:OLD BCC
4305	026102	013737	027320	026100	MOV	CALBCC,6\$	:LOAD OLD BCC
4306	026110	005303			DEC	R3	:DEC COUNT
4307	026112	001364			BNE	7\$	:BR IF NOT DONE YET
4308	026114	004537	027620		JSR	R5,MESLD	:LOAD SILO
4309	026120	030102			MESDAT		:MESSAGE ADDRESS
4310	026122	000073			59.		:CHARACTER COUNT
4311	026124	004737	027574		JSR	PC,EOM	:LOAD AN EOM
4312	026130	004737	026350		JSR	PC,OCOR	:WAIT FOR OCOR
4313	026134	005011			CLR	(R1)	:CLEAR LINE UNIT LOOP
4314	026136	012700	000073		MOV	#59,R0	:R0= CHARACTER COUNT
4315	026142	012702	030102		MOV	#MESDAT,R2	:LOAD MESSAGE POINTER IN R2
4316	026146	004737	027156		1\$: JSR	PC,INRDY	:WAIT FOR INRDY
4317	026152	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4318	026154	021204			021204		:GET DATA FROM IN SILO
4319	026156	016104	000004		MOV	4(R1),R4	:PUT CHARACTER IN "FOUND"
4320	026162	112205			MOVB	(R2)+,R5	:PUT "EXPECTED" IN R5
4321	026164	120504			CMPB	R5,R4	:IS RECEIVED DATA CORRECT
4322	026166	001401			BEQ	2\$	:BR IF OK
4323	026170	104025			HLT	25	:DATA ERROR
4324	026172				2\$: DEC	R0	:DECREMENT COUNTER
4325	026172	005300			BNE	1\$	:BR IF NOT DONE
4326	026174	001364					
4327							
4328							
4329							
4330							
4331	026176	004737	027156		JSR	PC,INRDY	:WAIT FOR INRDY
4332	026202	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4333	026204	021204			021204		:GET FIRST HALF OF CRC
4334	026206	116137	000004	001252	MOVB	4(R1),TEMP3	:PUT IN TEMP3
4335	026214	042737	177400	001252	BIC	#177400,TEMP3	:CLEAR HI BYTE
4336	026222	004737	027156		JSR	PC,INRDY	:WAIT FOR INRDY
4337	026226	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4338	026230	021244			021244		
4339	026232	016104	000004		MOV	4(R1),R4	:PUT "FOUND" IN R4
4340	026236	042704	000376		BIC	#376,R4	:CLEAR UNWANTED BITS
4341	026242	012705	000001		MOV	#1,R5	:PUT "EXPECTED" IN R5
4342	026246	120504			CMPB	R5,R4	:IS IN BCC MATCH SET
4343	026250	001401			BEQ	25\$	
4344	026252	104015			HLT	15	:IN BCC MATCH ERROR
4345	026254				25\$: ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4346	026254	104414			021204		:GET LAST HALF
4347	026256	021204			MOVB	4(R1),TEMP2+1	:PUT IN TEMP2
4348	026260	116137	000004	001251	BIC	#377,TEMP2	:CLEAR LO BYTE
4349	026266	042737	000377	001250	BIS	TEMP2,TEMP3	:15 BIT BCC NOW IN TEMP3
4350	026274	053737	001250	001252			

4351	026302	023737	027320	001252		CMP	CALBCC,TEMP3	;IS IT CORRECT?
4352	026310	001401				BEO	3\$	;BR IF OK
4353	026312	104027				HLT	27	
4354	026314	104400			3\$:	SCOPE		;SCOPE THIS TEST
4355				00300				
4356				00400				
4357				00500		:SUBROUTINES		
4358				00600		:-----		
4359				00700				
4360	026316			00800		GETSI:		
4361				00900				;THIS SUBROUTINE READS LU 17, AND PUTS IT INTO NITCH.
4362				01000				;NITCH IS ROTATED LEFT UNTILL THE SI BIT IS IN CARRY
4363				01100				
4364	026316	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4365	026320	021364		01300		021364		;PORT4+LU 17
4366	026322	017737	153064	026346	01400	MOV	@DMP04,NITCH	;STORE LU 17
4367	026330	106137	026346		01500	ROLB	NITCH	
4368	026334	106137	026346		01600	ROLB	NITCH	
4369	026340	106137	026346		01700	ROLB	NITCH	;PUT SI IN THE CARRY BIT
4370	026344	000207			01800	RTS	PC	
4371	026346	000000			01900	NITCH:	0	
4372				02000				
4373				02100				
4374	026350			02200		OCOR:		
4375				02300				;THIS SUBROUTINE SPINS ON OCOR
4376				02400				
4377	026350	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4378	026352	021364		02600		021364		;PORT4+LU 17
4379	026354	032777	000020	153030	02700	BIT	#BIT4,@DMP04	;IS OCOR SET?
4380	026362	001772			02800	BEO	OCOR	;BR IF NO
4381	026364	000207			02900	RTS	PC	;OK OCOR IS SET, GO BACK
4382				03000				
4383				03100				
4384	026366			03200		SYNC:		
4385				03300				;THIS SUBROUTINE LOADS THE SILO WITH THE NUMBER OF SYNC
4386				03400				;CHARACTERS PASSED TO IT IN THE WORD AFTER THE JSR CALL
4387				03500				;AND A NON-SYNC CHARACTER (301)
4388				03600				
4389	026366	013637	001246		03700	MOV	@(SP)+,TEMP1	;GET COUNT
4390	026372	062746	000002		03800	ADD	#2,-(SP)	;ADJUST STACK
4391	026376	012761	000026	000004	03900	MOV	#26,4(R1)	;LOAD PORT4
4392	026404	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4393	026406	122114			04100	122114		;LOAD SYNC REGISTER
4394	026410	004737	026502		04200	JSR	PC,OUTRDY	;WAIT FOR OUTRDY
4395	026414	012761	000001	000004	04300	MOV	#1,4(R1)	;LOAD PORT4
4396	026422	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4397	026424	122111			04500	122111		;SET SOM
4398	026426	012761	000026	000004	04600	MOV	#26,4(R1)	;LOAD PORT4
4399	026434	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4400	026436	122110			04800	122110		;LOAD OUT DATA
4401	026440	005337	001246		04900	DEC	TEMP1	;ALL DONE?
4402	026444	001361			05000	BNE	1\$	;BR IF NOT
4403	026446	004737	026502		05100	JSR	PC,OUTRDY	;WAIT FOR OUTRDY
4404	026452	005061	000004		05200	CLR	4(R1)	;LOAD PORT4
4405	026456	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4406	026460	122111			05400	122111		;SET SOM

4407	026462	012761	000301	000004	05500	MOV	#301,4(R1)	;LOAD PORT4
4408	026470	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4409	026472	122110			05700	122110		;LOAD OUT DATA
4410	026474	004737	026350		05800	JSR	PC,OCOR	;WAIT FOR OCOR
4411	026500	000207			05900	RTS	PC	
4412					06000			
4413					06100			
4414	026502				06200			
4415					06300			
4416					06400			
4417	026502	005037	001256		06500	CLR	TEMP5	;CLEAR TIMER
4418	026506				06600	1\$:		
4419	026506	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4420	026510	021224			06800	021224		;PORT4-LU11
4421	026512	032777	000020	152672	06900	BIT	#BIT4,ADMP04	;IS OUT RDY SET?
4422	026520	001004			07000	BNE	2\$	;BR IF YES
4423	026522	005237	001256		07100	INC	TEMP5	;INC TIMER
4424	026526	001367			07200	BNE	1\$	;KEEP CHECKING IF NOT DONE
4425	026530	104036			07300	HLT	36	;ERROR, OUT READY NOT SET
4426	026532	000207			07400	2\$:		
4427					07500	RTS	PC	
4428					07600			
4429	026534				07700			
4430					07800	CHAR:		
4431					07900			;THIS SUBROUTINE LOADS THE SILO WITH 3 SYNC
4432					08000			;AND THE CHARACTER PASSED TO IT.
4433	026534	013637	001250		08100	MOV	2(SP)+,TEMP2	;GET CHARACTER
4434	026540	062746	000002		08200	ADD	#2,-(SP)	;ADJUST STACK
4435	026544	012737	000003	001246	08300	MOV	#3,TEMP1	;SET FOR 3 SYNC
4436	026552	012761	000026	000004	08400	MOV	#26,4(R1)	;LOAD PORT4
4437	026560	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4438	026562	122114			08600	122114		;LOAD SYNC REGISTER
4439	026564	004737	026502		08700	JSR	PC,OUTRDY	;WAIT FOR OUTRDY
4440	026570	012761	000001	000004	08800	MOV	#1,4(R1)	;LOAD PORT4
4441	026576	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4442	026600	122111			09000	122111		;SET SOM
4443	026602	012761	000026	000004	09100	MOV	#26,4(R1)	;LOAD PORT4
4444	026610	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4445	026612	122110			09300	122110		;LOAD OUT DATA
4446	026614	005337	001246		09400	DEC	TEMP1	;ALL DONE?
4447	026620	001361			09500	BNE	1\$	;BR IF NOT
4448	026622	004737	026502		09600	JSR	PC,OUTRDY	;WAIT FOR OUTRDY
4449	026626	013761	001250	000004	09700	MOV	TEMP2,4(R1)	;LOAD PORT4
4450	026634	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4451	026636	122110			09900	122110		;LOAD OUT DATA
4452	026640	004737	026350		10000	JSR	PC,OCOR	;WAIT FOR OCOR
4453	026644	000207			10100	RTS	PC	
4454					10200			
4455					10300			
4456	026646				10400			
4457					10500			
4458					10600			
4459	026646	013637	001250		10700	MOV	2(SP)+,TEMP2	;GET CHARACTER
4460	026652	062746	000002		10800	ADD	#2,-(SP)	;ADJUST STACK
4461	026656	004737	026502		10900	JSR	PC,OUTRDY	;WAIT FOR OUTRDY
4462	026662	013761	001250	000004	11000	MOV	TEMP2,4(R1)	;LOAD PORT4

4463	026670	104414			11200	ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4464	026672	122110			11300	122110		;LOAD OUT DATA
4465	026674	004737	026502			JSR	PC,OUTRDY	;WAIT FOR OUTRDY
4466	026700	104414			11500	ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4467	026702	122110			11600	122110		;LOAD GARBAGE CHAR
4468	026704	004737	026350		11700	JSR	PC,OCOR	;WAIT FOR OCOR
4469	026710	000207			11800	RTS	PC	
4470					11900			
4471					12000			
4472	026712				12100			
4473					12200			
4474					12300			
4475					12400			
4476	026712	012737	000073	001250	12500	MOV	#73,TEMP2	;LOAD COUNT
4477	026720	005737	027152		12600	TST	SCHAR	;FIRST TIME HERE?
4478	026724	100470			12700	BMI	4\$	;BR IF BITSTUFF
4479	026726	001032			12800	BNE	2\$	;BR IF NO
4480	026730	062737	000002	001250	12900	ADD	#2,TEMP2	;ADD 2 TO CHARACTER COUNT
4481	026736	012737	000003	001246	13000	MOV	#3,TEMP1	;SET FOR 3 SYNC
4482	026744	012761	000026	000004	13200	MOV	#26,4(R1)	;LOAD PORT4
4483	026752	104414			13300	ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4484	026754	122114			13400	122114		;LOAD SYNC REGISTER
4485	026756	004737	026502			JSR	PC,OUTRDY	;WAIT FOR OUTRDY
4486	026762	012761	000001	000004		MOV	#1,4(R1)	;LOAD PORT4
4487	026770	104414			13600	ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4488	026772	122111			13700	122111		;SET SOM
4489	026774	012761	000026	000004		MOV	#26,4(R1)	;LOAD PORT4
4490	027002	104414			13900	ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4491	027004	122110			14000	122110		;LOAD OUT DATA
4492	027006	005337	001246		14100	DEC	TEMP1	;ALL DONE?
4493	027012	001361			14200	BNE	1\$	;BR IF NOT
4494	027014	004737	026502		14300	JSR	PC,OUTRDY	;WAIT FOR OUTRDY
4495	027020	013761	027152	000004		MOV	SCHAR,4(R1)	;LOAD PORT4
4496	027026	104414			14500	ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4497	027030	122110			14600	122110		;LOAD OUT DATA
4498	027032	005737	027154		14700	TST	STUFLG	;BITSTUFF???
4499	027036	001407			14800	BEQ	6\$	;BR IF NO
4500	027040	013737	027152	027052	14900	MOV	SCHAR,5\$	;IT IS SOLD SO CHECK BITSTUFFING
4501	027046	004537	027702		15000	JSR	RS,STFFCL	;ADD ANY BIT STUFF CLOCK TICKS
4502	027052	000000			15100	0		;CHARACTER
4503	027054	000010			15200	10		;CHIFT COUNT
4504	027056	005237	027152		15300	INC	SCHAR	;NEXT CHARACTER
4505	027062	022737	000400	027152	15400	CMP	#400,SCHAR	;ALL DONE?
4506	027070	001403			15500	BEQ	3\$	
4507	027072	005337	001250		15600	DEC	TEMP2	;DECREMENT COUNT
4508	027076	001346			15700	BNE	2\$	;BR IF NOT DONE
4509	027100	004737	026350		15800	JSR	PC,OCOR	;WAIT FOR OCOR
4510	027104	000207			15900	RTS	PC	
4511	027106	005037	027152		16000	CLR	SCHAR	;START PATTERN AT ZERO
4512	027112	012737	177777	027154	16100	MOV	#-1,STUFLG	;SET BITSTUFF FLAG
4513	027120	005037	030100		16200	CLR	BITCON	;CLEAR STUFF COUNT
4514	027124	062737	000002	001250	16300	ADD	#2,TEMP2	;ADD 2 TO CHARACTER COUNT
4515	027132	012761	000001	000004		MOV	#1,4(R1)	;SET BIT0 IN PORT4
4516	027140	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4517	027142	122111				122111		;SET SOM!
4518	027144	104414				ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

SILOLD:

;THIS SUBROUTINE FILLS THE OUT SILO
; WITH A BINARY COUNT PATTERN

1\$:

2\$:

5\$:

6\$:

3\$:

4\$:

DZDME MACY11 30(1046) 11-JUL-77 11:40 PAGE 85
 DZDME.P11 12-MAY-77 14:18 SUBROUTINES

```

4519 027146 122110 16700 122110 ;LOAD GARBAGE CHAR
4520 027150 000721 16800 BR 2$ ;GO LOAD SILO
4521 027152 000000 16900 SCHAR: 0
4522 027154 000000 17000 STUFLG: 0
4523 17100
4524 17200
4525 027156 17300 INRDY:
4526 17400 ;THIS SUBROUTINE SPINS ON INRDY
4527 17500 ;IF INRDY FAILS TO SET THE DELAY TIMES OUT AND AN
4528 17600 ;ERROR IS REPORTED. FOR BETTER SCOPE LOOPS THIS
4529 17700 ;DELAY CAN BE MADE SHORTER BY ALTERING THE NUMBER
4530 17800 ;INITIALLY LOADED INTO TEMP1. THE SMALLER THE NUMBER
4531 17900 ;THE SHORTER THE DELAY. 0 IS THE LONGEST DELAY.
4532 18000
4533 027156 012737 000000 001246 18100 1$: MOV #0,TEMP1 ;SET UP DELAY COUNTER
4534 027164 18200 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4535 027164 104414 021244 ;PORT4+LUI2
4536 027166 021244 000020 152214 18400 BIT #BIT4,ADMP04 ;IS INRDY SET?
4537 027170 032777 18500 BNE 2$ ;BR IF YES
4538 027176 001004 18600 INC TEMP1 ;INC DELAY
4539 027200 005237 001246 18700 BNE 1$ ;TRY AGAIN
4540 027204 001367 18800 HLT 37 ;ERROR, NO INRDY
4541 027206 104037 18900 2$: RTS PC ;RETURN
4542 027210 000207 19000
4543 19100
4544 19200
4545 027212 SIMBCC:
4546 ;THIS SUBROUTINE CALCULATES THE CRC USING POLYNOMIAL GIVEN
4547 ;IN XPOLY. THE CORRECT CRC IS RETURNED IN CALBCC, AND THE
4548 ;STATE OF THE LSB OF THE BCC IS RETURNED IN THE C BIT.
4549
4550 027212 010046 MOV RO, -(SP) ;SAVE RO ON STACK
4551 027214 012537 001246 MOV (R5)+,TEMP1 ;TEMP1 = SHIFT COUNT
4552 027220 012537 001250 MOV (R5)+,TEMP2 ;TEMP2 = CHARACTER
4553 027224 012537 027320 MOV (R5)+,CALBCC ;CALBCC = OLD BCC
4554 027230 013700 027320 1$: MOV CALBCC,RO ;PUT OLD BCC IN RO
4555 027234 000241 CLC
4556 027236 006037 027320 ROR CALBCC ;SHIFT OLD BCC
4557 027242 006037 001250 ROR TEMP2 ;SHIFT CHARACTER
4558 027246 005500 ADC RO ;ADD CHAR CARRY TO OLD BCC
4559 027250 006000 ROR RO ;PUT BIT0 TO CARRY BIT
4560 027252 103011 BCC 2$ ;CARRY IS FEEDBACK BIT
4561 027254 013700 027316 MOV XPOLY,RO ;IF FEEDBACK = 1
4562 027260 043700 027320 BIC CALBCC,RO ;EXCLUSIVLY OR XPOLY TO CALBCC
4563 027264 043737 027316 BIC XPOLY,CALBCC
4564 027272 050037 027320 BIS RO,CALBCC
4565 027276 005337 001246 2$: DEC TEMP1 ;DEC SHIFT COUNT
4566 027302 001352 BNE 1$ ;BR IF NOT DONE
4567 027304 013700 027320 MOV CALBCC,RO ;PUT RESULT IN RO
4568 027310 006000 ROR RO ;SHIFT BIT0 TO CARRY
4569 027312 012600 MOV (SP)+,RO ;RESTORE RO
4570 027314 000205 RTS RS ;RETURN
4571 027316 000000 XPOLY: 0
4572 027320 000000 CALBCC: 0
4573 000200 LRC8=200
4574 120001 CRC16=120001

```

4575		102010			CRC.CCITT=102010	
4576						
4577						
4578	027322			19800	BCCLD:	
4579				19900		: THIS SUBROUTINE LOADS THE OUT SILO WITH 2 SYNCs
4580				20000		: WITH SOM SET, AND ONE CHARACTER PASSED TO IT
4581				20100		: WITH THE SOM BIT CLEAR (ENABLE CRC)
4582				20200		
4583	027322	013637	001250	20300	MOV	2(SP)+,TEMP2 ;GET CHARACTER
4584	027326	062746	000002	20400	ADD	2,-(SP) ;ADJUST STACK
4585	027332	012737	000002	20500	MOV	2,TEMP1 ;SET FOR 2 SYNCs
4586	027340	012761	000026	20600	MOV	26,4(R1) ;LOAD PORT4
4587	027346	104414			ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4588	027350	122114		20800	122114	;LOAD SYNC REGISTER
4589	027352	004737	026502	20900	15: JSR	PC,OUTRDY ;WAIT FOR OUTRDY
4590	027356	012761	000001	21000	MOV	1,4(R1) ;LOAD PORT4
4591	027364	104414			ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4592	027366	122111		21200	122111	;SET SOM
4593	027370	012761	000026	21300	MOV	26,4(R1) ;LOAD PORT4
4594	027376	104414			ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4595	027400	122110		21500	122110	;LOAD OUT DATA
4596	027402	005337	001246	21600	DEC	TEMP1 ;ALL DONE?
4597	027406	001361		21700	BNE	15 ;BR IF NOT
4598	027410	004737	026502	21800	JSR	PC,OUTRDY ;WAIT FOR OUTRDY
4599	027414	013761	001250	21900	MOV	TEMP2,4(R1) ;LOAD PORT4
4600	027422	104414			ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4601	027424	122110		22100	122110	;LOAD OUT DATA
4602	027426	004737	026350	22200	JSR	PC,OCOR ;WAIT FOR OCOR
4603	027432	000207		22300	RTS	PC
4604				22400		
4605				22500		
4606	027434				GETQO:	
4607						: THIS SUBROUTINE READS THE STATE OF THE TRANSMIT
4608						: BCC LSB AND PUTS IT IN THE CARRY BIT
4609						
4610	027434	104414			ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4611	027436	021364			021364	;PORT4+LU-17
4612	027440	106177	151746		ROLB	2DMP04 ;PUT QO IN CARRY
4613	027444	000207			RTS	PC ;RETURN
4614						
4615						
4616	027446				GETQI:	
4617						: THIS SUBROUTINE READS THE STATE OF THE RECEIVE
4618						: BCC LSB AND PUTS IT IN THE CARRY BIT
4619						
4620	027446	104414			ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4621	027450	021364			021364	;PORT4+LU-17
4622	027452	106177	151734		ROLB	2DMP04 ;PUT QO IN CARRY
4623	027456	106177	151730		ROLB	2DMP04 ;PUT QI IN CARRY
4624	027462	000207			RTS	PC ;RETURN
4625						
4626						
4627	027464			22800	SYNLD:	
4628				22900		: THIS SUBROUTINE LOADS OUT SILO WITH
4629				23000		: 2 SYNC CHARACTERS WITH SOM SET
4630				23100		

4631	027464	012737	000002	001246	23200
4632	027472	012761	000026	000004	23300
4633	027500	104414			
4634	027502	122114			23500
4635	027504	004737	026502		23600
4636	027510	012761	000001	000004	23700
4637	027516	104414			
4638	027520	122111			23900
4639	027522	012761	000026	000004	24000
4640	027530	104414			
4641	027532	122110			24200
4642	027534	005337	001246		24300
4643	027540	001361			24400
4644	027542	000207			24500
4645					24600
4646					24700
4647	027544				24800
4648					24900
4649					25000
4650					25100
4651	027544	004737	026502		25200
4652	027550	012761	000001	000004	25300
4653	027556	104414			
4654	027560	122111			25500
4655	027562	005061	000004		25600
4656	027566	104414			
4657	027570	122110			25800
4658	027572	000207			25900
4659					26000
4660					26100
4661	027574				26200
4662					26300
4663					26400
4664					26500
4665	027574	004737	026502		26600
4666	027600	012761	000002	000004	26700
4667	027606	104414			
4668	027610	122111			26900
4669	027612	104414			
4670	027614	122110			27100
4671	027616	000207			27200
4672					27300
4673					27400
4674	027620				27500
4675					27600
4676					27700
4677					27800
4678					27900
4679	027620	010046			28000
4680	027622	012500			28100
4681	027624	012537	001246		28200
4682	027630	004737	026502		28300
4683	027634	112061	000004		28400
4684	027640	104414			
4685	027642	122110			28600
4686	027644	005337	001246		28700

```

MOV    #2,TEMP1      ;LOAD COUNTER FOR 2 SYNCs
MOV    #26,4(R1)      ;PORT4+26
ROMCLK 122114         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
JSR    PC,OUTRDY      ;LOAD SYNC REG
MOV    #1,4(R1)       ;WAIT FOR OUTRDY
ROMCLK 122111         ;LOAD PORT4
MOV    #26,4(R1)      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
ROMCLK 122111         ;SET SOM
DEC    TEMP1          ;PORT+26
BNE    1$             ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
RTS    PC              ;LOAD OUT DATA WITH SYNC
                     ;DECREMENT COUNTER
                     ;BR IF NOT DONE
                     ;RETURN

```

SOM:

```

;THIS SUBROUTINE LOADS SOM AND OUT DATA WITH A
;GARBAGE CHARACTER (0)

```

```

JSR    PC,OUTRDY      ;WAIT FOR OUTRDY
MOV    #1,4(R1)       ;PORT4+1
ROMCLK 122111         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
CLR    4(R1)          ;SET SOM
ROMCLK 122110         ;CLEAR DATA CHAR
RTS    PC              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
                     ;LOAD GARBAGE CHARACTER
                     ;RETURN

```

EOM:

```

;THIS SUBROUTINE LOADS EOM AND OUT DATA WITH A
;GARBAGE CHARACTER (2) TO ENABLE TRANSMISSION OF ECC

```

```

JSR    PC,OUTRDY      ;WAIT FOR OUTRDY
MOV    #2,4(R1)       ;PORT4+2
ROMCLK 122111         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
CLR    4(R1)          ;SET EOM
ROMCLK 122110         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
RTS    PC              ;LOAD GARBAGE CHARACTER
                     ;RETURN

```

MESLD:

```

;THIS SUBROUTINE LOADS SILO WITH MESSAGE
;THE FIRST ARGUMENT IS THE ADDRESS OF THE MESSAGE
;THE SECOND ARGUMENT IS THE NUMBER OF CHARACTERS IN THE MESSAGE

```

```

MOV    R0,-(SP)       ;SAVE R0
MOV    (R5)+,R0       ;R0=MESSAGE POINTER
MOV    (R5)+,TEMP1    ;TEMP1=CHARACTER COUNT
JSR    PC,OUTRDY      ;WAIT FOR OUT RDY
MOV    (R0)+,4(R1)    ;LOAD PORT4 WITH CHARACTER
ROMCLK 122110         ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
DEC    TEMP1          ;LOAD OUT DATA SILO
                     ;DEC CHAR COUNT

```

4687	027650	001367		28800	BNE	1\$	;BR IF NOT DONE
4688	027652	004737	026350	28900	JSR	PC,OCOR	;WAIT FOR OCOR
4689	027656	012600		29000	MOV	(SP)+,RO	;RESTORE RO
4690	027660	000205		29100	RTS	R5	;RETURN
4691				29200			
4692				29300			
4693	027662			29400	CLRIO:		
4694				29500			;THIS SUBROUTINE SETS IN CLR AND OUT CLR TO
4695				29600			;CLEAR THE TRANSMIT AND RECEIVE BCC REGISTERS
4696				29700			
4697	027662	012761	000200 000004	29800	MOV	#BIT7,4(R1)	;LOAD PORT4
4698	027670	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4699	027672	122112		30000	122112		;SET IN CLR!
4700	027674	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4701	027676	122111		30200	122111		;SET OUT CLR!
4702	027700	000207		30300	RTS	PC	;RETURN
4703				30400			
4704				30500			
4705	027702			30600	STFFCL:		
4706				30700			;THIS SUBROUTINE ADDS ANY NECESSARY BIT STUFF CLOCK TICKS
4707				30800			;FIRST ARGUMENT IS CHAR, SECOND ARGUMENT IS SHIFT COUNT.
4708				30900			
4709	027702	010046		310	MOV	RO, -(SP)	;SAVE RO
4710	027704	012500		31100	MOV	(R5)+,RO	;PUT CHAR IN RO
4711	027706	012537	001252	31200	MOV	(R5)+,TEMP3	;PUT SHIFT COUNT IN TEMP3
4712	027712	106000		31300	1\$:	RORB	RO
4713	027714	103403		31400			;LOOK AT NEXT BIT
4714	027716	005037	030100	31500	BCS	2\$	;BR IF A MARK
4715	027722	000412		31600	CLR	BITCON	;IT WAS A SPACE, CLEAR I'S COUNTER
4716	027724	005237	030100	31700	BR	3\$	;CONTINUE
4717	027730	022737	000005 030100	31800	2\$:	INC	BITCON
4718	027736	001004		31900			;INC CONSECUTIVE I'S COUNTER
4719	027740	005037	030100	32000	CMP	#5,BITCON	;IS IT 5 YET?
4720	027744	104415	000001	32100	BNE	3\$	;BR IF NO
4721	027750	005337	001252	32200	CLR	BITCON	;YES! SO START AGAIN
4722	027754	001356		32300	DATACLK,	1	;GIVE EXTRA TICK TO STUFF ZERO
4723	027756	012600		32400	3\$:	DEC	TEMP3
4724	027760	000205		32500			;DEC SHIFT COUNT
4725				32600	BNE	1\$	;BR IF NOT DONE
4726				32700	MOV	(SP)+,RO	;RESTORE RO
4727	027762			32800	RTS	R5	;RETURN
4728				32900			
4729				33000	STFFCK:		
4730				33100			;THIS SUBROUTINE CHECKS TO SEE IF TRANSMITTER
4731				33200			;IS STUFFING ZEROS WHEN IT SHOULD. FIRST ARGUMENT
4732	027762	010046		33300			;IS THE CHARACTER, SECOND ARGUMENT IS SHIFT COUNT.
4733	027764	012500		33400	MOV	RO, -(SP)	;SAVE RO
4734	027766	012537	001252	33500	MOV	(R5)+,RO	;PUT CHAR IN RO
4735	027772	106000		33600	MOV	(R5)+,TEMP3	;PUT SHIFT COUNT IN TEMP3
4736	027774	103403		33700	1\$:	RORB	RO
4737	027776	005037	030100	33800			;SHIFT OUT NEXT BIT
4738	030002	000416		33900	BCS	2\$	;BR IF IT IS A MARK
4739	030004	005237	030100	34000	CLR	BITCON	;IT WAS A SPACE, CLEAR I'S COUNTER
4740	030010	022737	000005 030100	34100	BR	3\$	;CONTINUE
4741	030016	001010		34200	2\$:	INC	BITCON
4742	030020	005037	030100	34300			;INC CONSECUTIVE I'S COUNTER
					CMP	#5,BITCON	;IS IN A ROW YET?
					BNE	3\$	;BR IF NO
					CLR	BITCON	;YES, SO START OVER

4743	030024	104415	000001	34400	DATA CLK, 1	: EXTRA TICK TO STUFF ZERO
4744	030030	004737	026316	34500	JSR PC, GETSI	: LOOK AT WINDOW
4745	030034	103001		34600	BCC 3\$	: IS IT A ZERO, BR IF YES
4746	030036	104030		34700	HLT 30	: NO, ERROR ZERO WAS NOT STUFFED
4747	030040	005337	001252	34800	3\$: DEC TEMP3	: DEC SHIFT COUNT
4748	030044	001352		34900	BNE 1\$	: BR IF NOT DONE
4749	030046	012600		35000	MOV (SP)+, R0	: RESTORE R0
4750	030050	000205		35100	RTS R5	: RETURN
4751				35200		
4752				35300		
4753	030052			35400	CTSDLY:	
4754				35500		: THIS SUBROUTINE WASTES TIME UNTIL CTS SETS,
4755				35600		: BUT HOPEFULLY NOT SO LONG THAT THE SILO RUNS OUT
4756				35700		
4757	030052	010046		35800	MOV R0, -(SP)	: SAVE R0
4758	030054	012700	000032	35900	MOV #32, R0	: LOAD R0 WITH COUNT
4759	030060	027777	151120	36000	1\$: CMP @TKCSR, @TKCSR	: WASTE TIME
4760	030066	005300		36100	DEC R0	: DECREMENT COUNTER
4761	030070	001373		36200	BNE 1\$	: DO IT AGAIN IF NOT = 0
4762	030072	012600		36300	MOV (SP)+, R0	: RESTORE R0
4763	030074	000207		36400	RTS PC	: RETURN
4764				36500		
4765				36600		
4766	030076	000176		36700	FLAG: 1B<01111110>	: FLAG CHARACTER
4767	030100	000000		36800	BITCON: 0	
4768	030102	000	125	36900	MESDAT: .BYTE 0, 125, 252, 377	
4769	030105	377				
4770	030106	001	002	004	37000	FLTDAT: .BYTE 1, 2, 4, 10, 20, 40, 100, 200, 376, 375, 373, 367, 357, 337, 277, 177
4771	030111	010	020	040		
4772	030114	100	200	376		
4773	030117	375	373	367		
4774	030122	357	337	277		
4775	030125	177				
4776	030126	100	140	160	37100	STUFDT: .BYTE 100, 140, 160, 170, 3, 300, 174, 176, 177, 1
4777	030131	170	003	300		
4778	030134	174	176	177		
4779	030137	001				
4780	030140	363	347	317	37200	.BYTE 363, 347, 317, 200, 0, 377, 377, 377, 200, 37
4781	030143	200	000	377		
4782	030146	377	377	200		
4783	030151	037				
4784						
4785	030152	046377	047111	020105	00300	.EVEN
	030210	046377	047111	020105	00400	EM1: .ASCIZ <377>/LINE UNIT INITIALIZATION TEST/
	030253	377	044514	042516	00500	EM2: .ASCIZ <377>/LINE UNIT REGISTER READ/ONLY TEST/
	030317	377	044514	042516	00600	EM3: .ASCIZ <377>/LINE UNIT REGISTER WRITE/READ TEST/
	030361	377	051124	047101	00700	EM4: .ASCIZ <377>/LINE UNIT INTERNAL CLOCK FAILURE
	030411	377	042522	042503	00800	EM5: .ASCIZ <377>/TRANSMITTER DATA ERROR/
	030430	051377	041505	044505	00900	EM6: .ASCIZ <377>/RECEIVER TEST/
	030455	377	047515	042504	01000	EM7: .ASCIZ <377>/RECEIVER DATA ERROR/
	030501	377	051124	047101	01100	EM10: .ASCIZ <377>/MODEM SIGNAL ERROR/
	030530	051377	041505	044505	01200	EM11: .ASCIZ <377>/TRANSMITTER CRC ERROR
	030554	044777	020116	041502	01300	EM12: .ASCIZ <377>/RECEIVER CRC ERROR/
	030614	052377	040522	051516	01400	EM13: .ASCIZ <377>/IN BCC MATCH ERROR (LU REG 12)
	030664	041777	041101	042514	01500	EM14: .ASCIZ <377>/TRANSMITTER FAILED TO GO TO MARK STATE
	030705	377	046106	043501	01600	EM15: .ASCIZ <377>/CABLE DATA TEST
						EM16: .ASCIZ <377>/FLAG ERROR

030721	377	051124	047101	01700	EM17:	.ASCIZ	<377>/TRANSMITTER FAILED TO STUFF A ZERO/
030765	377	053523	052111	01800	EM20:	.ASCIZ	<377>/SWITCH PAC TEST/
031006	040777	047502	052122	01900	EM21:	.ASCIZ	<377>/ABORT ERROR/
031023	377	051124	047101	02000	EM22:	.ASCIZ	<377>/TRANSMITTER ERROR/
031046	044377	046101	020106	02100	EM23:	.ASCIZ	<377>/HALF DUPLEX TEST/
031070	047777	052125	051040	02200	EM24:	.ASCIZ	<377>/OUT READY NOT SET/
031113	377	047111	051040	02300	EM25:	.ASCIZ	<377>/IN READY NOT SET/
				02400			
031135	377	054105	042520	02500	DH1:	.ASCIZ	<377>/EXPECTED FOUND/
031156	042777	050130	041505	02600	DH2:	.ASCIZ	<377>/EXPECTED FOUND LU-REGISTER/
031214	041777	040510	040522	02700	DH3:	.ASCIZ	<377>/CHARACTER BIT THAT FAILED/
031252	041777	051117	042522	02800	DH4:	.ASCIZ	<377>/CORRECT CRC BIT THAT FAILED/
031312	042777	050130	041505	02900	DH5:	.ASCIZ	<377>/EXPECTED FOUND SHIFT/
031344	042777	050130	041505	03000	DH6:	.ASCIZ	<377>/EXPECTED FOUND CHARACTER SHIFT/
031412	041377	047514	045503	03100	DH7:	.ASCIZ	<377>/BLOCK END NOT SET/
031435	377	052122	020123	03200	DH10:	.ASCIZ	<377>/RTS DID NOT CLEAR/
				03300			
				03400			
031460	000002			03500	DT1:	2	
031462	003	007		03600		.BYTE	3.7
031464	001272			03700		SAVR5	
031466	003	002		03800		.BYTE	3.2
031470	001270			03900		SAVR4	
031472	000003			04000	DT2:	3	
031474	003	007		04100		.BYTE	3.7
031476	001272			04200		SAVR5	
031500	003	010		04300		.BYTE	3.10
031502	001270			04400		SAVR4	
031504	003	002		04500		.BYTE	3.2
031506	001264			04600		SAVR2	
031510	000002			04700	DT3:	2	
031512	003	017		04800		.BYTE	3.17
031514	001272			04900		SAVR5	
031516	002	002		05000		.BYTE	2.2
031520	001266			05100		SAVR3	
031522	000002			05200	DT4:	2	
031524	006	021		05300		.BYTE	6.21
031526	027320			05400		CALBCC	
031530	002	002		05500		.BYTE	2.2
031532	001266			05600		SAVR3	
031534	000003			05700	DT5:	3	
031536	001	011		05800		.BYTE	1.11
031540	001300			05900		ZERO	
031542	001	011		06000		.BYTE	1.11
031544	001302			06100		ONE	
031546	002	002		06200		.BYTE	2.2
031550	001260			06300		SAVR0	
031552	000003			06400	DT6:	3	
031554	001	011		06500		.BYTE	1.11
031556	001302			06600		ONE	
031560	001	011		06700		.BYTE	1.11
031562	001300			06800		ZERO	
031564	002	002		06900		.BYTE	2.2
031566	001260			07000		SAVR0	
031570	000004			07100	DT7:	4	
031572	001	011		07200		.BYTE	1.11

031574	001300		07300	ZERO	
031576	001	011	07400	.BYTE	1,11
031600	001302		07500	ONE	
031602	003	007	07600	.BYTE	3,7
031604	001272		07700	SAVR5	
031606	002	001	07800	.BYTE	2,1
031610	001266		07900	SAVR3	
031612	000004		08000	DT10:	4
031614	001	011	08100	.BYTE	1,11
031616	001302		08200	ONE	
031620	001	011	08300	.BYTE	1,11
031622	001300		08400	ZERO	
031624	003	007	08500	.BYTE	3,7
031626	001272		08600	SAVR5	
031630	002	001	08700	.BYTE	2,1
031632	001266		08800	SAVR3	
031634	000002		08900	DT11:	2
031636	003	007	09000	.BYTE	3,7
031640	030076		09100	FLAG	
031642	002	002	09200	.BYTE	2,2
031644	001266		09300	SAVR3	
031646	000002		09400	DT12:	2
031650	006	004	09500	.BYTE	6,4
031652	027320		09600	CALBCC	
031654	006	002	09700	.BYTE	6,2
031656	001252		09800	TEMP3	
			09900		
031660			10000	.ERRTAB:	
031660	000000		10100	0	
031662	000000		10200	0	
031664	000000		10300	0	
031666	030152		10400	EM1	
031670	031156		10500	DH2	;HLT 1
031672	031472		10600	DT2	
031674	030210		10700	EM2	
031676	031156		10800	DH2	;HLT 2
031700	031472		10900	DT2	
031702	030253		11000	EM3	
031704	031156		11100	DH2	;HLT 3
031706	031472		11200	DT2	
031710	030317		11300	EM4	
031712	000000		11400	0	;HLT 4
031714	000000		11500	0	
031716	030361		11600	EM5	
031720	031156		11700	DH2	;HLT 5
031722	031472		11800	DT2	
031724	030361		11900	EM5	
031726	031214		12000	DH3	;HLT 6
031730	031510		12100	DT3	
031732	030411		12200	EM6	
031734	031135		12300	DH1	;HLT 7
031736	031460		12400	DT1	
031740	030430		12500	EM7	
031742	031135		12600	DH1	;HLT 10
031744	031460		12700	DT1	
031746	030455		12800	EM10	

031750	031135	12900	DH1	;HLT	11
031752	031460	13000	DT1		
031754	030501	13100	EM11		
031756	031312	13200	DH5	;HLT	12
031760	031534	13300	DT5		
031762	030530	13400	EM12		
031764	031312	13500	DH5	;HLT	13
031766	031534	13600	DT5		
031770	030501	13700	EM11		
031772	031252	13800	DH4	;HLT	14
031774	031522	13900	DT4		
031776	030554	14000	EM13		
032000	000000	14100	0	;HLT	15
032002	000000	14200	0		
032004	030501	14300	EM11		
032006	031312	14400	DH5	;HLT	16
032010	031552	14500	DT6		
032012	030530	14600	EM12		
032014	031312	14700	DH5	;HLT	17
032016	031552	14800	DT6		
032020	030501	14900	EM11		
032022	031344	15000	DH6	;HLT	20
032024	031570	15100	DT7		
032026	030501	15200	EM11		
032030	031344	15300	DH6	;HLT	21
032032	031612	15400	DT10		
032034	030530	15500	EM12		
032036	031344	15600	DH6	;HLT	22
032040	031570	15700	DT7		
032042	030530	15800	EM12		
032044	031344	15900	DH6	;HLT	23
032046	031612	16000	DT10		
032050	030614	16100	EM14		
032052	000000	16200	0	;HLT	24
032054	000000	16300	0		
032056	030664	16400	EM15		
032060	031135	16500	DH1	;HLT	25
032062	031460	16600	DT1		
032064	030705	16700	EM16		
032066	031214	16800	DH3	;HLT	26
032070	031634	16900	DT11		
032072	030530	17000	EM12		
032074	031135	17100	DH1	;HLT	27
032076	031646	17200	DT12		
032100	030721	17300	EM17		
032102	000000	17400	0	;HLT	30
032104	000000	17500	0		
032106	030765	17600	EM20		
032110	031135	17700	DH1	;HLT	31
032112	031460	17800	DT1		
032114	031006	17900	EM21		
032116	031412	18000	DH7	;HLT	32
032120	000000	18100	0		
032122	031006	18200	EM21		
032124	031214	18300	DH3	;HLT	33
032126	031510	18400	DT3		

DDOME MACY11 30(1046) 11-JUL-77 11:40 PAGE 93
 DDOME.P11 12-MAY-77 14:18 SUBROUTINES

PAGE: 0112

032130	031023	18500	EM22		
032132	031435	18600	DH10	;HLT	34
032134	000000	18700	0		
032136	031046	18800	EM23		
032140	031156	18900	DH2	;HLT	35
032142	031472	19000	DT2		
032144	031070	19100	EM24		
032146	000000	19200	0	;HLT	36
032150	000000	19300	0		
032152	031113	19400	EM25		
032154	000000	19500	0	;HLT	37
032156	000000	19600	0		
032160	030411	19700	EM6		
032162	031156	19800	DH2	;HLT	40
032164	031472	19900	DT2		
032166	030361	20000	EM5		
032170	031312	20100	DH5	;HLT	41
032172	031534	20200	DT5		
032174	030554	20300	EM13		
032176	031135	20400	DH1	;HLT	42
032200	031460	20500	DT1		
		20600			
		20700			
032202		20800	CORMAX:		
	00000:	21300	.END		

[illegible]

[illegible]

[illegible]

[illegible]

DZDME MACY11 30.1046) 11-JUL-77 11:40 PAGE 99

DZDME.P11 12-MAY-77 14:18

CROSS REFERENCE TABLE -- USER SYMBOLS

GETQI	027446	3026	3030	3101	3105	3176	3180	3251	3255	3372	3376	4616#		
GETQO	027434	2997	3001	3072	3076	3147	3151	3222	3226	3306	3310	4606#		
GETSI	026316	2205	2209	2256	2260	2307	2311	2358	2362	2417	2421	3445	3449	3466
		3470	3482	3491	3610	3614	3631	3635	3661	3665	3682	3686	3698	3707
		3858	3862	3879	3883	3899	3903	3928	3932	3949	3953	3965	3974	4360#
		4744												
HALTS	005222	1033	1079#											
HILIM	004366	876*	903	921#										
ICOUNT	001222	159#	793	798*										
INBUF	007502	846	882	1202#										
INCHAR	010020	1231	1259#											
INIFLG	001324	201#	507	527	534*									
INRDY	027156	2768	2812	2849	2857	2865	3524	3537	3542	3720	3734	3739	3763	3777
		3782	3987	4001	4006	4027	4041	4055	4060	4116	4126	4134	4231	4247
		4252	4316	4331	4336	4525#								
INSTER=	104404	223#	897											
INSTR =	104403	221#	1329	1381	1394	1403	1482	1491						
INSTR2	004166	853	865#											
INTTY	012266	1414	1431	1441	1454	1470	1655#							
KMCM	007330	637	1187#											
LIMITS	004314	892	903#											
LINE	007016	1187#	1483											
LOBITS	004372	878*	907	923#	924									
LOCK	001220	158#	797*	814	816	1056	1779*	1795*	1821*	1837*	1863*	1986*	1921*	1941*
		2978*	3010*	3053*	3085*	3128*	3160*	3203*	3235*					
LOKFLG	001326	203#												
LOLIM	004364	875*	905	920#										
LPCNT	001224	160#	792*	793	796*									
LRCB =	000200	4573#												
LSTERR	001234	164#	490*	732*	1034	1036*	1124*							
LUTYPE=	***** U	1	4545											
MASKX	001244	172#												
MASTEK	006142	1058	1187#											
MCRLF	005672	831	954	1054	1055	1063	1187#	1328	1390					
MCSRX	006072	737	1187#											
MDATA	007544	984	994	1204#										
MEMLIM	001304	188#	700*											
MEPASS	005733	736	1187#											
MERRPC	006217	1061	1187#											
MERRX	006117	743	1187#											
MERR2	005760	1187#	1590											
MERR3	006005	673	1187#											
MESDAT	030102	2949	3417	3423	3514	3519	3578	3584	3588	3644	3719	3762	3825	3831
		3836	3911	3986	4040	4096	4101	4106	4166	4299	4309	4315	4768#	
		2948	3422	3518	3583	3587	3830	3835	4100	4105	4165	4214	4218	4222
MESLD	027620	4308	4674#											
MILK	001322	196#	484*	745	1288*	1293*	1297							
MLOCK	006043	714	1187#											
MNEW	006144	668	1187#											
MODU	006704	1187#	1453											
MPASSX	006106	741	1187#											
MPFAIL	005675	1121	1187#											
MQM	005666	861	1187#	1352	1427	1437	1447	1462	1476					
MR	005755	723	1187#	1346										
MRESET=	004000	96#												
METCLP=	104412	235#	1126	1725	1749	1781	1823	1865	1923	1974	1996	2016	2052	2085

		2129	2187	2238	2289	2340	2392	2450	2478	2506	2534	2563	2611	2648
		2685	2722	2759	2800	2844	2888	2935	2980	3055	3130	3205	3278	3344
		3412	3512	3573	3820	4094	4157	4196	4290					
MTITLE	001000	136#	511											
MTSTN	006130	1059	1187#	1330										
MTSTPC	006031	1187#												
MVECX	006100	739	1187#											
NEXT	001216	157#	799	1092	1676*	1700*	1723*	1747*	1778*	1820*	1862*	1920*	1972*	1994*
		2016*	2050*	2083*	2127*	2185*	2236*	2287*	2338*	2390*	2448*	2476*	2504*	2532*
		2561*	2609*	2646*	2683*	2720*	2757*	2798*	2842*	2886*	2933*	2977*	3052*	3127*
		3202*	3276*	3342*	3410*	3510*	3571*	3818*	4092*	4155*	4194*	4288*		
NITCH	026346	4366*	4367*	4368*	4369*	4371#								
NOACT	007154	523	1187#	1282										
NODEV	002674	577	616#											
NJM	006450	1187#	1382											
OCOR	026350	2092	2136	2198	2249	2300	2351	2410	3297	3363	3426	3522	3591	3839
		4103	4108	4168	4226	4312	4374#	4380	4410	4452	4468	4509	4602	4589
OK	002646	603	610#	639										
ONE	001302	187#	4785											
OUTRDY	026502	2189	2194	2240	2245	2291	2296	2342	2347	2397	2401	2406	2429	3421
		3517	3582	3829	4099	4164	4394	4403	4414#	4439	4448	4461	4465	4485
		4494	4589	4598	4635	4651	4665	4682						
PACT00	001702	365#												
PACT01	001706	368#												
PACT02	001712	371#												
PACT03	001716	374#												
PACT04	001722	377#												
PACT05	001726	380#												
PACT06	001732	383#												
PACT07	001736	386#												
PACT10	001742	389#												
PACT11	001746	392#												
PACT12	001752	395#												
PACT13	001756	398#												
PACT14	001762	401#												
PACT15	001766	404#												

DZOME MACY11 30(1046) 11-JUL-77 11:40 PAGE 101
DZOME.P11 12-MAY-77 14:18 CROSS REFERENCE TABLE -- USER SYMBOLS

RESTART	007534	749	753	761#												
RESOS =	104407	229#	1078													
RETURN	001214	156#	492*	720*	724	761*	799*	803	1092*	1095	1127	1345*	1355*	1357		
ROMCLK=	104414	239#	1134	1137	1174	1179	1680	1703	1727	1751	1784	1786	1797	1799		
		1826	1828	1839	1841	1871	1873	1892	1894	1928	1930	1946	1948	1975		
		1997	2021	2030	2055	2057	2061	2088	2090	2095	2105	2132	2134	2139		
		2143	2153	2163	2191	2196	2217	2242	2247	2268	2293	2298	2319	2344		
		2349	2370	2399	2403	2408	2431	2456	2484	2512	2540	2570	2580	2582		
		2618	2627	2655	2664	2692	2701	2729	2738	2769	2807	2813	2850	2858		
		2866	2895	2898	2907	2910	2942	2946	2954	3287	3291	3295	3325	3353		
		3357	3361	3391	3525	3538	3543	3552	3721	3735	3740	3749	3764	3778		
		3783	3792	3988	4002	4007	4016	4028	4042	4056	4061	4070	4110	4117		
		4127	4135	4161	4170	4232	4248	4253	4262	4317	4332	4337	4346	4364		
		4377	4392	4396	4399	4405	4408	4419	4437	4441	4444	4450	4463	4466		
		4483	4487	4490	4496	4516	4518	4535	4587	4591	4594	4600	4610	4620		
		4633	4637	4640	4653	4656	4667	4669	4684	4698	4700					
RUN	001316	193#	485*	1286*	1287*	1294										
SAVACT	001312	191#	671	1588*												
SAVNUM	001314	192#	479*	748*	751*	1581*										
SAVPC	001276	185#	622*	642	929*	1099										
SAVR0	001260	178#	938*	943	4785											
SAVR1	001262	179#	628*	937*	944											
SAVR2	001264	180#	936*	945	4785											
SAVR3	001266	181#	935*	946	4785											
SAVR4	001270	182#	934*	947	4785											
SAVR5	001272	183#	933*	948	4785											
SAVSP	001274	184#														
SAVOS =	104406	227#	1038													
SCHAR	027152	2760*	2801*	4477	4495	4500	4504*	4505	4511*	4521#						
SCOPE =	104400	215#	1688	1711	1735	1766	1808	1850	1908	1960	1982	2004	2037	2070		
		2114	2171	2222	2273	2324	2375	2436	2464	2492	2520	2548	2597	2634		
		2671	2708	2745	2784	2828	2873	2921	2964	3039	3114	3189	3264	3330		
		3396	3497	3554	3800	4078	4143	4178	4273	4354						
SCOP1 =	104401	217#	1794	1807	1836	1849	1882	1903	1937	1955	3009	3038	3			

SWMES	007205	1187*	1227											
SWMES1	007215	1187*	1230											
SWR	001202	143*	497*	499	503*	513	671	676	781	788	812	827	1027	1032
		1081	1088	1090	1152	1212	1250*							
SWREG	000176	129*	503	1212	1270									
SW00	= 000001	45*	517	1377	1601									
SW01	= 000002	44*	721	1323	1347									
SW02	= 000004	43*												
SW03	= 000010	42*	666											
SW04	= 000020	41*												
SW05	= 000040	40*												
SW06	= 000100	39*	1152											
SW07	= 000200	38*												
SW08	= 000400	37*	1088											
SW09	= 001000	36*	812											
SW10	= 002000	35*	1090											
SW11	= 004000	34*	788											
SW12	= 010000	33*	827	1027										
SW13	= 020000	32*	1032											
SW14	= 040000	31*												
SW15	= 100000	30*												
SYNC	026366	2453	2481	2509	2537	4384*								
SYNLD	027464	3285	3351	3420	3516	3581	3828	4098	4163	4200	4201	4294	4295	4627*
TEMP	001416	271*	971	1118*	1119*	1160*	1165*	1171*	1183*	2019*	2025*	2028*	2034*	
TEMP1	001246	173*	537*	1189	1613*	1614*	4389*	4401*	4435*	4446*	4481*	4492*	4533*	4539*
		4551*	4565*	4585*	4596*	4631*	4642*	4681*	4686*					
TEMP2	001250	174*	538*	1191	3751*	3752*	3753	3794*	3795*	3796	4018*	4019*	4020	4072*
		4073*	4074	4264*	4265*	4266	4348*	4349*	4350	4433*	4449	4459*	4462	4476*
		4480*	4507*	4514*	4552*	4557*	4583*	4599						
TEMP3	001252	175*	540*	566*	618	627*	1193	1386	1389	1501*	3540*	3541*	3737*	3738*
		3753*	3754	3780*	3781*	3796*	3797	4004*	4005*	4020*	4021	4058*	4059*	4074*
		4075	4250*	4251*	4266*	4267	4334*	4335*	4350*	4351	4711*	4721*	4734*	4747*
		4785												
TEMP4	001254	176*	541*	1195	1399	1402	1408	1411	1487	1490	1496	1499		
TEMP5	001256	177*	542*	1197	1380*	1393*	1599	4417*	4423*					
TIMER	= 104416	243*	2059	2569	2617	2654	2691	2728	2897	2909	2944			
TKCSR	001204	148*	848	1214	1259	1655	4759							
TKDBR	001206	149*	850	856	1217	1219	1261	1657						
TLAST	= 026000	1350	4355*											
TPCSR	001210	150*	832	854	1029	1262	1658							
TPDBR	001212	151*	834*	856*	1031*	1264*	1660*							

[illegible]

XLOC	003022	593*	611*	614	645	658*								
XPASS	003562	742	769*											
XPOLY	027316	2984*	3013*	3059*	3088*	3134*	3163*	3209*	3238*	3284*	3350*	3435*	3600*	3651*
		3848*	3918*	4202*	4296*	4561	4563	4571*						
XSTAT0	007454	544	1187*											
XTSTN	005330	1060	1100*											
XVEC	003554	740	766*											
ZERO	001300	186*	4785											
\$COD = ***** U		1												
\$CRAP = 177777		1*	1665*	1668	1671*	1689*	1692	1695*	1712*	1715	1718*	1736*	1739	1742*
		1767*	1770	1773*	1809*	1812	1815*	1851*	1854	1857*	1909*	1912	1915*	1961*
		1964	1967*	1983*	1986	1989*	2005*	2008	2011*	2038*	2041	2045*	2071*	2074
		2078*	2115*	2118	2122*	2172*	2175	2180*	2223*	2226	2231*	2274*	2277	2282*
		2325*	2328	2333*	2376*	2379	2385*	2437*	2440	2443*	2465*	2468	2471*	2493*
		2496	2499*	2521*	2524	2527*	2549*	2552	2556*	2598*	2601	2604*	2635*	2638
		2641*	2672*	2675	2678*	2709*	2712	2715*	2746*	2749	2752*	2785*	2788	2793*
		2829*	2832	2837*	2874*	2877	2881*	2922*	2925	2928*	2965*	2968	2972*	3040*
		3043	3047*	3115*	3118	3122*	3190*	3193	3197*	3265*	3268	3271*	3331*	3334
		3337*	3397*	3400	3405*	3498*	3501	3505*	3555*	3558	3566*	3801*	3804	3813*
		4079*	4082	4087*	4144*	4147	4150*	4179*	4182	4189*	4274*	4277	4283*	
		123	509	755*	1079									
\$ENDAD = 003522		1*	1665	1671	1673	1678*	1689	1695	1697	1702*	1712	1718	1720	1725
\$N = 000057		1726*	1736	1742	1744	1749	1750*	1767	1773	1775	1781	1782*	1809	1815
		1817	1823	1824*	1851	1857	1859	1865	1866*	1909	1915	1917	1923	1924*
		1961	1967	1969	1974	1975*	1983	1989	1991	1996	1997*	2005	2011	2013
		2018	2019*	2038	2045	2047	2052	2053*	2071	2078	2080	2085	2086*	2115
		2122	2124	2129	2130*	2172	2180	2182	2187	2188*	2223	2231	2233	2238
		2239*	2274	2282	2284	2289	2290*	2325	2333	2335	2340	2341*	2376	2385
		2387	2392	2393*	2437	2443	2445	2450	2451*	2465	2471	2473	2478	2479*
		2493	2499	2501	2506	2507*	2521	2527	2529	2534	2535*	2549	2556	2558
		2563	2564*	2598	2604	2606	2611	2612*	2635	2641	2643	2648	2649*	2672
		2678	2680	2685	2686*	2709	2715	2717	2722	2723*	2746	2752	2754	2759
		2760*	2785	2793	2795	2800	2801*	2829	2837	2839	2844	2845*	2874	2881
		2883	2888	2889*	2922	2928	2930	2935	2936*	2965	2972	2974	2980	2981*
		3040	3047	3049	3055	3056*	3115	3122	3124	3130	3131*	3190	3197	3199
		3205	3206*	3265	3271	3273	3278	3279*	3331	3337	3339	3344	3345*	3397
		3405	3407	3412	3413*	3498	3505	3507	3512	3513*	3555	3566	3568	3573
		3574*	3801	3813	3815	3820	3821*	4079	4087	4089	4094	4095*	4144	4150
		4152	4157	4158*	4179	4189	4191	4196	4197*	4274	4283	4285	4290	4291*
		4355*												
\$S = 000061		1*	1676	1678*	1700	1702*	1723	1726*	1747	1750*	1778	1782*	1820	1824*
		1862	1866*	1920	1924*	1972	1975*	1994	1997*	2016	2019*	2050	2053*	2083
		2086*	2127	2130*	2185	2188*	2236	2239*	2287	2290*	2338	2341*	2390	2393*
		2448	2451*	2476	2479*	2504	2507*	2532	2535*	2561	2564*	2609	2612*	2646
		2649*	2683	2686*	2720	2723*	2757	2760*	2798	2801*	2842	2845*	2886	2889*
		2933	2936*	2977	2981*	3052	3056*	3127	3131*	3202	3206*	3276	3279*	3342
		3345*	3410	3413*	3510	3513*	3571	3574*	3818	3821*	4092	4095*	4155	4158*
		4194	4197*	4288	4291*									
\$Y = 000017		1*	207*	215	217*	219*	221*	223*	225*	227*	229*	231*	233*	235*
		237*	239*	241*	243*	245*								
.	= 032202	108*	109	112*	119*	124*	127*	131*	135*	137*	199*	190*	191*	192*
		272*	277*	279*	280*	281*	282*	284*	285*	286*	287*	289*	290*	291*
		292*	294*	295*	296*	297*	299*	300*	301*	302*	304*	305*	306*	307*
		309*	310*	311*	312*	314*	315*	316*	317*	319*	320*	321*	322*	324*
		325*	326*	327*	329*	330*	331*	332*	334*	335*	336*	337*	339*	340*
		341*	342*	344*	345*	346*	347*	349*	350*	351*	352*	354*	355*	356*

020ME.F11 12-MAY-77 14:18

CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

CMEND	18	725													
CMERN	18														
18	758	1687	1710	1734	1765	1793	1806	1835	1848	1881	1902	1936	1954	1981	2000
2027	2036	2068	2102	2112	2150	2160	2163	2163	2207	2211	2221	2258	2262	2272	2300
2313	2323	2360	2364	2374	2419	2423	2463	2463	2491	2519	2547	2577	2589	2596	2620
2633	2662	2670	2699	2707	2736	2744	2775	2775	2819	2856	2864	2872	2905	2920	2960
2999	3003	3028	3032	3074	3078	3103	3107	3107	3149	3153	3178	3182	3224	3228	3253
3257	3308	3312	3374	3378	3447	3451	3468	3472	3484	3493	3531	3550	3550	3612	3616
3633	3637	3663	3667	3684	3688	3700	3709	3727	3747	3756	3770	3790	3799	3799	3860
3864	3881	3885	3901	3905	3930	3934	3951	3955	3967	3976	3994	4014	4023	4023	4034
4048	4068	4077	4114	4123	4133	4141	4177	4238	4260	4269	4323	4344	4353	4353	4425
4541	4746														
\$ABORT	18														
\$ALTC	18	547													
\$BCC	18	2965	3040	3115	3190										
\$BINCR	18	3265	3331												
\$BINWI	18	2376													
\$BLFFE	18	1199													
\$CDATA	18	4179	4274												
\$CLOCK	18	2005													
\$COMP	18	2572	2584	2591	2620	2629	2657	2666	2694	2703	2731	2740	2852	2860	2868
2900	2912	3545	3742	3785	4009	4063	4255	4339							
\$CRC	18	2982	3011	3057	3086	3132	3161	3207	3236						
\$CRCSH	18	3279	3345												
\$CYCLE	18	1271													
\$EMPTY	18	4079													
\$EOP	18	725													
\$FINI	18	4355													
\$FLAG	18														
\$FLOAT	18	1867	1887	1925	1942										
\$GETPA	18														
\$HALF	18	4144													
\$HEADE	18														
\$INACT	18	2437	2465	2493	2521										
\$INIT	18														
\$LINE1	18	1851	1909												
\$LU1	18	1665	1689	1712	1736										
\$LU12	18	1767													
\$LU17	18	1809													
\$MARHI	18														
\$MARK	18	2829													
\$MATCH	18	3534	3730	3773	3997	4051	4243	4327							
\$MOCK	18														
\$MODEM	18	2874													
\$MSG	18	1187													
\$MULT	18														
\$PATTE	18	2746	2785												
\$PFAIL	18	1103													
\$GOOI	18	4606	4616												
\$QUEST	18	1381	1394	1403	1482	1491									
\$RAMCL	18	1131													
\$RCLK	18	1134	1137	1174	1179	1680	1703	1727	1751	1784	1786	1797	1799	1826	1830
1839	1841	1871	1873	1892	1894	1928	1930	1946	1948	1975	1997	2021	2026	2030	2050
2057	2061	2088	2090	2095	2105	2132	2134	2139	2143	2153	2163	2191	2196	2200	2205
2242	2247	2268	2293	2298	2319	2344	2349	2370	2399	2403	2408	2431	2456	2456	2464
2512	2540	2570	2580	2582	2618	2627	2655	2664	2692	2701	2709	2738	2769	2769	2774

DZDME.P11 12-MAY-77 14:18

CROSS REFERENCE TABLE -- MACRO NAMES

	2813	2850	2858	2866	2895	2898	2907	2910	2942	2946	2953	3287	3291	3295	3325
	3353	3357	3361	3391	3525	3538	3543	3552	3721	3735	3740	3749	3764	3778	3783
	3792	3988	4002	4007	4016	4028	4042	4056	4061	4070	4110	4117	4127	4135	4161
	4170	4232	4248	4253	4262	4317	4332	4337	4346	4364	377	4392	4396	4399	4405
	4408	4419	4437	4441	4444	4450	4463	4466	4483	4487	4490	4496	4516	4518	4535
	4587	4591	4594	4600	4610	4620	4633	4637	4640	4653	4656	4667	4669	4684	4698
	4700														
\$RCRC	18	3498													
\$REC	18	2549	2598	2635	2672	2709									
\$SCOPE	18	775													
\$S:MBC	18	4545													
\$SINAC	18														
\$SOFTC	18	1207													
\$STUFF	18														
\$SWPAC	18	1961	1983												
\$TCHAR	18	3416	3513	3577	3824										
\$TCRC	18	3397	3555	3801											
\$TRANW	18	3435	3600	3651	3848	3918									
\$TRANI	18	2038	2071	2115											
\$TRPDE	18	215	217	219	221	223	225	227	229	231	233	235	237	239	241
	243														
\$TSTN	18	1673	1697	1720	1744	1775	1817	1859	1917	1969	1991	2013	2047	2080	2124
	2182	2233	2284	2335	2387	2445	2473	2501	2529	2558	2606	2643	2680	2717	2754
	2795	2839	2883	2930	2974	3049	3124	3199	3273	3339	3407	3507	3568	3815	4099
	4152	4191	4285												
\$VARIA	18	134													
\$WINDO	18	2172	2223	2274	2325										
\$XZ	18	1665	1671	1689	1695	1712	1718	1736	1742	1767	1773	1809	1815	1851	1857
	1909	1917	1961	1967	1983	1989	2005	2011	2038	2045	2071	2078	2115	2122	2173
	2180	2223	2231	2274	2282	2325	2333	2376	2385	2437	2443	2465	2471	2493	2499
	2521	2527	2549	2556	2598	2604	2635	2641	2672	2678	2709	2715	2746	2752	2795
	2793	2829	2837	2874	2881	2922	2928	2965	2972	3040	3047	3115	3122	3190	3197
	3265	3271	3331	3337	3397	3405	3498	3505	3555	3566	3801	3813	4079	4087	4144
	4150	4179	4189	4274	4283										
\$ZEROS	18														

. ABS. 032202 000

ERRORS DETECTED: 0

DZDME,DZDME/SOL/CRF+IPLUTL,DZDME
 RUN-TIME: 16 22 1 SECONDS
 RUN-TIME RATIO: 779/39=19.5
 CORE USED: 31K (61 PAGES)