

DMC11

BITSTUFF MODE LINE TESTS
MD-11-DZDMF-B

EP-DZDMF-B-DL-A
COPYRIGHT © 76-77
FICHE 1 OF 1

AUG 1977
digital
MADE IN USA

B01

ECF10ZDMFBSEQ

PDP10 MODE: 0001

00010000

770712

PDP10 411

83H0R10ZDMFBSEQ

00010000

770712

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZDMF-B-C
PRODUCT NAME: BITSTUFF 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

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.

DZDMF 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 BITSTUFF mode. The modem signals are also checked. DZDMF requires a DMC Micro-Processor (M8200 or M3204) 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
 ASA 33 (or equivalent)
 DMC11-AR with DMC11-DA or DMC11-FA or
 DMC11-AL with DMC11-MA or DMC11-MC

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

D1
CSR ADDRESS?160010
VECTOR ADDRESS?310
BR PRIORITY LEVEL? (4,5,6,7)?5
DOES MICRO-PROCESSOR HAVE CRAM? (Y OR N,N
WHICH LINE UNIT? IF NONE TYPE "N", IF M8201 TYPE "1", IF
M8202 TYPE "2"?1
IS THE LOOP BACK CONNECTOR ON?Y
SWITCH PAC#1 (DDCMP LINE#)?377
SWITCH PAC#2 (BM873 BOOT ADD)?377

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

4.1 CONTROL SWITCH SETTINGS

SW 15 Set: Halt on error
SW 14 Set: Loop on current test
SW 13 Set: Inhibit error print out
SW 12 Set: Inhibit type out/abell on error.
SW 11 Set: Inhibit iterations. (quick pass)
SW 10 Set: Escape to next test on error
SW 09 Set: Loop with current data
SW 08 Set: Catch error and loop on it
SW 07 Set: Use previous status table.
SW 06 Set: Halt in ROMCLK routine before clocking
micro-processor
SW 05 Set: Reserved
SW 04 Set: Reserved
SW 03 Set: Reselect DMC11's desired active
SW 02 Set: Lock on selected test
SW 01 Set: Restart program at selected test
SW 00 Set: Build new status table from questions. (If SW07=0
and SW00=0 a new status table is built by
auto-sizing)

Switch 06 and 08-15 are dynamic and can be changed as needed while the diagnostic is running. Switches 00-03 and switch 07 are static, and are used only on starting or restarting the diagnostic.

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 DMC11s; bits 00,01,02,03 will be set in loc 'DMACTV' from the switch register. Using this switch(SW00) alters that location; therefore if four DMC11s are in the system ***DO NOT*** set 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 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(MB200)- Jumper W1 must be in, and switch 7 of E76 must be in the OFF position.

KMC(MB204)- Jumper W1 must be in.

LINE UNIT(MB201)- 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 (MB202)- 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. EXAMP E: (RUN) 1302/0000000001000000 Means that DMC11 no.06 is the DMC11 now running.

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

These locations contain the information needed to test up to 16 (decimal) DMC11s 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) 1276/00000000000011111 means that DMC11 no. 00,01,02,03,04 will be tested. EXAMPLE: (DMACTV) 1276/00000000000010001 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 a 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 DMC11-AR. Further tests are performed at this point to determine which line unit, if any, is installed, if a loop-back connector is installed and various switch settings on the line unit. THIS IS WHY THE STATUS TABLE MUST BE VERIFIED BY THE USER AND IF ANY OF THE INFORMATION DOES NOT AGREE WITH THE HARDWARE THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS MUST BE ANSWERED. All DMC11's in the system will be found by the auto-sizer. If it does not find a DMC11 the diagnostic must be restarted and the questions answered.

8.5.2 FINDING THE VECTOR AND BR LEVEL

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

DZDMF LST

B02

DECDOC VER 00.04 11-JUL-77 12:04 PAGE 01 PAGE: 0014

DOCUMENT

DZDMF LST

COPYRIGHT 1977
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

6 MAINDEC-11-DZDMF-8 DMC11 BITSTUFF 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

1911 ***** 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

2077 ***** TEST 15 *****
BITSTUFF 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

2125 ***** 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

2186 ***** TEST 17 *****
BITSTUFF TRANSMITTER TEST
SINGLE CLOCK THE CHARACTER 0
CHECK FLAG AND DATA IN THE BIT WINDOW
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

2260 ***** TEST 20 *****
BITSTUFF TRANSMITTER TEST
SINGLE CLOCK THE CHARACTER 125
CHECK FLAG AND DATA IN THE BIT WINDOW
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

2334 ***** TEST 21 *****

2335 BITSTUFF TRANSMITTER TEST
SINGLE CLOCK THE CHARACTER 252
CHECK FLAG AND DATA IN THE BIT WINDOW
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

2408 ***** TEST 22 *****
BIT STUFF TEST
THIS TEST CHECKS ZERO BIT STUFFING OF
THE TRANSMITTER IN THE BIT WINDOW

2485 ***** TEST 23 *****
BITSTUFF TRANSMITTER TEST
SINGLE CLOCK THE CHARACTER 377
CHECK FLAG AND DATA IN THE BIT WINDOW
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

2565 ***** TEST 24 *****
BITSTUFF 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

2654 ***** TEST 25 *****
MULTIPLE FLAG AND TRANSMITTER ABORT TEST
LOAD SILO WITH 5 FLAGS AND A CHAR (000)
VERIFY IN THE BIT WINDOW THAT THE FLAGS
AND DATA ARE CORRECT AND FOLLOWED BY AN ABORT
SEQUENCE (8 CONTIGUOUS 1'S)

2729 ***** TEST 26 *****
LEADING ZEROS TEST
VERIFY THAT THE SETTING OF SOM AND EOM TOGETHER
AND THEN SOM ALONE WILL GENERATE 16 LEADING ZEROS
AND A FLAG, THE CHECK IS MADE USING THE BIT WINDOW

2789 ***** TEST 27 *****
BITSTUFF STRIP FLAG TEST
SET LU LOOP, SINGLE STEP 5 FLAGS,
VERIFY THAT IN ACTIVE DOES NOT SET

2821 ***** TEST 30 *****
BITSTUFF IN ACTIVE TEST
SET LU LOOP, SINGLE STEP 5 FLAGS AND A NON-FLAG (301)
VERIFY THAT IN ACTIVE IS SET

2853 ***** TEST 31 *****
BITSTUFF IN ACTIVE TEST
SET LINE UNIT LOOP, SINGLE STEP ONE FLAG AND A CHAR (301)
VERIFY THAT IN ACTIVE IS SET

2893 ***** TEST 32 *****
BITSTUFF IN ACTIVE TEST

2895 SET LU LOOP, SINGLE STEP 2 FLAGS AND A NON-FLAG (301)
VERIFY THAT IN ACTIVE IS SET

2925 ***** TEST 33 *****
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

2983 ***** TEST 34 *****
BITSTUFF BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 0
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

3029 ***** TEST 35 *****
BITSTUFF BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 125
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

3075 ***** TEST 36 *****
BITSTUFF BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 252
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

3121 ***** TEST 37 *****
BITSTUFF BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 377
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

3167 ***** TEST 40 *****
BITSTUFF DATA TEST
THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
CHECKING EACH CHARACTER AS IT IS RECEIVED

3212 ***** TEST 41 *****
BITSTUFF 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

3262 ***** TEST 42 *****
RECEIVER ABORT TEST
SINGLE CLOCK 3 FLAGS, A 301, ANOTHER 301 AND 10 EXTRA
CLOCK TICKS, VERIFY THAT A 301 AND A BLOCK END
WERE RECEIVED INDICATING THAT THE RECEIVER RECOGNIZED
THE ABORT SEQUENCE (8 CONTIGUIOUS 1'S)

3307 ***** TEST 43 *****
CABLE TURNAROUND TEST
CLEAR LINE UNIT LOOP, SET DTR
VERIFY THAT MODEM READY IS SET
CLEAR DTR, VERIFY THAT MRDY IS CLEARED

3355 ***** TEST 44 *****
CABLE TURNAROUND TEST
CLEAR LINE UNIT LOOP, LOAD OUT DATA SILO
VERIFY THAT ALL MODEM SIGNALS ARE SET

3398 ***** TEST 45 *****

3399 TEST OF CRC OPERATION
USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
0, VERIFY THE LSB OF THE BCC ON EACH SHIFT
TEST TRANSMITTER FIRST THEN THE RECEIVER BCC

3480 ***** TEST 46 *****
TEST OF CRC OPERATION
USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
377, VERIFY THE LSB OF THE BCC ON EACH SHIFT
TEST TRANSMITTER FIRST THEN THE RECEIVER BCC

3568 ***** TEST 47 *****
TEST OF CRC OPERATION
USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
125, VERIFY THE LSB OF THE BCC ON EACH SHIFT
TEST TRANSMITTER FIRST THEN THE RECEIVER BCC

3650 ***** TEST 50 *****
TEST OF CRC OPERATION
USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
252, VERIFY THE LSB OF THE BCC ON EACH SHIFT
TEST TRANSMITTER FIRST THEN THE RECEIVER BCC

3732 ***** TEST 51 *****
TRANSMITTER CRC TEST
USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK A BINARY

3735 COUNT PATTERN, VERIFY THE LSB OF THE TRANSMITTER BCC ON EACH SHIFT

3815 ***** TEST 52 *****
RECEIVER CRC TEST
USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK A BINARY
COUNT PATTERN, VERIFY THE LSB OF THE RECEIVER BCC ON EACH SHIFT

3901 ***** TEST 53 *****
TRANSMITTER BITSTUFF CRC TEST

3903 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

4038 ***** TEST 54 *****
RECEIVER BITSTUFF 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

4100 ***** TEST 55 *****
BITSTUFF EOM FUNCTION TEST
THIS TEST LOADS OUT SILO WITH: 2 FLAGS, 4 CHAR MESSAGE, EOM
4 CHARACTER MESS, EOM. THE DATA STREAM IS CHECKED TO BE
4 CHAR, BCC, FLAG, 4 CHAR, BCC, FLAG, 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

4413 ***** TEST 56 *****
BITSTUFF EOM FUNCTION TEST
THIS TEST LOADS OUT SILO WITH: 2 FLAGS, 4 CHAR MESSAGE, EOM
SOM, 4 CHAR MESS, EOM. THE DATA STREAM IS CHECKED TO BE
4 CHAR, BCC, FLAG, 4 CHAR, BCC, FLAG, 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

4746 ***** TEST 57 *****
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 CHARACTERS AND A BLOCK END WERE RECEIVED, AND IN ACTIVE IS CLEAR

4810 ***** TEST 60 *****
BITSTUFF CABLE DATA TEST
THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
2 FLAGS, 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

4913 ***** TEST 61 *****
BITSTUFF CABLE DATA TEST
THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
2 FLAGS, 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

020MF LST

102

DECD0C VER 00.04 11-JUL-77 12:04 PAGE 08 PAGE: 0021

LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST

INTRODUCTION TO DMC11 DIAGNOSTIC

```

; *MAINDEC-11-DZDMF-B DMC11 BITSTUFF 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-DZDMF-B DMC11 BITSTUFF 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, INHIBIT ERROR TYPEOUT
SW11=4000         ;=1, DELETE TYPEOUT/BELL ON ERROR.
SW10=2000         ;=1, INHIBIT ITERATIONS
SW09=1000         ;=1, ESCAPE TO NEXT TEST ON ERROR
SW08=400          ;=1, LOOP WITH CURRENT DATA
SW07=200          ;=1, LOOP ON ERROR
SW06=100          ;=1, USE CURRENT DMC11 PARAMETERS, =0, AUTOSIZE DMC11
SW05=40           ;=1, HALT BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION
SW04=20
SW03=10           ; RESELECT DMC11'S TO BE TESTED (ACTIVE)
SW02=4            ; LOCK ON TEST SELECT
SW01=2            ; RESTART PROGRAM AT SELECTED TEST
SW00=1            ; INPUT DMC11 PARAMETERS

```

;REGISTER DEFINITIONS

```

;-----
R0=%0      ;GENERAL REGISTER
R1=%1      ;GENERAL REGISTER
R2=%2      ;GENERAL REGISTER
R3=%3      ;GENERAL REGISTER
R4=%4      ;GENERAL REGISTER
R5=%5      ;GENERAL REGISTER
SP=%6      ;PROCESSOR STACK POINTER
PC=%7      ;PROGRAM COUNTER

```

;LOCATION EQUIVALENCIES

```

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

```

;INSTRUCTION DEFINITIONS

```

;-----
PUSH1SP=5746 ;DECREMENT PROCESSOR STACK 1 WORD
POP1SP=5726  ;INCREMENT PROCESSOR STACK 1 WORD
PUSHRO=10046 ;SAVE R0 ON STACK
POPRO=12600  ;RESTORE R0 FROM STACK
PUSH2SP=24646 ;DECREMENT STACK TWICE
POP2SP=22626 ;INCREMENT STACK TWICE
.EQUIV EMT,HLT ;BASIC DEFINITION OF ERROR CALL

```

;BIT DEFINITIONS

```

;-----
BIT15=100000
BIT14=40000
BIT13=20000
BIT12=10000
BIT11=4000
BIT10=2000
BIT9=1000
BIT8=400
BIT7=200
BIT6=100
BIT5=40
BIT4=20
BIT3=10
BIT2=4
BIT1=2
BIT0=1

```

```

000000
000001
000002
000003
000004
000005
000006
000007

```

```

177776
001200

```

```

005746
005726
010046
012600
024646
022626

```

```

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

```

```

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
120      000040 000000
121      000042 000000
122      000044 000000
123      000046 003522
124      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      001025      104 041515 030461
138
139      001200
140
141
142      001200 177570
143      001202 177570

```

```

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

```

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

```

144
145      ;INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS
146      ;-----
147
148      C71204 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      STARTSW: 0      ;SWITCHES AT START OF PROGRAM
170      001240 000000      STAT: 0      ;DM STATUS WORD STORAGE
171      001242 000000      CLKX: 0
172      001244 000000      MASKX: 0
173      001246 000000      TEMP1: 0      ;TEMPORARY STORAGE
174      001250 000000      TEMP2: 0      ;TEMPORARY STORAGE
175      001252 000000      TEMP3: 0      ;TEMPORARY STORAGE
176      001254 000000      TEMP4: 0      ;TEMPORARY STORAGE
177      001256 000000      TEMP5: 0      ;TEMPORARY STORAGE
178      001260 000000      SAVR0: 0      ;R0 STORAGE
179      001262 000000      SAVR1: 0      ;R1 STORAGE
180      001264 000000      SAVR2: 0      ;R2 STORAGE
181      001266 000000      SAVR3: 0      ;R3 STORAGE
182      001270 000000      SAVR4: 0      ;R4 STORAGE
183      001272 000000      SAVR5: 0      ;R5 STORAGE
184      001274 000000      SAVSP: 0      ;STACK POINTER STORAGE
185      001276 000000      SAVPC: 0      ;PROGRAM COUNTER STORAGE
186      001300 000000      ZERO: 0
187      001302 000001      ONE: 1
188      001304 000000      MEMLIM: 0      ;HIGHEST LOCATION FOR NPR'S
189      001306 000001      DMACTV: .BLKW 1      ;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.
197      MILK: CNT.MAP-4      ;TABLE POINTER

```

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

```

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 104400
216 001330 003576
217 104401
218 001332 003736
219 104402
220 001334 003766
221 104403
222 001336 004050
223 104404
224 001340 004154
225 104405
226 001342 004174
227 104406
228 001344 004374
229 104407
230 001346 004434
231 104410
232 001350 004466
233 104411
234 001352 004472
235 104412
236 001354 005466
237 104413
238 001356 005436
239 104414
240 001360 005504
241 104415
242 001362 005552
243 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 INPLT ERROR HANDLER
.INSTER
PARAM=TRAP+5 ;CALL TO NUMERICAL DATA INPLT 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 ISUE 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      DMRVLV: 0      ; POINTER TO DMC11 RECEIVER INTERRUPT SERVICE PS
260      001400 000000      DMTVEC: 0      ; POINTER TO DMC11 TRANSMITTER INTERRUPT VECTOR
261      001402 000000      DMTVLV: 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      DMFO4: 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      001500      . = 1500
278      001500      DM.MAP:
279      001500 000001      DMC00: .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      DMC01: .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      DMC02: .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      DMC03: .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      DMC04: .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

```

020MF MAC11 30(1046) 11-JUL-77 11:59 PAGE 8
 020ME.P11 12-MAY-77 14:18

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

PAGE: 0028

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		

DDMMF MACY11 3011046 11-JUL-77 11:59 PAGE 9
 DDMMF.P11 12-MAY-77 14:18

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

PAGE: 0029

```

360
361                                     ;DMC11 PASS COUNT AND ERROR COUNT TABLE
362                                     ;-----
363
364 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 002030 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	R	E	G	I	S	T	E	R	
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	STAT1
I	*	I	*	I	*	I	*	I	*	I	*	I	*	I	*	
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	STAT2
I	*	I	B	I	M	I	I	A	D	I	D	*	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	

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 DZDM! ONLY])
 KMC MUST HAVE MICRO-CODE WRITTEN FROM RUNNING
 DZDMG TEST 2 FIRST
 BIT1=1 DMC11-AL LOCAL HIGH SPEED MICRO-CODE
 BIT1=0 DMC11-AR REMOTE LOW SPEED MICRO-CODE

```

468
469
470
471
472
473
474
475
476 002002 012737 000340 177776 .START: MOV #340,PS ;LOCK OUT INTERRUPTS
477 002010 012706 001200 MOV #STACK,SP ;SET UP STACK
478 002014 012737 005336 000024 MOV #.PFAIL,2#24 ;SET UP POWER FAIL VECTOR
479 002022 013737 001310 001314 MOV DMNUM,SAVNUM ;SAVE NUMBER OF DEVICES IN SYSTEM.
480 002030 005037 010016 CLR SWFLG ;CLEAR SOFT TYPEOUT FLAG
481 002034 105037 001325 CLRB ERRFLG ;CLEAR ERROR FLAG
482 002040 105037 001327 CLRB QV.FLG ;ZERO QUICK VERIFY FLAG
483 002044 012737 001470 001320 MOV #DM.MAP-10,CREAM ;GET MAP POINTER.
484 002052 012737 001676 001322 MOV #CNT.MAP-4,MILK ;GET PASS COUNT MAP POINTER
485 002060 012737 100000 001316 MOV #BIT15,RUN ;POINT POINTER TO FIRST DEVICE.
486 002066 012700 001702 MOV #CNT.MAP,RO ;PASS COUNT POINTER TO RO
487 002072 005020 23$: CLR (RO)+ ;CLEAR TABLE
488 002074 022700 002002 CMP #CNT.MAP+100,RO ;DONE YET?
489 002100 001374 BNE 23$ ;KEEP GOING
490 002102 005037 001234 CLR LSTERR ;CLEAR LAST ERROR POINTER
491 002106 012737 000001 001226 MOV #1,TSTNO ;SET UP FOR TEST 1
492 002114 012737 002002 001214 MOV #.START,RETURN ;SET UP FOR POWER FAIL BEFORE
493 ;TESTING STARTS
494 002122 013746 000006 MOV 2#6,-(SP) ;SAVE CURRENT VECTORS
495 002126 013746 000004 MOV 2#4,-(SP)
496 002132 012737 002166 000004 MOV #6$ ,2#4 ;SET UP FOR TIMEOUT
497 002140 012737 177570 001202 MOV #177570,SWR ;SET SWR TO HARD SWR ADDRESS
498 002146 012737 177570 001200 MOV #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
499 002154 022777 177777 177020 CMP #-1,2SWR ;REFERENCE HARDWARE SWITCH REGISTER
500 002162 001402 BEQ 6$+2 ;IF = -1 USE SOFT SWR ANYWAY
501 002164 000407 BR 7$ ;IF IT EXISTS AND NOT = -1 USE HARD SWR
502 002166 022626 6$: CMP (SP)+,(SP)+ ;ADJUST STACK
503 002170 012737 000176 001202 MOV #SWREG,SWR ;POINTER TO SOFT SWR
504 002176 012737 000174 001200 MOV #DISPREG,DISPLAY ;POINTER TO SOFT DISPLAY REG
505 002204 012637 000004 7$: MOV (SP)+,2#4 ;RESTORE VECTORS
506 002210 012637 000006 MOV (SP)+,2#6
507 002214 105737 001324 TSTB INIFLG ;HAS INITIALIZATION BEEN PERFORMED
508 002220 001006 BNE 20$ ;BR IF YES
509 002222 022737 003522 000042 CMP #SENDAD,2#42 ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
510 002230 001402 BEQ 20$
511 002232 104402 001000 TYPE #MTITLE ;TYPE TITLE MESSAGE
512 002236 004737 007606 20$: JSR PC,CKSWR ;CHECK FOR SOFT SWR
513 002242 017737 176734 001236 MOV 2SWR,STATSW ;STORE STARTING SWITCHES
514 002250 005737 000042 TST 2#42 ;IS IT RUNNING IN AUTO MODE?
515 002254 001402 BEQ .+6 ;BR IF NO
516 002256 005037 001236 CLRB STATSW ;IF YES, CLEAR SWITCHES
517 002262 032737 000001 001236 BIT #SW00,STATSW ;IF SW00=1, QUESTIONS ARE ASKED.
518 002270 001012 BNE 17$ ;BR IF SW00=1
519 002272 105737 001236 TSTB STATSW ;BIT7=1?
520 002276 100007 BPL 17$ ;BR IF SW07=0
521 002300 005737 001306 TST DMACTV ;ARE ANY DEVICES SELECTED?
522 002304 001006 BNE 16$ ;BR IF YES
523 002306 104402 007154 TYPE #NOACT ;NO DEVICES SELECTED.

```

G03

02DMF MACY11 30(1046) 11-JUL-77 11:59 PAGE 12
 02DME.P11 12-MAY-77 14:18

PAGE: 0032

PROGRAM INITIALIZATION AND START UP.

```

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 21$            ;BR IF YES
529 002330 105737 001236      TSTB STATSW ;IF USING SAME PARAMETERS DONT TYPE MAP
530 002334 100431      BMI 1$             ;
531 002336 032737 000006 001236      BIT #BIT1!BIT2,STATSW ;IS TEST NO. OR LOCK SELECTED
532 002344 001403      BEQ 24$            ;IF NO THEN TYPE STATUS
533 002346 000424      BR 1$              ;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$: MOV R4,TEMP1 ;SET ADDRESS
538 002370 012437 001250      MOV (R4)+,TEMP2 ;SET CSR
539 002374 001411      BEQ 1$             ;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 5$              ;
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 2#4,-(SP) ;SAVE LOC 4
564 002430 013746 000006      MOV 2#6,-(SP) ;SAVE LOC 6
565 002434 005037 000006      CLR 2#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 #MODEV,2#4 ;SET UP FOR TIMEOUT
578 002510 005705      TST R5 ;DMC OR KMC?
579 002512 001003      BNE 3$ ;BR IF KMC

```

H03

020MF MACY11 30(1046) 11-JUL-77 11:59 PAGE 13
 020ME.P11 12-MAY-77 14:18 PROGRAM INITIALIZATION AND START UP.

PAGE: 0033

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	002556	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	103000 000302	1\$:	BIT	#BIT15,2(R0)	:CHECK FOR KMC CSR
600	002612	001407			BEG	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			BEG	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			BEG	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			BEG	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			CONVRT		:TYPE OUT ERROR PC
624	002722	002770			ERRPC		:
625	002724	104402			TYPE		:TYPE REST OF HEADER
626	002726	007277			CONERR		:
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			CONVRT		:
630	002744	002776			CONTAB		: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 OUT
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	:DG
651	003013	007			.BYTE	7	:DJ
652	003014	007			.BYTE	7	:DJP
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	15	: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,RO	:RESET RO TO START OF TABLE
663	003040	000602			BR	AUSTR	:GO DO KMC'S
664	003042	012637	000006	15:	MOV	(SP)+,2#6	:RESTORE LOC 6
665	003046	012637	000004		MOV	(SP)+,2#4	:RESTORE LOC 4
666	003052	032737	000010	001236	BIT	#SW03,STR*SW	:SELECT SPECIFIC DEVICES??
667	003060	001422			BEQ	35	:BR IF NO.
668	003062	104402	006144		TYPE	MNEW	:TYPE THE MESSAGE.
669	003066	005000			CLR	RO	:ZERO DATA LIGHTS
670	003070	000000			HALT		:WAIT FOR USER TO TELL WHAT DEVICES TO RUN
671	003072	027737	176104	001312	CMP	2SWR,SAVACT	:IS THE NUMBER VALID?
672	003100	101404			BLOS	25	:BR IF NUMBER IS OK.
673	003102	104402	006005		TYPE	MERR3	: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	2SWR,DMACTV
677	003120	013700	001306		MOV	DMACTV,RO	:GET NEW DEVICE PATTERN
678	003124	000000			HALT		:SHOW THE USER WHAT HE SELECTED.
679	003126	012700	000300	3\$:	MOV	#300,RO	:CONTINUE DYNAMIC SWITCHES.
680	003132	012701	000302		MOV	#302,R1	:PREPARE TO CLEAR THE FLOATING
681	003136	010120		4\$:	MOV	R1,(RO)+	:VECTOR AREA. 300-776
682	003140	005021			CLR	(R1)+	:START PUTTING "PC+2 - HALT"
683	003142	022021			CMP	(RO)+,(R1)+	:IN VECTOR AREA.
684	003144	022700	001000		CMP	#1000,RO	:POP POINTERS
685	003150	001372			BNE	45	:ALL DONE??
686							:BR IF NO.
687							
688							
689							
690	003152	012706	001200		.BEGIN:	MOV	#STACK,SP
691	003156	013746	000006		MOV	2#6,-(SP)	:SET UP STACK
							:SAVE LOC 6

:TEST START AND RESTART

692	003162	013746	000004		MOV	#4, -(SP)	:SAVE LOC 4
693	003166	005000			CLR	RO	:START AT 0
694	003170	012737	003234	000004	MOV	#25, #4	:SET UP FOR TIME OUT
695	003176	005037	000006		CLR	#6	:TO AUTOSIZE MEMORY
696	003202	005720			TST	(RO)+	:CHECK ADDRESS IN RO
697	003204	022700	157776		CMP	#157776, RO	:IS IT AT LEAST 28K
698	003210	001374			BNE	6\$	:BR IF NO
699	003212	162700	007776		SUB	#7776, RO	:SAVE 2K FOR MONITORS
700	003216	010037	001304		MOV	RO, MEMLIM	:STORE MEMORY LIMIT
701	003222	012637	000004		MOV	(SP)+, #4	:RESTORE LOC 4
702	003226	012637	000006		MOV	(SP)+, #6	:RESTORE LOC 6
703	003232	000413			BR	10\$	:CONTINUE
704	003234	022626			CMP	(SP)+, (SP)+	:ADJUST STACK
705	003236	162700	000004		SUB	#4, RO	:GET LAST GOOD ADDRESS
706	003242	162700	007776		SUB	#7776, RO	:SAVE 2K FOR MONITORS
707	003246	022700	030000		CMP	#30000, RO	:IS IT 8K?
708	003252	001361			BNE	7\$	:BR IF NO
709	003254	012700	037400		MOV	#37400, RO	:IF 8K DON'T SAVE 2K
710	003260	000756			BR	7\$	
711	003262	012737	000340	177776	MOV	#340, PS	:LOCK OUT INTERRUPTS
712	003270	032737	000004	001236	BIT	#BIT2, STARTSW	:CHECK FOR LOCK ON TEST
713	003276	001411			BEQ	1\$	:BR IF NO LOCK DESIRED.
714	003300	104402	006043		TYPE	, MLOCK	:TYPE LOCK SELECTED.
715	003304	012737	000240	003612	MOV	#NOP, TTST	:ADJUST SCOPE ROUTINE.
716	003312	012737	000240	003614	MOV	#NOP, TTST+2	:SET UP TO LOCK
717	003320	000406			BR	3\$	:CONTINUE ALONG.
718	003322	013737	003730	003612	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	MOV	#CYCLE, RETURN	:START AT "CYCLE" FIND WHICH DEVICE TO TEST
721	003344	032737	000002	001236	BIT	#SW01, STARTSW	:IS TEST NO. SELECTED?
722	003352	001002			BNE	5\$	:BR IF YES
723	003354	104402	005755		TYPE	MR	:TYPE R
724	003360	000177	175630		JMP	\$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 001234 CLR LSTERR ;CLEAR LAST ERROR PC
733 003372 105037 001325 CLRB ERRFLG ;CLEAR ERROR FLAG
734 003376 005237 001230 INC PASCNT ;UPDATE PASS COUNT
735 003402 013777 001230 175570 MOV PASCNT, @DISPLAY ;DISPLAY PASS COUNT
736 003410 104402 005733 TYPE ,MEPASS ;TYPE END PASS
737 003414 104402 006072 TYPE ,MCSR X ;TYPE CSR
738 003420 104411 003546 CNVRT ,XCSR ;SHOW IT
739 003424 104402 006100 TYPE ,MVECX ;TYPE VECTOR
740 003430 104411 003554 CNVRT ,XVEC ;SHOW IT
741 003434 104402 006106 TYPE ,MPASSX ;TYPE PASSES
742 003440 104411 003562 CNVRT ,XPASS ;SHOW IT
743 003444 104402 006117 TYPE ,MERRX ;TYPE ERRORS
744 003450 104411 003570 CNVRT ,XERR ;SHOW IT
745 003454 013700 001360 MOV MILK, @ ;GET POINTER TO PASS COUNT
746 003460 013720 001230 MOV PASCNT, (R0)+ ;STORE PASS COUNT FOR THIS DMC11
747 003464 013720 001232 MOV ERRCNT, (R0)+ ;STORE ERROR COUNT FOR THIS DMC11
748 003470 005337 001314 DEC SAVNUM ;ARE ALL DEVICES TESTED?
749 003474 001017 BNE RESTART ;BR IF NO.
750 003476 112737 000377 001327 MOV B #377, @V.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 DCP
753 003516 001406 BEQ RESTART ;IF NOT, CONTINUE TESTING
754 003520 000005 RESET ;STOP THE SHOW--CLEAR THE WORLD
755 003522
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 RESTART: 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 DMXVEC
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 ITERATION HANDLER
777 -----
778
779 003576 004737 007606 .SCOPE: JSR PC, CKSWR ;CHECK FOR SOFT SWR
780 003602 010016 MOV RD, (SP) ;SAVE RD 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 BECOMPLETED?
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),RO        ;POP RO 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

;CHECK FOR FREEZE ON CURRENT DATA
;-----
807
808
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.

;TELETYPE OUTPUT ROUTINE
;-----
818
819
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 000002 000002      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$:  STB     (R5)          ;IS NUMBER MINUS? (MSB=1(BIT7))
830 004022 100002                      BPL     2$          ;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

```

```

337 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    24(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$:  TSTB    2TKCSR
849 004110 100375          BPL      1$
850 004112 117714 175070  MOVB    2TKDBR,(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$:  TSTB    2TPCSR
855 004134 100375          BPL      2$
856 004136 017777 175044 175046  MOV    2TKDBR,2TPDBR
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 005656  .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          MOVB   (R5)+,LOBITS
879 004224 112537 004373          MOVB   (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	IS	
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	IS: MOV	DEVAOR,R4	
913	004342	010524		MOV	R5,(R4)+	
914	004344	062705	000002	ADD	#2,R5	
915	004350	105337	004373	DECB	ADRCNT	
916	004354	001372		BNE	IS	
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		DEVAOR: 0		
923	004372	000000		LOBITS: 0		
924		004373		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

B04

DDMMF MACY11 30(1046) 11-JUL-77 11:59 PAGE 20
DDMMF.P11 12-MAY-77 14:18

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

PAGE: 0040

```

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 15: MOVB (R1)+,CHRCNT
964 004526 112137 004713 MOVB (R1)+,SPACNT
965 004532 013137 004714 MOV @2(R1)+,BINWRD
966 004536 122737 000003 004712 CMPB #3,CHRCNT
967 004544 001003 BNE 25
968 004546 042737 177400 004714 BIC #177400,BINWRD
969 004554 013704 004714 25: MOV BINWRD,R4
970 004560 113705 004712 MOVB CHRCNT,R5
971 004564 012700 001416 MOV #TEMP,R0
972 004570 010403 35: 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 35
984 004624 012703 007544 MOV #MDATA,R3
985 004630 114023 45: MOVB -(R0),(R3)+
986 004632 105337 004712 DECB CHRCNT
987 004636 001374 BNE 45
988 004640 105737 004713 TSTB SPACNT
989 004644 001405 BEQ 65
990 004646 112723 000040 55: MOVB #040,(R3)+
991 004652 105337 004713 DECB SPACNT
992 004656 001373 BNE 55
993 004660 105013 65: CLRB (R3)
994 004662 104402 007544 TYPE ,MDATA
995 004666 005337 004710 DEC WRDCNT
996 004672 001313 BNE 15
997 004674 012605 MOV (SP)+,R5
998 004676 012604 MOV (SP)+,R4
999 004700 012603 MOV (SP)+,R3
:000 004702 012601 MOV (SP)+,R1
:001 004704 012600 MOV (SP)+,R0
:002 004706 000002 RTI
:003 004710 000000 WRDCNT: 0
:004 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      2(SP),(SP)          ;GET TRAP
1017 004730 006316          TRPOK: ASL      (SP)          ;MULTIPLY TRAP ARG BY 2
1018 004732 042716 177001          BIC      #177001,(SP)          ;CLEAR UNWANTED BITS
1019 004736 062716 00133C          ADD      #.TRPTAB,(SP)          ;POINTER TO SUBROUTINE ADDRESS
1020 004742 017616 000000          MOV      2(SP),(SP)          ;SUBROUTINE ADDRESS
1021 004746 00C136          JMP      2(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      #SW12,2SWR          ;BELL ON ERROR?
1028 004762 001406          BEQ      XBX          ;BR IF NO BELL
1029 004764 105777 174220          TSTB     2TPCSR          ;TTY READY?
1030 004770 100003          BPL      XBX          ;DON'T WAIT IF TTY NOT READY.
1031 004772 112777 000207 174212          MOVB     #207,2TPDBR          ;PUSH A BELL AT THE TTY.
1032 005000 032777 020000 174174          XBX: BIT      #SW13,2SWR          ;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$: SAVO5          ;SAVE ALL PROC REGISTERS
1039 005030 011605          MOV      (SP),R5          ;GET THE PC OF ERROR
1040 005032 162705 000002          SUB      #2,R5          ;GET ADDRESS OF TRAP CALL
1041 005036 011504          MOV      (R5),R4          ;GET HLT INSTRUCTION
1042 005040 006304          ASL      R4          ;MULT BY TWO
1043 005042 061504          ADD      (R5),R4          ;DOUBLE IT
1044 005044 006304          ASL      R4          ;MULT AGAIN
1045 005046 042704 177001          BIC      #177001,R4          ;CLEAR JUNK
1046 005052 062704 035372          ADD      #.ERRTAB,R4          ;GET POINTER
1047 005056 012437 005172          MOV      (R4)+,ERRMSG          ;GET ERROR MESSAGE
1048 005062 012437 005204          MOV      (R4)+,DATAHD          ;GET DATA HEADER
1049 005066 011437 005216          MOV      (R4),DATABP          ;GET DATA TABLE
1050 005072 105737 001325          TSTB     ERRFLG          ;TYPE HEADREER
1051 005076 001403          BEQ      TYPMSG          ;BR IF YES
1052 005100 005737 005216          TST      DATABP          ;DOES DATA TABLE EXIST?
1053 005104 001040          BNE      TYPDAT          ;BR IF YES.
1054 005106 104402 005672          TYPMSG: TYPE     ,MCRLF          ;
1055 005112 104402 005672          TYPE     ,MCRLF          ;
1056 005116 005737 001220          TST      LOCK          ;
1057 005122 001402          BEQ      1$          ;
1058 005124 104402 006142          TYPE     ,MASTEK          ;
1059 005130 104402 006130          1$: TYPE     ,MTSTN          ;
1060 005134 104411 005330          CNVRT     ,XTSTN          ;SHOW IT

```

004

DEDMF MACY11 30(1046) 11-JUL-77 11:59 PAGE 22
 DEEME.P11 12-MAY-77 14:18

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

PAGE: 0042

1061	005140	104402	006217		TYPE	,MERRPC	;TYPE PC.
1062	005144	104411	005322		CNVRT	,ERTABO	;SHOW IT
1063	005150	104402	005672		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	TYPDAT	;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: RES05		;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	JSWR	;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,JSWR	;GOTO TOP OF TEST?
1089	005264	001007			BNE	1\$	;BR IF YES
1090	005266	032777	002000	173706	BIT	#SW10,JSWR	;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	ARETURN	;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	005350	012706	001200		MOV	#STACK,SP	;RESET THE STACK POINTER

E04

02DMF MACY11 30(1046) 11-JUL-77 11:59 PAGE 23
 02DME.P11 12-MAY-77 14:18

PAGE: 0043

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

```

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          .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
1137 005450 104414              1$: ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1138 005452 121224              121224      ;PORT4+IBUS*11
1139 005454 032777 000020 173730      BIT     #BIT4,@DMP04      ;IS CLOCK BIT SET?
1140 005462 001772              BEQ     1$          ;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    @2(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$: BISB   #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
1156 005542 142777 000007 173636      BICB   #BIT2!BIT1!BIT0,@DMCSRH ;CLEAR ROM0, ROM1, STEP
1157 005550 000002              RTI
1158
1159 005552              .DATACLK:
1160 005552 013637 001416              MOV    @2(SP)+,TEMP      ;PUT TICK COUNT IN TEMP
1161 005556 062746 000002              ADD    #2,-(SP)          ;ADJUST STACK
1162 005562 152777 000020 173616      1$: 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    @2(SP)+,TEMP      ;MOVE COUNT TO TEMP
1172 005622 062746 000002              ADD    #2,-(SP)          ;ADJUST STACK

```

F04

DZDMF MACY11 30(1046) 11-JUL-77 11:59 PAGE 24
 DZDME.P11 12-MAY-77 14:18

PAGE: 0044

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

```

1173 005626 104414 000002 173552 1S: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1174 005626 021364 021364 ;PORT4+IBUS* REG11
1175 005630 032777 000002 173552 BIT #2, DMP04 ;IS PGM CLOCK BIT CLEAR?
1176 005632 001772 BEQ 1S ;BR IF YES
1177 005642 104414 2S: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1178 005642 021364 021364 ;PORT4+IBUS* REG11
1179 005644 032777 000002 173536 BIT #2, DMP04 ;IS PGM CLOCK BIT SET?
1180 005646 001372 BNE 2S ;BR IF YES
1181 005654 005337 001416 DEC TEMP ;DEC COUNT
1182 005662 001361 BNE 1S ;BR IF NOT DONE
1183 005664 000002 RTI ;RETURN
1184
1185
1186
1187 005666 020040 000077 MQM: .ASCIZ / ? /
1188 005672 005015 000 MCRLF: .ASCIZ (15)(12)
1189 005675 377 053520 020122 MPFAIL: .ASCIZ (377)/PWR FAILED. RESTART AT TEST /
1190 005733 377 047105 020104 MEFASS: .ASCIZ (377)/END PASS DZDMF /
1191 005755 377 000122 MR: .ASCIZ (377)/R/
1192 005760 047377 020117 042504 MERR2: .ASCIZ (377)/NO DEVICES PRESENT./
1193 006005 377 047111 052523 MERR3: .ASCIZ (377)/INSUFFICIENT DATA!/
1194 006031 377 042524 052123 MTSTPC: .ASCIZ (377)/TEST PC-/
1195 006043 377 047514 045503 MLOCK: .ASCIZ (377)/LOCK ON SELECTED TEST/
1196 006072 051503 035122 000040 MCSRX: .ASCIZ /CSR: /
1197 006100 042526 035103 000040 MVECX: .ASCIZ /VEC: /
1198 006106 040520 051523 051505 MPASSX: .ASCIZ /PASSES: /
1199 006117 105 051122 051117 MERRX: .ASCIZ /ERRORS: /
1200 006130 042524 052123 047040 MTSTN: .ASCIZ /TEST NO: /
1201 006142 000052 MASTEK: .ASCIZ /*/
1202 006144 051777 052105 051440 MNEW: .ASCIZ (377)/SET SWITCH REG TO DMC11'S DESIRED ACTIVE.
1203 006217 120 035103 000040 MERRPC: .ASCIZ /PC: /
1204 006224 020212 020040 020040 XHEAD: .ASCIZ (212)/
1205 006263 377 020040 020040 .ASCIZ (377)/
1206 006322 020212 050040 020103 .ASCIZ (212)/ PC CSR STAT1 STAT2 STAT3
1207 006374 026777 026455 026455 .ASCIZ (377)/-----
1208 006450 044377 053517 046440 NUM: .ASCIZ (377)/HOW MANY DMC11'S TO BE TESTED?
1209 006510 041777 051123 040440 CSR: .ASCIZ (377)/CSR ADDRESS?
1210 006526 053377 041505 047524 VEC: .ASCIZ (377)/VECTOR ADDRESS?
1211 006547 377 051102 050040 PRIO: .ASCIZ (377)/BR PRIORITY LEVEL? (4,5,6,7)?
1212 006606 044777 020106 046504 CRAM: .ASCIZ (377)/IF DMC HAS CRAM (M8204) TYPE "Y", IF CROM (M8200) TYPE "N"
1213 006704 053777 044510 044103 MODU: .ASCIZ (377)/WHICH LINE UNIT? IF NONE TYPE "N", IF M8201 TYPE "1", IF "
1214 007016 051777 044527 041524 LINE: .ASCIZ (377)/SWITCH PAC#1 (DDCMP LINE #)?
1215 007054 051777 044527 041524 BM: .ASCIZ (377)/SWITCH PAC#2 (BM873 BOOT ADD)?
1216 007114 044777 020123 044124 CONN: .ASCIZ (377)/IS THE LOOP BACK CONNECTOR ON?
1217 007154 047377 020117 042504 NOACT: .ASCIZ (377)/NO DEVICES ARE SELECTED
1218 007205 377 051412 051127 SWMES: .ASCIZ (377)(12)/SWR= /
1219 007215 116 053505 020077 SWMES1: .ASCIZ /NEW? /
1220 007223 377 042377 041515 CONERR: .ASCIZ (377)(377)/DMC11 FOUND AT NON-STANDARD ADDRESS PC:
1221 007277 377 054105 042520 CNERR: .ASCIZ (377)/EXPECTED FOUND/
1222 007320 024040 046504 024503 DMCM: .ASCIZ / (DMC) /
1223 007330 024040 046513 024503 KMCM: .ASCIZ / (KMC) /
1224 007340 042377 041515 030461 SPEED: .ASCIZ (377)/DMC11-AR.REMOTE,LOW SPEED) OR DMC11-AL.LOCAL,HIGH SPEED.
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
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
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207

```

G04

DDMMF MACY11 30(1046) 11-JUL-77 11:59 PAGE 25
 DDMMF.P11 12-MAY-77 14:18

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

PAGE: 0045

1190 007462 006 003
 1191 007464 001250
 1192 007466 006 003
 1193 007470 001252
 1194 007472 006 003
 1195 007474 001254
 1196 007476 006 002
 1197 007500 001256

.BYTE 6,3
 TEMP2
 .BYTE 6,3
 TEMP3
 .BYTE 6,3
 TEMP4
 .BYTE 6,2
 TEMPS

.EVEN

;BUFFERS FOR INPUT-OUTPUT

1202 007502 000000
 1203 007544
 1204 007544 000000
 1205 007606

INBUF: 0
 . = +40
 MDATA: 0
 . = +40

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

1212 007606 022737 000176 001202
 1213 007614 001077
 1214 007616 105777 171362
 1215 007622 100003
 1216 007624 012737 177777 003734
 1217 007632 022777 000007 171346
 1218 007640 001404
 1219 007642 022777 000207 171336
 1220 007650 001061
 1221 007652 010246
 1222 007654 010346
 1223 007656 010446
 1224 007660 012737 177777 010016
 1225 007666 005002
 1226 007670 012704 177777
 1227 007674 104402 007205
 1228 007700 104411
 1229 007702 010052
 1230 007704 104402 007215
 1231 007710 004737 010020
 1232 007714 022703 000015
 1233 007720 001424
 1234 007722 022703 000012
 1235 007726 001416
 1236 007730 022703 000025
 1237 007734 001754
 1238 007736 022703 000007
 1239 007742 001762
 1240 007744 005004
 1241 007746 042703 177770
 1242 007752 006302
 1243 007754 006302
 1244 007756 006302
 1245 007760 050302

CKSWR: CMP #SWREG,SWR ;IS THE SOFT SWR BEING USED?
 BNE CKSWR5 ;BR IF NO
 TSTB @TKCSR ;IS DONE SET?
 BPL 2\$;GO ON IF NOT SET
 MOV #-1,DONE ;IF DONE SET SET FLAG
 2\$: CMP #7,@TKDBR ;WAS CTRL G TYPED? 7 BIT ASCII?
 BEQ 1\$;BR IF YES
 CMP #207,@TKDBR ;WAS CTRL G TYPED? 8 BIT ASCII?
 BNE CKSWR5 ;BR IF NO
 1\$: MOV R2,-(SP) ;STORE R2
 MOV R3,-(SP) ;STORE R3
 MOV R4,-(SP) ;STORE R4
 MOV #-1,SWFLG ;SET SOFT TYPE OUT FLAG
 CKSWR1: CLR R2 ;CLEAR NEW SWR CONTENTS
 MOV #-1,R4 ;SET FLAG TO ALL ONES
 TYPE ,SWMES ;TYPE "SWR=" "
 CKSWR2: CNVRT ;TYPE OUT PRESENT CONTENTS
 SOFTSW ;OF SOFT SWITCH REGISTER
 CKSWR3: TYPE ,SWMES1 ;TYPE "NEW" "
 CKSWR4: JSR PC,INCHAR ;GET RESPONSE
 CMP #15,R3 ;WAS IT A CR?
 BEQ 5\$;BR IF YES
 CMP #12,R3 ;WAS IT A LF?
 BEQ 4\$;BR IF YES
 CMP #25,R3 ;WAS IT CTRL U?
 BEQ CKSWR1 ;BR IF YES(START OVER,
 CMP #7,R3 ;IF CNTL G GET NEXT CHAR
 BEQ CKSWR4
 CLR R4 ;IT MUST BE A DIGIT SO CLR FLAG
 BIC #177770,R3 ;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
 ASL R2 ;SHIFT R2 3 TIMES
 ASL R2
 ASL R2
 BIS R3,R2 ;ADD LAST DIGIT

H04

020MF MACY11 30(1046) 11-JUL-77 11:59 PAGE 26
 020ME.P11 12-MAY-77 14:18

PAGE: 0046

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

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			BNE	6\$	:IF NOT DON'T CHANGE SOFT SWR
1250	007776	010277	171200		MOV	R2,2SWR	:IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
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			CKSWRS: 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	8BIT7,R3	
1266	010050	000207			RTS	PC	
1267							
1268	010052	000001			SOFTSW: 1		
1269	010054	000	002		.BYTE	6,2	
1270	010056	000176			SWREG		

```

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

```

Address	Op Code	Op 1	Op 2	Op 3	Op 4	Op 5	Op 6	Op 7	Op 8	Op 9	Op 10	Op 11	Op 12	Op 13	Op 14	Op 15	Op 16	Op 17	Op 18	Op 19	Op 20	Op 21	Op 22	Op 23	Op 24	Op 25	Op 26	Op 27	Op 28	Op 29	Op 30	Op 31	Op 32	Op 33	Op 34	Op 35	Op 36	Op 37	Op 38	Op 39	Op 40	Op 41	Op 42	Op 43	Op 44	Op 45	Op 46	Op 47	Op 48	Op 49	Op 50	Op 51	Op 52	Op 53	Op 54	Op 55	Op 56	Op 57	Op 58	Op 59	Op 60	Op 61	Op 62	Op 63	Op 64	Op 65	Op 66	Op 67	Op 68	Op 69	Op 70	Op 71	Op 72	Op 73	Op 74	Op 75	Op 76	Op 77	Op 78	Op 79	Op 80	Op 81	Op 82	Op 83	Op 84	Op 85	Op 86	Op 87	Op 88	Op 89	Op 90	Op 91	Op 92	Op 93	Op 94	Op 95	Op 96	Op 97	Op 98	Op 99	Op 100	Op 101	Op 102	Op 103	Op 104	Op 105	Op 106	Op 107	Op 108	Op 109	Op 110	Op 111	Op 112	Op 113	Op 114	Op 115	Op 116	Op 117	Op 118	Op 119	Op 120	Op 121	Op 122	Op 123	Op 124	Op 125	Op 126	Op 127	Op 128	Op 129	Op 130	Op 131	Op 132	Op 133	Op 134	Op 135	Op 136	Op 137	Op 138	Op 139	Op 140	Op 141	Op 142	Op 143	Op 144	Op 145	Op 146	Op 147	Op 148	Op 149	Op 150	Op 151	Op 152	Op 153	Op 154	Op 155	Op 156	Op 157	Op 158	Op 159	Op 160	Op 161	Op 162	Op 163	Op 164	Op 165	Op 166	Op 167	Op 168	Op 169	Op 170	Op 171	Op 172	Op 173	Op 174	Op 175	Op 176	Op 177	Op 178	Op 179	Op 180	Op 181	Op 182	Op 183	Op 184	Op 185	Op 186	Op 187	Op 188	Op 189	Op 190	Op 191	Op 192	Op 193	Op 194	Op 195	Op 196	Op 197	Op 198	Op 199	Op 200	Op 201	Op 202	Op 203	Op 204	Op 205	Op 206	Op 207	Op 208	Op 209	Op 210	Op 211	Op 212	Op 213	Op 214	Op 215	Op 216	Op 217	Op 218	Op 219	Op 220	Op 221	Op 222	Op 223	Op 224	Op 225	Op 226	Op 227	Op 228	Op 229	Op 230	Op 231	Op 232	Op 233	Op 234	Op 235	Op 236	Op 237	Op 238	Op 239	Op 240	Op 241	Op 242	Op 243	Op 244	Op 245	Op 246	Op 247	Op 248	Op 249	Op 250	Op 251	Op 252	Op 253	Op 254	Op 255	Op 256	Op 257	Op 258	Op 259	Op 260	Op 261	Op 262	Op 263	Op 264	Op 265	Op 266	Op 267	Op 268	Op 269	Op 270	Op 271	Op 272	Op 273	Op 274	Op 275	Op 276	Op 277	Op 278	Op 279	Op 280	Op 281	Op 282	Op 283	Op 284	Op 285	Op 286	Op 287	Op 288	Op 289	Op 290	Op 291	Op 292	Op 293	Op 294	Op 295	Op 296	Op 297	Op 298	Op 299	Op 300	Op 301	Op 302	Op 303	Op 304	Op 305	Op 306	Op 307	Op 308	Op 309	Op 310	Op 311	Op 312	Op 313	Op 314	Op 315	Op 316	Op 317	Op 318	Op 319	Op 320	Op 321	Op 322	Op 323	Op 324	Op 325	Op 326	Op 327	Op 328	Op 329	Op 330	Op 331	Op 332	Op 333	Op 334	Op 335	Op 336	Op 337	Op 338	Op 339	Op 340	Op 341	Op 342	Op 343	Op 344	Op 345	Op 346	Op 347	Op 348	Op 349	Op 350	Op 351	Op 352	Op 353	Op 354	Op 355	Op 356	Op 357	Op 358	Op 359	Op 360	Op 361	Op 362	Op 363	Op 364	Op 365	Op 366	Op 367	Op 368	Op 369	Op 370	Op 371	Op 372	Op 373	Op 374	Op 375	Op 376	Op 377	Op 378	Op 379	Op 380</
---------	---------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	----------

```

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 010375 001000      1000
1334 010400 001226      TSTNO
1335 010402      000      .BYTE 0
1336 010403      001      .BYTE 1
1337 010404 012700 012320 MOV      #TST1,R0
1338 010410 022710 5$: CMP      (PC)+,(R0) ;CMP FIRST WORD TO 12737
1339 010412 012737      MOV      (PC)+,2(PC)+
1340 010414 001020      BNE      6$      ;BR IF NOT SAME
1341 010416 023760 001226 000002 CMP      TSTNO,2(R0) ;DOES TSTNO MATCH?
1342 010424 001014      BNE      6$      ;BR IF NO
1343 010426 022760 001226 000004 CMP      #TSTNO,4(R0) ;IS LAST WORD OK?
1344 010434 001010      BNE      6$      ;BR IF NO
1345 010436 010037 001214      MOV      R0,RETURN ;IT IS A LEGAL TEST SO DO IT
1346 010442 104402 005755      TYPE      ,MR
1347 010446 042737 000002 001236 BIC      #SW01,STRTSW
1348 010454 000412      BR      8$
1349 010456 005720 6$: TST      (R0)+ ;POP R0
1350 010460 020027 031460      CMP      R0,#TLAST+10 ;AT END YET?
1351 010464 001351      BNE      5$      ;BR IF NO
1352 010466 104402 005666      TYPE      ,MQM ;YES ILLEGAL TEST NO.
1353 010472 000730      BR      4$      ;TRY AGAIN
1354
1355 010474 012737 012320 001214 7$: MOV      #TST1,RETURN ;PREPARE RETURN ADDRESS
1356 010502 013701 001404 8$: MOV      DMC1R,R1 ;R1 = BASE DMC11 ADDRESS
1357 010506 000177 170502      JMP      @RETURN ;GO START TESTING.
1358
1359
1360      ;ROUTINE USED TO "AUTO SIZE" THE DMC11
1361      ;CSR AND VECTOR.
1362      ;NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
1363      ;ADDRESS RANGE (160000:164000)
1364      ;AND THE VECTOR MAY BE ANY WHERE IN THE
1365      ;FLOATING VECTOR RANGE (300:770)
1366      ;
1367      ;
1368
1368 010512      AUTO.SIZE:
1369 010512 000005      RESET
1370 010514 012702 001500 CSRMAP: MOV      #DM.MAP,R2 ;INSURE A BUS INIT.
1371 010520 005022 1$: CLR      (R2)+ ;LOAD MAP POINTER.
1372 010522 022702 001700      CMP      #DM.END,R2 ;ZERO ENTIRE MAP
1373 010526 001374      BNE      1$      ;ALL DONE?
1374 010530 005037 001310      CLR      DMNUM ;SET OCTAL NUMBER OF DMC11'S TO 3
1375 010534 012702 001500      MOV      #DM.MAP,R2 ;R2 POINTS TO DMC MAP
1376 010540 005037 001306      CLR      DMACTV ;CLEAR ACTIVE
1377 010544 032737 000001 001236 BIT      #SW00,STRTSW ;QUESTIONS?
1378 010552 001002      BNE      .+6 ;BR IF YES
1379 010554 000137 011252      JMP      7$ ;IF NO SKIP QUESTIONS
1380 010560 012737 000001 001256 MOV      #1,TEMPS ;START WITH 1
1381 010566 104403      INSTR
1382 010570 006450      NUM

```

K04

SDMF MACY11 30(1046) 11-JUL-77 11:59 PAGE 29
 SDME.P11 12-MAY-77 14:18

GENERAL UTILITIES (TYPEOUT, ERROR, SCOPE, ETC)

PAGE: 0049

1383	010572	104405			PARAM		
1384	010574	000001			1		
1385	010576	000020			16.		
1386	010600	001252			TEMP3		
1387	010602	000			.BYTE	0	
1388	010603	001			.BYTE	1	
1389	010604	013737	001252	001310	MOV	TEMP3, DMNUM	:DMNUM = HOW MANY
1390	010612	104402	005672		TYPE	, MCRLF	
1391	010616	104410			CONVRT		:TYPE WHICH DMC IS BEING DONE
1392	010620	012002			WHICH		:TEMP5 IS WHICH DMC
1393	010622	005237	001256		INC	TEMP5	
1394	010626	104403			INSTR		
1395	010630	006510			CSR		
1396	010632	104405			PARAM		
1397	010634	160000			160000		
1398	010636	164000			164000		
1399	010640	001254			TEMP4		
1400	010642	000			.BYTE	0	
1401	010643	001			.BYTE	1	
1402	010644	013722	001254		MOV	TEMP4, (R2)+	:STORE CSR IN MAP
1403	010650	104403			INSTR		
1404	010652	006526			VEC		
1405	010654	104405			PARAM		
1406	010656	000000			0		
1407	010660	000776			776		
1408	010662	001254			TEMP4		
1409	010664	000			.BYTE	0	
1410	010665	001			.BYTE	1	
1411	010666	013712	001254		MOV	TEMP4, (R2)	:STORE VECTOR IN MAP
1412	010672	104402			TYPE		
1413	010674	006547			PRI0		:ASK WHAT BR LEVEL
1414	010676	004737	012266		JSR	PC, INTTY	:GET RESPONSE
1415	010702	022703	000024		CMP	#24, R3	
1416	010706	101014			BHI	50\$	:BR IF LESS THAN 4
1417	010710	022703	000027		CMP	#27, R3	
1418	010714	103411			BLO	50\$	:BR IF GREATER THAN 7
1419	010716	012704	000011		MOV	#11, R4	:R4 = NUMBER OF SHIFTS
1420	010722	006303			ASL	R3	:SHIFT R3 LEFT
1421	010724	005304			DEC	R4	:DEC SHIFT COUNT
1422	010726	001375			BNE	-4	:BR IF NOT DONE
1423	010730	042703	170777		BIC	#170777, R3	:BIC UNWANTED BITS
1424	010734	050312			BIS	R3, (R2)	:PUT BR LEVEL IN STATUS MAP
1425	010736	000403			BR	8\$	:CONTINUE
1426	010740	104402			TYPE		
1427	010742	005666			MOVM		:RESPONSE IS OUT OF LIMITS
1428	010744	000752			BR	10\$	:TRY AGAIN
1429	010746	104402			TYPE		
1430	010750	006606			CRAM		:DOES DMC HAVE CRAM?
1431	010752	004737	012266		JSR	PC, INTTY	:GET REPLY
1432	010756	022703	000131		CMP	#131, R3	
1433	010762	001427			BEQ	9\$	:YES
1434	010764	022703	000116		CMP	#116, R3	:NO
1435	010770	001403			BEQ	40\$	:NOT A Y OR N
1436	010772	104402			TYPE		
1437	010774	005666			MOVM		:TYPE ""
1438	010776	000763			BR	9\$	:ASK AGAIN

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	MOVB	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	C:1242	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	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 DMCSR 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 DMCRIC 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	MOV	#125252,6(R1)	;WRITE THIS TO SEL6
1516	011322	052711	020000	BIS	#BIT13,(R1)	;WRITE IT!
1517	011326	022761	125252	CMP	#125252,4(R1)	;WAS IT WRITTEN?
1518	011334	001004		BNE	21\$	;IF NO IT IS NOT CROM
1519	011336	052762	100000	BIS	#BIT15,2(R2)	;SET BIT15 IF CROM
1520	011344	000431		BR	22\$	
1521	011346	012711	001000	21\$: MOV	#BIT9,(R1)	;SET ROM1
1522	011352	012761	100417	MOV	#100417,6(R1)	;PUT INSTRUCTION IN SEL6
1523	011360	012711	001400	MOV	#BIT9!BIT8,(R1)	;CLOCK INSTRUCTION (MICRO PROC PC TO C.
1524	011364	012711	002000	MOV	#BIT10,(R1)	;SET ROM0
1525	011370	022761	000626	CMP	#626,6(R1)	;IS IT LOCAL CROM
1526	011376	001411		BEQ	23\$	;BR IF YES
1527	011400	022761	016520	CMP	#16520,6(R1)	;IS IT REMOTE CROM?
1528	011406	001410		BEQ	22\$	;BR IF YES
1529	011410	022761	177777	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	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	MOV	#122113,6(R1)	;LOAD INSTRUCTION (CLR DTR)
1538	011450	052711	000400	BIS	#BIT8,(R1)	;CLOCK INSTRUCTION
1539	011454	012761	021264	MOV	#021264,6(R1)	;LOAD INSTRUCTION
1540	011462	052711	000400	BIS	#BIT8,(R1)	;CLOCK INSTRUCTION
1541	011466	122761	000377	CMPB	#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	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	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 CTR,
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 CCNNECTOR
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			TEMP5		
1600							
1601	012010	032737	000001	001236	VECMAP: BIT	#SW00,STRTSW	
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	J12044	012701	000302		MOV	#302,R1	;PC OF IOT INSTR.
1608	012050	010120		15:	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	15	;BR IF MORE TO FILL
1613	012066	013737	001306	001246	MOV	DMACTV,TEMP1	;STORE TEMPORALLY
1614	012074	006037	001246	25:	ROR	TEMP1	;BRING OUT A BIT
1615	012100	103063			BCC	55	;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 PORT6
1622	012136	012711	001400		MOV	#BIT9:BIT8,(R1)	;FORCE AN INTERRUPT
1623	012142	105200		75:	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	65	;BR IF R4 = 0
1627	012154	016437	012252	177776	MOV	BRLVL(R4),PS	;MOVE NEXT LOWER LEVEL IN PS
1628	012162	000767			BR	75	;BR TO DELAY
1629	012164	052762	005300	000002	65:	BIS	#5300,2(R2)
1630	012172	005011		35:	CLP	(R1)	;NO INTERRUPT ASSUME 300 AT LEVEL 5 AND FIX DMC11
1631	012174	062702	000010		ADD	#10,R2	;CLEAR ROMI
1632	012200	000735			BR	25	;POP SOFTWARE POINTER
1633	012202	051662	000002		45:	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	#35,(SP)	;POP IOT JUNK OFF STACK
1644	012246	000002			RTI		;SET FOR RETURN
1645	012250	000207		55:	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:BIT5,R3	;MASK OFF LOWER CASE
1662	012316	000207			RTS	PC	;RETURN

C2100

```

1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675 012320 012737 000001 001226
1676 012326 012737 012374 001216
1677
1678 012334 005077 167344
1679 012340 012702 000011
1680 012344 104414
1681 012346 021224
1682 012350 016104 000004
1683 012354 042704 000054
1684 012360 012705 000020
1685 012364 120504
1686 012366 001401
1687 012370 104002
1688 012372 104400
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699 012374 012737 000002 001226
1700 012402 012737 012442 001216
1701
1702 012410 012702 000012
1703 012414 104414
1704 012416 021244
1705 012420 016104 000004
1706 012424 042704 000017
1707 012430 005005
1708 012432 120504
1709 012434 001401
1710 012436 104002
1711 012440 104400
1712
1713
1714
1715
1716
1717
1718

```

```

***** 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 SEL0
      ROMCLK             ;SAVE R2 FOR TYPEOUT
                        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
      021004!<20*11>    ;PORT4+LINE UNIT REG 11
      MOV 4(R1),R4        ;PUT "FOUND" IN R4
      BIC #54,R4         ;CLEAR UNKNOWN BITS
      MOV #20,R5         ;PUT "EXPECTED" IN R5
      CMPB R5,R4         ;IS OUT READY SET?
      BEQ 1$            ;BR IF YES
      HLT 2             ;ERROR IN LU 11
1$:    SCOPE             ;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
                        ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
      021004!<20*12>    ;PORT4+LINE UNIT REG 12
      MOV 4(R1),R4        ;PUT "FOUND" IN R4
      BIC #17,R4         ;CLEAR UNKNOWN BITS
      CLR R5             ;PUT "EXPECTED" IN R5
      CMPB R5,R4         ;ARE ALL BITS CLEARED?
      BEQ 1$            ;BR IF YES
      HLT 2             ;ERROR IN LU 12
1$:    SCOPE             ;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                                ;R1 CONTAINS BASE DMC11 ADDRESS
      MOV #13,R2                          ;MASTER CLEAR DMC11
      ROMCLK                               ;SAVE R2 FOR TYPEOUT
      021004!(<20*13>)                   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
      MOV 4(R1),R4                         ;PORT4+LINE UNIT REG 13
      BIC #213,R4                         ;PUT "FOUND" IN R4
      MOV #100,R5                         ;CLEAR UNKNOWN BITS
      CMPB R5,R4                         ;PUT "EXPECTED" IN R5
      BEQ 1$                             ;ARE RING, DTR, AND MODEM READY SET?
      HLT 2                               ;BR IF YES
      SCOPE                             ;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                                ;R1 CONTAINS BASE DMC11 ADDRESS
      MOV #17,R2                          ;MASTER CLEAR DMC11
      ROMCLK                               ;SAVE R2 FOR TYPEOUT
      021004!(<20*17>)                   ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
      MOV 4(R1),R4                         ;PORT4+LINE UNIT REG 17
      BIC #206,R4                         ;PUT "FOUND" IN R4
      MOV #51,R5                         ;CLEAR UNKNOWN BITS
      BIT #BIT13,STAT1                   ;PUT "EXPECTED" IN R5
      BNE 2$                             ;IS LU AN M8202 OR M8201?
      BIT #BIT14,STAT1                   ;BR IF M8202
      BNE 3$                             ;CONNECTOR???
      BIC #40,R4                         ;BR IF M8201 WITH CONNECTOR
      BIC #BIT5,R5                       ;MASK OFF SI BIT IF M8202 OR M8201, NO CONNECTOR
      CMPB R5,R4                         ;SI BIT IS UNKNOWN
      BEQ 1$                             ;ARE SI AND ICIR SET?
      HLT 2                               ;BR IF YES
      SCOPE                             ;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
:*****

```

E05

DDMF MACY11 30,1046) 11-JUL-77 11:59 PAGE 36
 DDME.P11 12-MAY-77 14:18 LINE UNIT WRITE/READ TESTS

PAGE: 0056

```

1775
1776
1777 012616 012737 000005 001226
1778 012624 012737 012756 001216
1779 012632 012737 012646 001220
1780
1781 012640 104412
1782 012642 012702 000012
1783 012646 012761 000040 000004
1784 012654 104414
1785 012656 122112
1786 012660 104414
1787 012662 021245
1788 012664 012705 000040
1789 012670 116104 000005
1790 012674 042704 000337
1791 012700 120504
1792 012702 001401
1793 012704 104003
1794 012706 104401
1795 012710 012737 012716 001220
1796 012716 005061 000004
1797 012722 104414
1798 012724 122112
1799 012726 104414
1800 012730 021245
1801 012732 005005
1802 012734 116104 000005
1803 012740 042704 000337
1804 012744 120504
1805 012746 001401
1806 012750 104003
1807 012752 104401
1808 012754 104400
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819 012756 012737 000006 001226
1820 012764 012737 013116 001216
1821 012772 012737 013006 001220
1822
1823 013000 104412
1824 013002 012702 000017
1825 013006 012761 000001 000004
1826 013014 104414
1827 013016 122117
1828 013020 104414
1829 013022 021365
1830 013024 012705 000001

; TEST 5
-----
TST5: MOV #5,TSTNO
      MOV #TST6,NEXT
      MOV #15,LOCK

      MSTCLR
      MOV #12,R2
15:    MOV #40,4(R1)
      ROMCLK
      122112
      ROMCLK
      021245
      MOV #40,R5
      MOVB 5(R1),R4
      BIC #337,R4
      CMPB R5,R4
      BEQ 25
      HLT 3
25:    SCOP1
      MOV #35,LOCK
35:    CLR 4(R1)
      ROMCLK
      122112
      ROMCLK
      021245
      CLR R5
      MOVB 5(R1),R4
      BIC #337,R4
      CMPB R5,R4
      BEQ 45
      HLT 3
45:    SCOP1
      SCOPE

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SAVE REGISTER ADDRESS FOR TYPEOUT
;LOAD PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET BITS IN LU-12
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;READ LU-12
;PUT "EXPECTED" IN R5
;PUT "FOUND" IN R4
;CLEAR UNWANTED BITS
;IS BITS SET?
;BR IF YES
;ERROR, BIT 5 IS NOT SET
;SCOPE SUBTEST (SW09=1)
;NEW SCOPE
;LOAD PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;CLEAR BIT 5 IN LU-12
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;READ LU-12
;PUT "EXPECTED" IN R5
;PUT "FOUND" IN R4
;CLEAR UNWANTED BITS
;IS BITS CLEAR?
;BR IF YES
;ERROR, BITS IS NOT CLEAR
;SCOPE SUBTEST (SW09=1)
;SCOPE THIS TEST

;***** 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
;*****

; TEST 6
-----
TST6: MOV #6,TSTNO
      MOV #TST7,NEXT
      MOV #15,LOCK

      MSTCLR
      MOV #17,R2
15:    MOV #1,4(R1)
      ROMCLK
      122117
      ROMCLK
      021365
      MOV #1,R5

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SAVE REGISTER ADDRESS FOR TYPEOUT
;LOAD PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET BIT1 IN LU-17
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;READ LU-17
;PUT "EXPECTED" IN R5

```

LINE UNIT WRITE/READ TESTS

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			2\$:	SCOPI		: SCOPE SUBTEST (SW09=1)
1837	013050	012737	013056	001220		MOV	#3\$, LOCK	: NEW SCOPI
1838	013056	005061	000004		3\$:	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			4\$:	SCOPI		: SCOPE SUBTEST (SW09=1)
1850	013114	104400				SCOPE		: SCOPE THIS TEST
1851								
1852								
1853								: ***** TEST 7 *****
1854								: *LINE UNIT REGISTER WRITE/READ TEST
1855								: *FLOAT A 1 THROUGH LINE UNIT REGISTER 13
1856								: *FLOAT A 0 THROUGH LINE UNIT REGISTER 13
1857								: *****
1858								
1859								
1860								: TEST 7
1861	013116	012737	000007	001226				
1862	013124	012737	013326	001216	TST7:	MOV	#7, TSTNO	
1863	013132	012737	013152	001220		MOV	#TST10, NEXT	
1864						MOV	#64\$, LOCK	
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 "PECL"
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			65\$:	SCOPI		: 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	#63\$, LOCK	: NEW SCOPI

LINE UNIT WRITE/READ TESTS

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 :
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,TSTNC	:***** TEST 10 *****
1920	013334	012737	013502	001216	MOV	#TST11,NEXT	:*LINE UNIT REGISTER WRITE/READ TEST
1921	013342	012737	013362	001220	MOV	#64\$,LOCK	:*FLOAT A 1 THROUGH LINE UNIT REGISTER 14
1922							:*FLOAT A 0 THROUGH LINE UNIT REGISTER 14
1923	013350	104412					:*****
1924	013352	012702	000014				
1925	013356	012700	000001				
1926	013362						
1927	013362	010061	000004		64\$: 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			ROLB	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

H05

DDMMF MACY11 30(1046) 11-JUL-77 11:59 PAGE 39
 DDMMF.P11 12-MAY-77 14:18 LINE UNIT WRITE/READ TESTS

PAGE: 0059

```

1943 013434 005100          69$: COM      R0          :CHANGE TO FLOATING ZERO
1944 013436          67$:          :
1945 013436 010061 000004    MOV      R0,R1          :PUT PATTERN INTO PORT4
1946 013442 104414          ROMCLK          :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1947 013444 122114          122100!14      :MOV DATA TO IBUS REGISTER 14
1948 013446 104414          ROMCLK          :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1949 013450 021305          21005!<14*20>  :READ FROM IBUS REGISTER 14
1950 013452 010005          MOV      R0,R5          :PUT EXPECTED IN R5
1951 013454 116104 000005    MOVB     5(R1),R4      :PUT "FOUND" INTO R4
1952 013460 120504          CMPB     R5,R4          :DATA CORRECT?
1953 013462 001401          BEQ      68$          :BR IF YES
1954 013464 104003          HLT      3              :ERROR
1955 013466 104401          69$: SCOP1          :SW09=1?
1956 013470 005100          COM      R0          :CHANGE TO FLOATING 1
1957 013472 000241          CLC              :CLEAR CARRY
1958 013474 106100          ROLB     R0          :SHIFT BIT IN R0
1959 013476 001356          BNE     69$          :IF R0=0 THEN DONE
1960 013500 104400          SCOPE          :SCOPE THIS TEST
1961
1962
1963          :***** TEST 11 *****
1964          :*SWITCH PAC TEST
1965          :*THIS TEST READS SWITCH PAC#1
1966          :*THIS SWITCH PAC CONTAINS THE DDCMP LINE #
1967          :*****
1968
1969          : TEST 11
1970          :-----
1971 013502 012737 000011 001226    TST11: MOV      #11,TSTNO
1972 013510 012737 013544 001216    MOV      #TST12,NEXT
1973
1974 013516 104412          MSTCLR          :R1 CONTAINS BASE DMC11 ADDRESS
1975 013520 104414          ROMCLK          :MASTER CLEAR DMC11
1976 013522 021324          021324          :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1977 013524 016104 000004    MOV      4(R1),R4      :PORT4+LUI5
1978 013530 113705 001370    MOVB     STAT2,R5      :PUT "FOUND" IN R4
1979 013534 120504          CMPB     R5,R4          :PUT "EXPECTED" IN R5
1980 013536 001401          BEQ      1$          :SW OK?
1981 013540 104031          HLT      31          :BR IF YES
1982 013542 104400          1$: SCOPE          :ERROR, SWITCH PAC READ ERROR
1983          :SCOPE THIS TEST
1984
1985          :***** TEST 12 *****
1986          :*SWITCH PAC TEST
1987          :*THIS TEST READS SWITCH PAC#2
1988          :*THIS SWITCH PAC CONTAINS THE BM873 BOOT ADD
1989          :*****
1990
1991          : TEST 12
1992          :-----
1993 013544 012737 000012 001226    TST12: MOV      #12,TSTNO
1994 013552 012737 013606 001216    MOV      #TST13,NEXT
1995
1996 013560 104412          MSTCLR          :R1 CONTAINS BASE DMC11 ADDRESS
1997 013562 104414          ROMCLK          :MASTER CLEAR DMC11
1998 013564 021344          021344          :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1999          :PORT4+LUI6

```

```

1999 013566 016104 000004      MOV      4(R1),R4      ;PUT "FOUND" IN R4
2000 013572 113705 001371      MOVVB   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 REAC ERROR
2004 013604 104400              1$: SCOPE              ;SCOPE THIS TEST
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015 013606 012737 000013 001226      TST13: MOV      #13,TSTNO
2016 013614 012737 013706 001216      MOV      #TST14,NEXT
2017
2018 013622 104412              MSTCLR                     ;R1 CONTAINS BASE DMC11 ADDRESS
2019 013624 005037 001416              CLR      TEMP          ;MASTER CLEAR DMC11
2020 013630              1$:              ;PREPARE FOR DELAY
2021 013630 104414              ROMCLK                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=53C4
2022 013632 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=53C4
2031 013662 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
2041
2042
2043
2044
2045
2046
2047
2048
2049 013706 012737 000014 001226      TST14: MOV      #14,TSTNO
2050 013714 012737 014022 001216      MOV      #TST15,NEXT
2051
2052 013722 104412              MSTCLR                     ;R1 CONTAINS BASE DMC11 ADDRESS
2053 013724 005061 000004              CLR      4(R1)          ;MASTER CLEAR DMC11
2054 013730 104414              ROMCLK                     ;CLEAR PORT4
                                ;NEXT WORD IS INSTRUCTION, ROMCLK PC=53C4

```


K05

2111	014140	012702	000013	15:	MOV	#13,R2	:SAVE ADDRESS FOR TYPEOUT
2112	014140				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2113	014144	104414			021264		:PORT4+LU 13
2114	014146	021264			MOV	4(R1),R4	:PUT EXPECTED IN R4
2115	014150	016104	000004		BIC	#337,R4	:CLEAR UNWANTED BITS
2116	014154	042704	000337		MOV	#BITS,R5	:PUT "EXPECTED" IN R5, RTS SHOULD BE SET
2117	014160	012705	000040		CMPB	R5,R4	:IS R5 OK?
2118	014164	120504			BEQ	25	:BR IF YES
2119	014166	001401			HLT	5	:RTS ERROR
2120	014170	104005		25:	SCOPE		:SCOPE THIS TEST
2121	014172						
2122	014172	104400					
2123							
2124							
2125							
2126							
2127							
2128							
2129							
2130							
2131							
2132							
2133							
2134	014174	012737	000016	001226	TST16:	MOV	#16,TSTNO
2135	014202	012737	014406	001216		MOV	#TST17,NEXT
2136							
2137	014210	104412					
2138	014212	005061	000004		MSTCLR		:R1 CONTAINS BASE DMC11 ADDRESS
2139	014216	104414			CLR	4(R1)	:MASTER CLEAR DMC11
2140	014220	122117			ROMCLK		:CLEAR PORT4
2141	014222	004737	033374		122117		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2142	014226	012711	004000		JSR	PC,CLR10	:PUT LINE UNIT IN BITSTUFF MODE
2143	014232	012761	000001	000004	MOV	#BIT11,(R1)	:DO THIS AFTER MODE IS SET
2144	014240	104414			MOV	#1.4(R1)	:SET LINE UNIT LOOP
2145	014242	122111			ROMCLK		:LOAD PORT4 WITH BIT0
2146	014244	104414			122111		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2147	014246	122110			ROMCLK		:SET SOM
2148	014250	004737	032044		122110		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2149	014254	104415	000002		JSR	PC,OCOR	:LOAD OUT DATA SILO
2150	014260	012761	000200	000004	DATACLK,	2	:WAIT FOR OCOR
2151	014266	104414			MOV	#BIT7.4(R1)	:CLOCK DATA FOUR TIMES
2152	014270	122111			ROMCLK		:SET BIT7 IN PORT4
2153	014272	104415	000001		122111		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2154	014276	012702	000017		DATACLK,	1	:SET OUT CLEAR
2155	014302	104414			MOV	#17.R2	:GIVE A TICK TO CLEAR RTS
2156	014304	021364			ROMCLK		:SAVE ADDRESS FOR TYPEOUT
2157	014306	016104	000004		021364		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2158	014312	042704	000357		MOV	4(R1),R4	:PORT4+LU 17
2159	014316	005005			BIC	#357,R4	:PUT "FOUND" IN R4
2160	014320	120504			CLR	R5	:CLEAR UNWANTED BITS
2161	014322	001401			CMPB	R5,R4	:PUT "EXPECTED" IN R5
2162	014324	104005			BEQ	15	:IS OCOR CLEARED?
2163	014326				HLT	5	:BR IF YES
2164	014328	012702	000013	15:	MOV	#13.R2	:SAVE ADDRESS FOR TYPEOUT
2165	014332	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2166	014334	021264			021264		:PORT4+LU 13

BASIC TRANSMITTER TESTS

2167	014336	016104	000004		MOV	4(R1),R4	;PUT EXPECTED IN R4
2168	014342	042704	000337		BIC	#337,R4	;CLEAR UNWANTED BITS
2169	014346	005005			CLR	R5	;PUT "EXPECTED" IN R5. RTS SHOULD BE CLEARED
2170	014350	120504			CMPB	R5,R4	;IS RTS OK?
2171	014352	001401			BEQ	2\$	;BR IF YES
2172	014354	104005			HLT	5	;RTS ERROR
2173	014356			2\$:			
2174	014356	012702	000011		MOV	#11,R2	;SAVE ADDRESS FOR TYPEOUT
2175	014362	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2176	014364	021224			021224		;PORT4+LUI1
2177	014366	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
2178	014372	012705	000020		MOV	#BIT4,R5	;ONLY OUT READY SHOULD BE SET
2179	014376	120504			CMPB	R5,R4	;IS ACTIVE CLEAR?
2180	014400	001401			BEQ	3\$	;BR IF YES
2181	014402	104005			HLT	5	;ERROR ACTIVE NOT CLEARED
2182	014404			3\$:			
2183	014404	104400			SCOPE		;SCOPE THIS TEST
2184							
2185							
2186							
2187							
2188							
2189							
2190							
2191							
2192							
2193							
2194							
2195							
2196							
2197	014406	012737	000017	001226	TST17:	MOV	#17,TSTNO
2198	014414	012737	014670	001216		MOV	#TST20,NEXT
2199							
2200	014422	104412			MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
2201	014424	005061	000004		CLR	4(R1)	;MASTER CLEAR DMC11
2202	014430	104414			ROMCLK		;CLEAR PORT4
2203	014432	122117			122117		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2204	014434	004737	033374		JSR	PC,CLR10	;PUT LINE UNIT IN BITSTUFF MODE
2205	014440	005037	033612		CLR	BITCON	;DO THIS AFTER MODE IS SET
2206	014444	012711	004000		MOV	#BIT11,(R1)	;CONSECUTIVE 1'S COUNTER INIT TO 0
2207	014450	004737	032176		JSR	PC,OUTRDY	;SET LINE UNIT LOOP
2208	014454	012761	000001	000004	MOV	#1.4(R1)	;WAIT FOR OUT-READY
2209	014462	104414			ROMCLK		;SET BIT0 IN PORT4
2210	014464	122111			122111		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2211	014466	104414			ROMCLK		;SET SOM!
2212	014470	122110			122110		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2213	014472	012705	000000		MOV	#0,R5	;LOAD GARBAGE CHAR
2214	014476	004737	032176		JSR	PC,OUTRDY	;LOAD CHARACTER IN R5 FOR TYPEOUT
2215	014502	010561	000004		MOV	R5,4(R1)	;WAIT FOR OUT-READY
2216	014506	104414			ROMCLK		;LOAD PORT4 WITH CHARACTER
2217	014510	122110			122110		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2218	014512	004737	032044		JSR	PC,OCOR	;LOAD OUT DATA
2219	014516	005003			CLR	R3	;WAIT FOR OCOR TO SET
2220	014520	010502			MOV	R5,R2	;CLEAR BIT COUNTER
2221	014522	104415	000002		DATACLK,	2	;LOAD CHARACTER IN R2
2222	014526	012737	000176	001252	MOV	#1B<01111110>,TEMP3	;2 TICKS TO SET UP TRANSMITTER
							;PUT FLAG CHARACTER IN TEMP3

2223	014534	104415	000001	64\$:	DATACLK,	1	;CLOCK FLAG ONCE
2224	014540	106037	001252		RORB	TEMP3	;SHIFT SOFT FLAG
2225	014544	103405			BCS	65\$	;BR IF BIT IS MARK
2226	014546	004737	032012		JSR	PC,GETSI	;LOOK AT BIT WINDOW
2227	014552	103006			BCC	66\$	;BR IF OK
2228	014554	104026			HLT	26	;ERROR IN FLAG CHAR
2229	014556	000404			BR	66\$	
2230	014560	004737	032012	65\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
2231	014564	103401			BCS	66\$	;BR IF OK
2232	014566	104026			HLT	26	;ERROR IN FLAG CHAR
2233	014570	005203		66\$:	INC	R3	;INC BIT COUNT
2234	014572	022703	000010		CMP	#10,R3	;FLAG DONE YET?
2235	014576	001356			BNE	64\$	;BR IF NO
2236	014600	005003			CLR	R3	;CLEAR BIT COUNT
2237	014602	104415	000001	1\$:	DATACLK,	1	;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2238	014606	106002			RORB	R2	;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2239	014610	103005			BCC	2\$	;BR IF CARRY CLEAR
2240	014612	004737	032012		JSR	PC,GETSI	;GET THE WINDOW
2241	014616	103406			BCS	3\$	;BR IF BIT IS A MARK
2242	014620	104006			HLT	6	;ERROR BIT WAS A SPACE
2243	014622	000404			BR	3\$	;CONTINUE WITH TEST
2244	014624	004737	032012	2\$:	JSR	PC,GETSI	;GET THE WINDOW
2245	014630	103001			BCC	3\$	;BR IF BIT IS A SPACE
2246	014632	104006			HLT	6	;ERROR BIT WAS A MARK
2247	014634			3\$:			
2248	014634	005203			INC	R3	;NEXT BIT
2249	014636	022703	000010		CMP	#10,R3	;DONE YET?
2250	014642	001357			BNE	1\$	;BR IF NO
2251	014644	104415	000014		DATACLK,	14	;CLOCK TRANSMITTER 14 MORE TICKS
2252	014650	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2253	014652	021264			021264		;PORT4+LU-13
2254	014654	032761	000040 000004		BIT	#BIT5.4(R1)	;RTS SHOULD BE CLEAR NOW
2255	014662	001401			BEG	4\$	;BR IF YES
2256	014664	104034			HLT	34	;ERROR, RTS NOT CLEAR
2257	014666	104400		4\$:	SCOPE		;SCOPE THIS TEST

2258
 2259
 2260 ;***** TEST 20 *****
 2261 ;*BITSTUFF TRANSMITTER TEST
 2262 ;*SINGLE CLOCK THE CHARACTER 125
 2263 ;*CHECK FLAG AND DATA IN THE BIT WINDOW
 2264 ;*VERIFY EACH BIT POSITION AS IT
 2265 ;*PASSES THE BIT WINDOW (SI BIT)
 2266 ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
 2267 ;*****
 2268
 2269
 2270 ; TEST 20

2271	014670	012737	000020	001226	TST20:	MOV	#20,TSTNO	
2272	014676	012737	015152	001216		MOV	#TST21,NEXT	
2273								
2274	014704	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
2275	014706	005061	000004			CLR	4(R1)	;MASTER CLEAR DMC11
2276	014712	104414				ROMCLK		;CLEAR PORT4
2277	014714	122117				122117		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2278	014716	004737	033374			JSR	PC,CLR10	;PUT LINE UNIT IN BITSTUFF MODE
								;DO THIS AFTER MODE IS SET

2279	014722	005037	033612	CLR	BITCON	:CONSECUTIVE 1'S COUNTER INIT TO 0
2280	014726	012711	004000	MOV	#BIT11,R1	:SET LINE UNIT LOOP
2281	014732	004737	032176	JSR	PC,OUTRDY	:WAIT FOR OUT-READY
2282	014736	012761	000001 000004	MOV	#1,4(R1)	:SET BIT0 IN PORT4
2283	014744	104414		ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2284	014746	122111		122111		:SET SOM!
2285	014750	104414		ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2286	014752	122110		122110		:LOAD GARBAGE CHAR
2287	014754	012705	000125	MOV	#125,R5	:LOAD CHARACTER IN R5 FOR TIMEOUT
2288	014760	004737	032176	JSR	PC,OUTRDY	:WAIT FOR OUT-READY
2289	014764	010561	000004	MOV	R5,4(R1)	:LOAD PORT4 WITH CHARACTER
2290	014770	104414		ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2291	014772	122110		122110		:LOAD OUT DATA
2292	014774	004737	032044	JSR	PC,OCOR	:WAIT FOR OCOR TO SET
2293	015000	005003		CLR	R3	:CLEAR BIT COUNTER
2294	015002	010502		MOV	R5,R2	:LOAD CHARACTER IN R2
2295	015004	104415	000002	DATACLK,	2	:2 TICKS TO SET UP TRANSMITTER
2296	015010	012737	000176 001252	MOV	#18<01111110>,TEMP3	:PUT FLAG CHARACTER IN TEMP3
2297	015016	104415	000001	DATACLK,	1	:CLOCK FLAG ONCE
2298	015022	106037	001252	RORB	TEMP3	:SHIFT SOFT FLAG
2299	015026	103405		BCS	65\$	:BR IF BIT IS MARK
2300	015030	004737	032012	JSR	PC,GETSI	:LOOK AT BIT WINDOW
2301	015034	103006		BCC	66\$	:BR IF OK
2302	015036	104026		HLT	26	:ERROR IN FLAG CHAR
2303	015040	000404		BR	66\$	
2304	015042	004737	032012	JSR	PC,GETSI	:LOOK AT BIT WINDOW
2305	015046	103401		BCS	66\$	:BR IF OK
2306	015050	104026		HLT	26	:ERROR IN FLAG CHAR
2307	015052	005203		INC	R3	:INC BIT COUNT
2308	015054	022703	000010	CMP	#10,R3	:FLAG DONE YET?
2309	015060	001356		BNE	64\$	:BR IF NO
2310	015062	005003		CLR	R3	:CLEAR BIT COUNT
2311	015064	104415	000001	DATACLK,	1	:SHIFT NEXT BIT IN THE WINDOW (5: BIT)
2312	015070	106002		RORB	R2	:SHIFT NEXT SOFTWARE BIT IN TO CARRY
2313	015072	103005		BCC	2\$	:BR IF CARRY CLEAR
2314	015074	004737	032012	JSR	PC,GETSI	:GET THE WINDOW
2315	015100	103406		BCS	3\$	:BR IF BIT IS A MARK
2316	015102	104006		HLT	6	:ERROR BIT WAS A SPACE
2317	015104	000404		BR	3\$	:CONTINUE WITH TEST
2318	015106	004737	032012	JSR	PC,GETSI	:GET THE WINDOW
2319	015112	103001		BCC	3\$	:BR IF BIT IS A SPACE
2320	015114	104006		HLT	6	:ERROR BIT WAS A MARK
2321	015116		3\$:			
2322	015116	005203		INC	R3	:NEXT BIT
2323	015120	022703	000010	CMP	#10,R3	:DONE YET?
2324	015124	001357		BNE	1\$	:BR IF NO
2325	015126	104415	000014	DATACLK,	14	:CLOCK TRANSMITTER 14 MORE TICKS
2326	015132	104414		ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2327	015134	021264		021264		:PORT4=LU-13
2328	015136	032761	000040 000004	BIT	#BIT5,4(R1)	:RTS SHOULD BE CLEAR NOW
2329	015144	001401		BEG	4\$	:BR IF YES
2330	015146	104034		HLT	34	:ERROR, RTS NOT CLEAR
2331	015150	104400		4\$:		:SCOPE THIS TEST
2332				SCOPE		
2333						
2334						

;***** TEST 21 *****

```

2335 ;*BITSTUFF TRANSMITTER TEST
2336 ;*SINGLE CLOCK THE CHARACTER 252
2337 ;*CHECK FLAG AND DATA IN THE BIT WINDOW
2338 ;*VERIFY EACH BIT POSITION AS IT
2339 ;*PASSES THE BIT WINDOW (SI BIT)
2340 ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2341 ;*****
2342
2343 ; TEST 21
2344 ;-----
2345 015152 012737 000021 001226 TST21: MOV #21,TSTNO
2346 015160 012737 015434 001216 MOV #TST22,NEXT
2347
2348 015166 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
2349 015170 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
2350 015174 104414 ROMCLK ;CLEAR PORT4
2351 015176 122117 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2352 015200 004737 033374 JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
2353 015204 005037 033612 CLR BITCON ;DO THIS AFTER MODE IS SET
2354 015210 012711 004000 MOV #BIT11,(R1) ;CONSECUTIVE 1'S COUNTER INIT TO 0
2355 015214 004737 032176 JSR PC,OUTRDY ;SET LINE UNIT LOOP
2356 015220 012761 000001 000004 MOV #1,4(R1) ;WAIT FOR OUT-READY
2357 015226 104414 ROMCLK ;SET BIT0 IN PORT4
2358 015230 122111 122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2359 015232 104414 ROMCLK ;SET SOM!
2360 015234 122110 122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2361 015236 012705 000252 MOV #252,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
2362 015242 004737 032176 JSR PC,OUTRDY ;WAIT FOR OUT-READY
2363 015246 010561 000004 MOV R5,4(R1) ;LOAD PORT4 WITH CHARACTER
2364 015252 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2365 015254 122110 122110 ;LOAD OUT DATA
2366 015256 004737 032044 JSR PC,OCOR ;WAIT FOR OCOR TO SET
2367 015262 005003 CLR R3 ;CLEAR BIT COUNTER
2368 015264 010502 MOV R5,R2 ;LOAD CHARACTER IN R2
2369 015266 104415 000002 DATACLK, 2 ;2 TICKS TO SET UP TRANSMITTER
2370 015272 012737 000176 001252 MOV #1B<01111110>,TEMP3 ;PUT FLAG CHARACTER IN TEMP3
2371 015300 104415 000001 64$: DATACLK, 1 ;CLOCK FLAG ONCE
2372 015304 106037 001252 RORB TEMP3 ;SHIFT SOFT FLAG
2373 015310 103405 BCS 65$ ;BR IF BIT IS MARK
2374 015312 004737 032012 JSR PC,GETSI ;LOOK AT BIT WINDOW
2375 015316 103006 BCC 66$ ;BR IF OK
2376 015320 104026 HLT 26 ;ERROR IN FLAG CHAR
2377 015322 000404 BR 66$
2378 015324 004737 032012 65$: JSR PC,GETSI ;LOOK AT BIT WINDOW
2379 015330 103401 BCS 66$ ;BR IF OK
2380 015332 104026 HLT 26 ;ERROR IN FLAG CHAR
2381 015334 005203 66$: INC R3 ;INC BIT COUNT
2382 015336 022703 000010 CMP #10,R3 ;FLAG DONE YET?
2383 015342 001356 BNE 64$ ;BR IF NO
2384 015344 005003 CLR R3 ;CLEAR BIT COUNT
2385 015346 104415 000001 1$: DATACLK, 1 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2386 015352 106002 RORB R2 ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2387 015354 103005 BCC 2$ ;BR IF CARRY CLEAR
2388 015356 004737 032012 JSR PC,GETSI ;GET THE WINDOW
2389 015362 103406 BCS 3$ ;BR IF BIT IS A MARK
2390 015364 104006 HLT 6 ;ERROR BIT WAS A SPACE

```

2391	015366	000404			BR	3\$		:CONTINUE WITH TEST
2392	015370	004737	032012		2\$: JSR	PC,GETSI		:GET THE WINDOW
2393	015374	103001			BCC	3\$		:BR IF BIT IS A SPACE
2394	015376	104006			HLT	6		:ERROR BIT WAS A MARK
2395	015400				3\$: INC	R3		:NEXT BIT
2396	015400	005203			CMP	#10,R3		:DONE YET?
2397	015402	022703	000010		BNE	1\$		:BR IF NO
2398	015406	001357			DATACLK,	14		:CLOCK TRANSMITTER 14 MORE TICKS
2399	015410	104415	000014		ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2400	015414	104414			021264			:PORT4+LU-13
2401	015416	021264			BIT	#BITS,4(R1)		:RTS SHOULD BE CLEAR NOW
2402	015420	032761	000040	000004	BEG	4\$		:BR IF YES
2403	015426	001401			HLT	34		:ERROR, RTS NOT CLEAR
2404	015430	104034			4\$: SCOPE			:SCOPE THIS TEST
2405	015432	104400						
2406								
2407								
2408								
2409								
2410								
2411								
2412								
2413								
2414								
2415								
2416	015434	012737	000022	001226	TST22: MOV	#22,TSTNO		
2417	015442	012737	015744	001216	MOV	#TST23,NEXT		
2418								
2419	015450	104412			MSTCLR			:R1 CONTAINS BASE DMC11 ADDRESS
2420	015452	005061	000004		CLR	4(R1)		:MASTER CLEAR DMC1.
2421	015456	104414			ROMCLK			:CLEAR PORT4
2422	015460	122117			122117			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2423	015462	004737	033374		JSR	PC,CLRIO		:PUT LINE UNIT IN BITSTUFF MODE
2424	015466	012711	004000		MOV	#BIT11,(R1)		:DO THIS AFTER MODE IS SET
2425	015472	004737	032176		JSR	PC,OUTRDY		:SET LU LOOP
2426	015476	012761	000001	000004	MOV	#1,4(R1)		:WAIT FOR OUT-READY
2427	015504	104414			ROMCLK			:SET BIT0 IN PORT4
2428	015506	122111			122111			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2429	015510	104414			ROMCLK			:SET SOM!
2430	015512	122110			122110			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2431	015514	004537	033332		JSR	R5,MESLD		:LOAD GARBAGE CHAR
2432	015520	033640			STUFD	20,		:LOAD OUT SILO DATA
2433	015522	000024			20,			:MESSAGE ADDRESS
2434	015524	012704	033640		MOV	#STUFD,R4		:NUMBER OF CHARACTERS
2435	015530	005003			CLR	R3		:R4=CHARACTER POINTER
2436	015532	012700	000006		MOV	#6,R0		:R3= BIT COUNTER
2437	015536	104415	000002		DATACLK,	2		:BIT COUNTER FOR FLAG CHARACTER
2438	015542	012737	000176	001252	MOV	#1B<01111110>,TEMP3		:SET UP TRANSMITTER
2439	015550	104415	000001		64\$: DATACLK,	1		:PUT FLAG CHARACTER IN TEMP3
2440	015554	106037	001252		RORB	TEMP3		:CLOCK FLAG ONCE
2441	015560	103405			BCC	65\$		:SHIFT SOFT FLAG
2442	015562	004737	032012		JSR	PC,GETSI		:BR IF BIT IS MARK
2443	015566	103006			BCC	66\$		:LOOK AT BIT WINDOW
2444	015570	104026			HLT	26		:BR IF OK
2445	015572	000404			BR	66\$		:ERROR IN FLAG CHAR
2446	015574	004737	032012		65\$: JSR	PC,GETSI		:LOOK AT BIT WINDOW

***** TEST 22 *****
 *BIT STUFF TEST
 *THIS TEST CHECKS ZERO BIT STUFFING OF
 * THE TRANSMITTER IN THE BIT WINDOW

: TEST 22

 TST22: MOV #22,TSTNO
 MOV #TST23,NEXT

MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
 CLR 4(R1) ;MASTER CLEAR DMC1.
 ROMCLK ;CLEAR PORT4
 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
 MOV #BIT11,(R1) ;DO THIS AFTER MODE IS SET
 JSR PC,OUTRDY ;SET LU LOOP
 MOV #1,4(R1) ;WAIT FOR OUT-READY
 ROMCLK ;SET BIT0 IN PORT4
 122111 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ROMCLK ;SET SOM!
 122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 JSR R5,MESLD ;LOAD GARBAGE CHAR
 STUFD 20, ;LOAD OUT SILO DATA
 20, ;MESSAGE ADDRESS
 MOV #STUFD,R4 ;NUMBER OF CHARACTERS
 CLR R3 ;R4=CHARACTER POINTER
 MOV #6,R0 ;R3= BIT COUNTER
 DATACLK, 2 ;BIT COUNTER FOR FLAG CHARACTER
 MOV #1B<01111110>,TEMP3 ;SET UP TRANSMITTER
 64\$: DATACLK, 1 ;PUT FLAG CHARACTER IN TEMP3
 RORB TEMP3 ;CLOCK FLAG ONCE
 BCC 65\$;SHIFT SOFT FLAG
 JSR PC,GETSI ;BR IF BIT IS MARK
 BCC 66\$;LOOK AT BIT WINDOW
 HLT 26 ;BR IF OK
 BR 66\$;ERROR IN FLAG CHAR
 65\$: JSR PC,GETSI ;LOOK AT BIT WINDOW

2447	015600	103401			BCS	66\$		:BR IF OK
2448	015602	104026			HLT	26		:ERROR IN FLAG CHAR
2449	015604	005203			INC	R3		:INC BIT COUNT
2450	015606	022703	000010		CMP	#10,R3		:FLAG DONE YET?
2451	015612	001356			BNE	64\$		:BR IF NO
2452	015614	005003			CLR	R3		:CLEAR BIT COUNT
2453	015616	012700	000024		MOV	#20,R0		:R0=CHARACTER COUNTER
2454	015622	005037	033612		CLR	BITCON		:CLEAR BIT STUFF COUNTER
2455	015626	112405			MOVB	(R4)+,R5		:LOAD CHARACTER IN R5
2456	015630	010502			MOV	R5,R2		:LOAD CHARACTER IN R2
2457	015632	104415	000001		DATACLK,	1		:SHIFT DTAT ONCE
2458	015636	106002			RORB	R2		:SHIFT SOFT DATA
2459	015640	103407			BCS	5\$		:BR IF CARRY SET
2460	015642	005037	033612		CLR	BITCON		:CLEAR BIT STUFF COUNTER
2461	015646	004737	032012		JSR	PC,GETSI		:LOOK AT WINDOW
2462	015652	103010			BCC	6\$		:BR IF SPACE
2463	015654	104006			HLT	6		:ERROR, WINDOW WAS A MARK
2464	015656	000406			BR	6\$		:CONTINUE
2465	015660	005237	033612		INC	BITCON		:ADD 1 TO BIT STUFF COUNTER
2466	015664	004737	032012		JSR	PC,GETSI		:LOOK AT WINDOW
2467	015670	103401			BCS	6\$		:BR IF MARK
2468	015672	104006			HLT	6		:ERROR, WINDOW WAS A SPACE
2469	015674	022737	000005	033612	CMP	#5,BITCON		:HAVE THERE BEEN 5 'S IN A ROW
2470	015702	001010			BNE	7\$		:BR IF NO
2471	015704	005037	033612		CLR	BITCON		:IF YES CLR BIT STUFF COUNTER
2472	015710	104415	000001		DATACLK,	1		:AND CLOCK TRANSMITTER ONCE
2473	015714	004737	032012		JSR	PC,GETSI		:CHECK WINDOW FOR A ZERO STUFF!!
2474	015720	103001			BCC	7\$		:BR IF WINDOW IS A SPACE
2475	015722	104030			HLT	30		:ERROR, TRANSMITTER DID NOT STUFF A ZERO
2476	015724	005203			INC	R3		:BUMP BIT COUNTER
2477	015726	022703	000010		CMP	#10,R3		:DONE THIS CHARACTER YET?
2478	015732	001337			BNE	4\$		:BR IF NO
2479	015734	005003			CLR	R3		:RESTART BIT COUNTER AT ZERO
2480	015736	005300			DEC	R0		:DEC CHARACTER COUNTER
2481	015740	001332			BNE	3\$		:BR IF NOT DONE YET
2482	015742	104400			9\$:	SCOPE		:SCOPE THIS TEST

***** TEST 23 *****
 :BITSTUFF TRANSMITTER TEST
 :SINGLE CLOCK THE CHARACTER 377
 :CHECK FLAG AND DATA IN THE BIT WINDOW
 :VERIFY EACH BIT POSITION AS IT
 :PASSES THE BIT WINDOW (SI BIT)
 :ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
 :*****

: TEST 23

2483								
2484								
2485								
2486								
2487								
2488								
2489								
2490								
2491								
2492								
2493								
2494								
2495								
2496	015744	012737	000023	001226	TEST23:	MOV	#23,TSTNO	
2497	015752	012737	016252	001216		MOV	#TST24,NEXT	
2498								
2499	015760	104412				MSTCLR		:R1 CONTAINS BASE DMC11 ADDRESS
2500	015762	005061	000004			CLR	4(R1)	:MASTER CLEAR DMC11
2501	015766	104414				ROMCLK		:CLEAR PORT4
2502	015770	122117				122117		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
								:PUT LINE UNIT IN BITSTUFF MODE

2503	015772	004737	033374	JSR	PC,CLR10	;DO THIS AFTER MODE IS SET
2504	015776	005037	033612	CLR	BITCON	;CONSECUTIVE 1'S COUNTER INIT TO 0
2505	016002	012711	004000	MOV	#BIT11,(R1)	;SET LINE UNIT LOOP
2506	016006	004737	032176	JSR	PC,OUTRDY	;WAIT FOR OUT-READY
2507	016012	012761	000001 000004	MOV	#1,4(R1)	;SET BIT0 IN PORT4
2508	016020	104414		ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2509	016022	122111		122111		;SET COM!
2510	016024	104414		ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2511	016026	122110		122110		;LOAD GARBAGE CHAR
2512	016030	012705	000377	MOV	#377,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT	
2513	016034	010537	016206	MOV	R5,5\$	;LOAD CHAR FOR STUFF CHECK
2514	016040	004737	032176	JSR	PC,OUTRDY	;WAIT FOR OUT-READY
2515	016044	010561	000004	MOV	R5,4(R1)	;LOAD PORT4 WITH CHARACTER
2516	016050	104414		ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2517	016052	122110		122110		;LOAD OUT DATA
2518	016054	004737	032044	JSR	PC,OCOR	;WAIT FOR OCOR TO SET
2519	016060	005003		CLR	R3	;CLEAR BIT COUNTER
2520	016062	010502		MOV	R5,R2	;LOAD CHARACTER IN R2
2521	016064	104415	000002	DATACLK,	2	;2 TICKS TO SET UP TRANSMITTER
2522	016070	012737	000176 001252	MOV	#1B<01111110>,TEMP3	;PUT FLAG CHARACTER IN TEMP3
2523	016076	104415	000001	DATACLK,	1	;CLOCK FLAG ONCE
2524	016102	106037	001252	RORB	TEMP3	;SHIFT SOFT FLAG
2525	016106	103405		BCS	65\$	;BR IF BIT IS MARK
2526	016110	004737	032012	JSR	PC,GETSI	;LOOK AT BIT WINDOW
2527	016114	103006		BCC	66\$	;BR IF OK
2528	016116	104026		HLT	26	;ERROR IN FLAG CHAR
2529	016120	000404		BR	66\$	
2530	016122	004737	032012	JSR	PC,GETSI	;LOOK AT BIT WINDOW
2531	016126	103401		BCS	66\$	;BR IF OK
2532	016130	104026		HLT	26	;ERROR IN FLAG CHAR
2533	016132	005203		INC	R3	;INC BIT COUNT
2534	016134	022703	000010	CMP	#10,R3	;FLAG DONE YET?
2535	016140	001356		BNE	64\$	;BR IF NO
2536	016142	005003		CLR	R3	;CLEAR BIT COUNT
2537	016144	005037	033612	CLR	BITCON	;CLEAR STUFF COUNT
2538	016150	104415	000001	DATACLK,	1	;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2539	016154	106002		RORB	R2	;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2540	016156	103005		BCC	2\$	;BR IF CARRY CLEAR
2541	016160	004737	032012	JSR	PC,GETSI	;GET THE WINDOW
2542	016164	103406		BCS	3\$	;BR IF BIT IS A MARK
2543	016166	104006		HLT	6	;ERROR BIT WAS A SPACE
2544	016170	000404		BR	3\$	;CONTINUE WITH TEST
2545	016172	004737	032012	JSR	PC,GETSI	;GET THE WINDOW
2546	016176	103001		BCC	3\$	;BR IF BIT IS A SPACE
2547	016200	104006		HLT	6	;ERROR BIT WAS A MARK
2548	016202					
2549	016202	004537	033474	JSR	R5,STFFCK	;CHECK FOR BIT STUFF
2550	016206	000377		377		;DATA CHARACTER
2551	016210	000001		1		;SHIFT COUNT
2552	016212	010237	016206	MOV	R2,5\$	;LOAD CHAR FOR STUFF CHECK
2553	016216	005203		INC	R3	;NEXT BIT
2554	016220	022703	000010	CMP	#10,R3	;DONE YET?
2555	016224	001351		BNE	1\$	;BR IF NO
2556	016226	104415	000014	DATACLK,	14	;CLOCK TRANSMITTER 14 MORE TICKS
2557	016232	104414		ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2558	016234	021264		021264		;PORT4+LU-13

```

2559 016236 032761 000040 000004      BIT      #BIT5,4(R1)      ;RTS SHOULD BE CLEAR NOW
2560 016244 001401                      BEQ        45              ;BR IF YES
2561 016246 104034                      HLT        34              ;ERROR, RTS NOT CLEAR
2562 016250 104400                      45:      SCOPE          ;SCOPE THIS TEST
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576 016252 012737 000024 001226      ;***** TEST 24 *****
2577 016260 012737 016604 001216      ;*BITSTUFF TRANSMITTER TEST
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614

```

016266 104412
 016270 005061 000004
 016274 104414
 016276 122117
 016300 004737 033374
 016304 005037 033612
 016310 012711 004000
 016314 005003
 016316 005004
 016320 005005
 016322 004737 032176
 016326 012761 000001 000004
 016334 104414
 016336 122111
 016340 104414
 016342 122110
 016344 004737 032176
 016350 010461 000004
 016354 104414
 016356 122110
 016360 005204
 016362 004737 032176
 016366 010461 000004
 016372 104414
 016374 122110
 016376 004737 032044
 016402 104415 000002
 016406 012737 000176 001252
 016414 104415 000001
 016420 106037 001252
 016424 103405
 016426 004737 032012
 016432 103006
 016434 104026
 016436 000404
 016440 004737 032012

```

TST24:  MOV      #24,TSTNO
        MOV      #TST25,NEXT
        MSTCLR
        CLR      4(R1)
        ROMCLK   122117
        JSR      PC,CLRIC
        CLR      BITCON
        MOV      #BIT11,(R1)
        CLR      R3
        CLR      R4
        CLR      R5
        JSR      PC,OUTRDY
        MOV      #1,4(R1)
        ROMCLK   122111
        ROMCLK   122110
        JSR      PC,OUTRDY
        MOV      R4,4(R1)
        ROMCLK   122110
        INC      R4
        JSR      PC,OUTRDY
        MOV      R4,4(R1)
        ROMCLK   122110
        JSR      PC,OCOR
        DATACLK, 2
        MOV      #1B<01111110>,TEMP3
        DATACLK, 1
        RORB     TEMP3
        BCS      65$
        JSR      PC,GETSI
        BCC      66$
        HLT      26
        BR       66$
        JSR      PC,GETSI

```

;R1 CONTAINS BASE DMC11 ADDRESS
 ;MASTER CLEAR DMC11
 ;CLEAR PORT4
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;PUT LINE UNIT IN BITSTUFF MODE
 ;DO THIS AFTER MODE IS SET
 ;CONSECUTIVE 1'S COUNTER INIT TO 0
 ;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!
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;LOAD GARBAGE CHAR
 ;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
 ;PUT FLAG CHARACTER IN TEMP3
 ;CLOCK FLAG ONCE
 ;SHIFT SOFT FLAG
 ;BR IF BIT IS MARK
 ;LOOK AT BIT WINDOW
 ;BR IF OK
 ;ERROR IN FLAG CHAR
 ;LOOK AT BIT WINDOW

64\$:
 65\$:
 66\$:

BASIC TRANSMITTER TESTS

2615	016444	103401			BCS	66\$	:BR IF OK
2616	016446	104026			HLT	26	:ERROR IN FLAG CHAR
2617	016450	005203			INC	R3	:INC BIT COUNT
2618	016452	022703	000010		CMP	#10,R3	:FLAG DONE YET?
2619	016456	001356			BNE	64\$	:BR IF NO
2620	016460	005003			CLR	R3	:CLEAR BIT COUNT
2621	016462	005037	033612		CLR	BITCON	:CLEAR BIT STUFF COUNTER
2622	016466	005003			CLR	R3	:CLEAR BIT COUNTER
2623	016470	010502			MOV	R5,R2	:LOAD CHARACTER IN R2
2624	016472	010237	016534		MOV	R2,6\$	:LOAD CHAR FOR STUFF CHECK
2625	016476	104415	000001		1\$: DATACLK,	1	:SHIFT NEXT BIT IN THE WINDOW SI BIT,
2626	016502	106002			RORB	R2	:SHIFT NEXT SOFTWARE BIT IN TO CARRY
2627	016504	103005			BCC	2\$	:BR IF CARRY CLEAR
2628	016506	004737	032012		JSR	PC,GETSI	:GET THE WINDOW
2629	016512	103406			BCS	3\$	:BR IF BIT IS A MARK
2630	016514	104006			HLT	6	:ERROR BIT WAS A SPACE
2631	016516	000404			BR	3\$	:CONTINUE WITH TEST
2632	016520	004737	032012		2\$: JSR	PC,GETSI	:GET THE WINDOW
2633	016524	103001			BCC	3\$	:BR IF BIT IS A SPACE
2634	016526	104006			HLT	6	:ERROR BIT WAS A MARK
2635	016530				3\$: JSR	R5,STFFCK	:CHECK FOR BIT STUFF
2636	016530	004537	033474		6\$: C		:DATA CHARACTER
2637	016534	000000			1		:SHIFT COUNT
2638	016536	000001			MOV	R2,6\$	:LOAD CHAR FOR STUFF CHECK
2639	016540	010237	016534		INC	R3	:NEXT BIT
2640	016544	005203			CMP	#10,R3	:DONE YET?
2641	016546	022703	000010		BNE	1\$	:BR IF NO
2642	016552	001351			INC	R4	:NEXT CHARACTER
2643	016554	005204			JSR	PC,OUTRDY	:WAIT FOR OUT-READY
2644	016556	004737	032176		MOV	R4,4(R1)	:LOAD PORT4 WITH CHARACTER
2645	016562	010461	000004		ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2646	016566	104414			122110		:LOAD OUT DATA
2647	016570	122110			INC	R5	:NEXT CHARACTER
2648	016572	005205			CMP	#400,R5	:DONE YET?
2649	016574	022705	000400		BNE	4\$	:BR IF NO
2650	016600	001332			5\$: SCOPE		:SCOPE THIS TEST
2651	016602	104400					
2652							
2653							
2654							
2655							
2656							
2657							
2658							
2659							
2660							
2661							
2662							
2663							
2664	016604	012737	000025	001226	TS*25: MOV	#25,TSTNO	
2665	016612	012737	017072	001216	MOV	#TST26,NEXT	
2666							
2667	016620	104412			MSTCLR		:R1 CONTAINS BASE DMC11 ADDRESS
2668	016622	005061	000004		CLR	4(R1)	:MASTER CLEAR DMC11
2669	016626	104414			ROMCLK		:CLEAR PORT4
2670	016630	122117			122117		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
							:PUT LINE UNIT IN BITSTUFF MODE

***** TEST 25 *****
 : *MULTIPLE FLAG AND TRANSMITTER ABORT TEST
 : *LOAD SILO WITH 5 FLAGS AND A CHAR (000)
 : *VERIFY IN THE BIT WINDOW THAT THE FLAGS
 : *AND DATA ARE CORRECT AND FOLLOWED BY AN ABORT
 : *SEQUENCE (8 CONTIGUOUS 1'S)
 : *****

: TEST 25

 TS*25: MOV #25,TSTNO
 MOV #TST26,NEXT

:R1 CONTAINS BASE DMC11 ADDRESS
 :MASTER CLEAR DMC11
 :CLEAR PORT4
 :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 :PUT LINE UNIT IN BITSTUFF MODE

2671	016632	004737	033374		JSR	PC,CLR10	:DO THIS AFTER MODE IS SET
2672	016636	012711	004000		MOV	#811,(R1)	:SET LU LOOP
2673	016642	012700	000005		MOV	#5,R0	:FLAG COUNT
2674	016646	005003			CLR	R3	:CLEAR BIT COUNTER
2675	016650	004737	032176	15:	JSR	PC,OUTRDY	:WAIT FOR OUT-READY
2676	016654	012761	000001	000004	MOV	#1,4(R1)	:SET BIT0 IN PORT4
2677	016662	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2678	016664	122111			122111		:SET SOM!
2679	016666	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2680	016670	122110			122110		:LOAD GARBAGE CHAR
2681	016672	005300			DEC	R0	:DEC COUNT
2682	016674	001365			BNE	15	:LOAD ANOTHER
2683	016676	004737	032176		JSR	PC,OUTRDY	:WAIT FOR OUTRDY
2684	016702	005061	000004		CLR	4(R1)	:CLEAR PORT4
2685	016706	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2686	016710	122110			122110		:LOAD A ZERO
2687	016712	004737	032044		JSR	PC,OCOR	:WAIT
2688	016716	012700	000005		MOV	#5,R0	:R0 = FLAG COUNT
2689	016722	104415	000002		DATACLK,	2	:SET UP TRANSMITTER
2690	016726			25:			
2691	016726	012737	000176	001252	MOV	#B<01111110>,TEMP3	:PUT FLAG CHARACTER IN TEMP3
2692	016734	104415	000001		DATACLK,	1	:CLOCK FLAG ONCE
2693	016740	106037	001252	645:	RORB	TEMP3	:SHIFT SOFT FLAG
2694	016744	103405			BCS	655	:BR IF BIT IS MARK
2695	016746	004737	032012		JSR	PC,GETSI	:LOOK AT BIT WINDOW
2696	016752	103006			BCC	665	:BR IF OK
2697	016754	104026			HLT	26	:ERROR IN FLAG CHAR
2698	016756	000404			BR	665	
2699	016760	004737	032012	655:	JSR	PC,GETSI	:LOOK AT BIT WINDOW
2700	016764	103401			BCS	665	:BR IF OK
2701	016766	104026			HLT	26	:ERROR IN FLAG CHAR
2702	016770	005203		665:	INC	R3	:INC BIT COUNT
2703	016772	022703	000010		CMP	#10,R3	:FLAG DONE YET?
2704	016776	001356			BNE	645	:BR IF NO
2705	017000	005003			CLR	R3	:CLEAR BIT COUNT
2706	017002	005300			DEC	R0	:DEC COUNT
2707	017004	001350			BNE	25	:BR IF NOT DONE
2708	017006	005003			CLR	R3	:R3 = BIT COUNT
2709	017010	005005			CLR	R5	:R5 = "EXPECTED"
2710	017012	104415	000001	35:	DATACLK,	1	:CLOCK ONCE
2711	017016	004737	032012		JSR	PC,GETSI	:GO LOOK AT WINDOW
2712	017022	103001			BCC	45	:BR IF A SPACE
2713	017024	104006			HLT	6	:ERROR, A MARK WAS SEEN
2714	017026	005203		45:	INC	R3	:INC BIT COUNT
2715	017030	022703	000010		CMP	#10,R3	:DONE YET?
2716	017034	001366			BNE	35	:BR IF NO
2717	017036	005003			CLR	R3	:CLEAR BIT COUNT
2718	017040	012705	000377		MOV	#377,R5	:R5 = "EXPECTED"
2719	017044	104415	000001	55:	DATACLK,	1	:CLOCK ONCE
2720	017050	004737	032012		JSR	PC,GETSI	:LOOK AT WINDOW
2721	017054	103401			BCS	65	:BR IF A MARK
2722	017056	104033			HLT	33	:ERROR, A SPACE WAS SEEN
2723	017060	005203		65:	INC	R3	:INC BIT COUNT
2724	017062	022703	000010		CMP	#10,R3	:DONE YET?
2725	017066	001366			BNE	55	:BR IF NO
2726	017070	104400			SCOPE		:SCOPE THIS TEST

```

***** TEST 26 *****
*LEADING ZERCS TEST
*VERIFY THAT THE SETTING OF SOM AND EOM TOGETHER
*AND THEN SOM ALONE WILL GENERATE 16 LEADING ZEROS
*AND A FLAG,THE CHECK IS MADE USING THE BIT WINDOW
*****

```

```

; TEST 26
;-----

```

```

TST26: MOV #26,TSTNO
MOV #TST27,NEXT

```

```

MSTCLR
CLR 4(R1)
ROMCLK
122117
JSR PC,CLRIO
MOV #BIT11,(R1)
JSR PC,OUTRDY
MOV #3,4(R1)

```

```

ROMCLK
122111
ROMCLK
122110
MOV #1,4(R1)
ROMCLK
122111
ROMCLK
122110
ROMCLK
122110
JSR PC,OCOR
CLR RO
DATACLK,2
DATACLK,1
JSR PC,GETSI
BCC +4
HLT 41
INC RO
CMP #16,.RO
BNE 15
CLR R3
MOV #1B<01111110>,TEMP3
DATACLK,1
RORB TEMP3
BCS 65$
JSR PC,GETSI
BCC 66$
HLT 26
BR 66$
JSR PC,GETSI
BCS 66$
HLT 26
INC R3

```

```

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;CLEAR PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET TO BITSTUFF MODE
;DO THIS AFTER MODE IS SET
;SET LU LOOP
;WAIT FOR OUTRDY
;LOAD PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET SOM & EOM
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;GARBAGE CHARACTER
;LOAD PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;SET SOM
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;GARBAGE CHAR
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;GARBAGE CHAR
;WAIT FOR OCOR
;RO = BIT COUNT
;SET UP TRANSMITTER
;SINGLE CLOCK TRANSMITTER
;LOOK AT BITWINDOW
;ERROR WINDOW WAS A MARK
;16 ZEROS YET?
;BR IF NO
;R3 = BIT COUNT
;PUT FLAG CHARACTER IN TEMP3
;CLOCK FLAG ONCE
;SHIFT SOFT FLAG
;BR IF BIT IS MARK
;LOOK AT BIT WINDOW
;BR IF OK
;ERROR IN FLAG CHAR
;LOOK AT BIT WINDOW
;BR IF OK
;ERROR IN FLAG CHAR
;INC BIT COUNT

```

15:

64\$:

65\$:

66\$:

```

2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738 017072 012737 000026 001226
2739 017100 012737 017312 001216
2740
2741 017106 104412
2742 017110 005061 000004
2743 017114 104414
2744 017116 122117
2745 017120 004737 033374
2746 017124 012711 004000
2747 017130 004737 032176
2748 017134 012761 000003 000004
2749 017142 104414
2750 017144 122111
2751 017146 104414
2752 017150 122110
2753 017152 012761 000001 000004
2754 017160 104414
2755 017162 122111
2756 017164 104414
2757 017166 122110
2758 017170 104414
2759 017172 122110
2760 017174 004737 032044
2761 017200 005000
2762 017202 104415 000002
2763 017206 104415 000001
2764 017212 004737 032012
2765 017216 103001
2766 017220 104041
2767 017222 005200
2768 017224 022700 000020
2769 017230 001366
2770 017232 005003
2771 017234 012737 000176 001252
2772 017242 104415 000001
2773 017246 106037 001252
2774 017252 103405
2775 017254 004737 032012
2776 017260 103006
2777 017262 104026
2778 017264 000404
2779 017266 004737 032012
2780 017272 103401
2781 017274 104026
2782 017276 005203

```

2783 017300 022703 000010
 2784 017304 001356
 2785 017306 005003
 2786 017310 104400

CMP #10,R3 ;FLAG DONE YET?
 BNE #45 ;BR IF NO
 CLR R3 ;CLEAR BIT COUNT
 SCOPE ;SCOPE THIS TEST

***** TEST 27 *****
 ;BITSTUFF STRIP FLAG TEST
 ;SET LU LOOP, SINGLE STEP 5 FLAGS
 ;VERIFY THAT IN ACTIVE DOES NOT SET
 ;*****

TEST 27

2797 017312 012737 000027 001226
 2798 017320 012737 017414 001216
 2799
 2800 017326 104412
 2801 017330 005061 000004
 2802 017334 104414
 2803 017336 122117
 2804 017340 004737 033374
 2805 017344 012711 004000
 2806 017350 012702 000012
 2807 017354 004737 032062
 2808 017360 000005
 2809 017362 104415 000054
 2810 017366 104414
 2811 017370 021244
 2812 017372 016104 000004
 2813 017376 042704 000277
 2814 017402 005005
 2815 017404 120504
 2816 017406 001401
 2817 017410 104040
 2818 017412 104400

TST27: MOV #27,TSTNO
 MOV #TST30,NEXT

MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
 CLR 4(R1) ;MASTER CLEAR DMC11
 ROMCLK ;CLEAR PORT4
 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=53C4
 JSR PC,CLR10 ;PUT LINE UNIT IN BITSTUFF MODE
 MOV #BIT11,(R1) ;DO THIS AFTER MODE IS SET
 MOV #12,R2 ;SET LU LOOP
 JSR PC,SYNC ;SAVE LU REG FOR TYPEOUT
 5 ;SINGLE CLOCK 5 SYNC CHARACTERS
 DATACLK, 54
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=53C4
 021244 ;PORT4+LU12
 MOV 4(R1),R4 ;PUT "FOUND" IN R4
 BIC #277,R4 ;CLEAR UNWANTED BITS
 CLR R5 ;PUT "EXPECTED" IN R5
 CMPB R5,R4 ;IS ACTIVE CLEAR?
 BEQ 15 ;BR IF YES
 HLT 40 ;ERROR ACTIVE IS NOT CLEAR
 15: SCOPE ;SCOPE THIS TEST

***** TEST 30 *****
 ;BITSTUFF IN ACTIVE TEST
 ;SET LU LOOP, SINGLE STEP 5 FLAGS AND A NON-FLAG (301)
 ;VERIFY THAT IN ACTIVE IS SET
 ;*****

TEST 30

2829 017414 012737 000030 001226
 2830 017422 012737 017520 001216
 2831
 2832 017430 104412
 2833 017432 005061 000004
 2834 017436 104414
 2835 017440 122117
 2836 017442 004737 033374
 2837 017446 012711 004000
 2838 017452 012702 000012

TST30: MOV #30,TSTNO
 MOV #TST31,NEXT

MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
 CLR 4(R1) ;MASTER CLEAR DMC11
 ROMCLK ;CLEAR PORT4
 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=53C4
 JSR PC,CLR10 ;PUT LINE UNIT IN BITSTUFF MODE
 MOV #BIT11,(R1) ;DO THIS AFTER MODE IS SET
 MOV #12,R2 ;SET LU LOOP
 ;SAVE LU REG FOR TYPEOUT

```

2839 017456 004737 032062      JSR      PC, SYNC      ; SINGLE CLOCK 5 SYNC CHARACTERS
2840 017462 000005      5
2841 017464 104415 000064      DATA CLK,      64
2842 017470 104414      ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2843 017472 021244      021244      ; PORT4+LU12
2844 017474 016104 000004      MOV      4(R1), R4      ; PUT "FOUND" IN R4
2845 017500 042704 000277      BIC      #277, R4      ; CLEAR UNWANTED BITS
2846 017504 012705 000100      MOV      #BIT6, R5      ; PUT "EXPECTED" IN R5
2847 017510 120504      CMPB      R5, R4      ; IS ACTIVE SET?
2848 017512 001401      BEQ      1$      ; BR IF YES
2849 017514 104040      HLT      40      ; ERROR ACTIVE IS NOT SET
2850 017516 104400      1$: SCOPE      ; SCOPE THIS TEST
2851
2852
2853 ;***** TEST 31 *****
2854 ;*BITSTUFF IN ACTIVE TEST
2855 ;*SET LINE UNIT LOOP, SINGLE STEP ONE FLAG AND A CHAR (301)
2856 ;*VERIFY THAT IN ACTIVE IS SET
2857 ;*****
2858
2859 ; TEST 31
2860 ;-----
2861 017520 012737 000031 001226      TST31: MOV      #31, TSTNO
2862 017526 012737 017656 001216      MOV      #TST32, NEXT
2863
2864 017534 104412      MSTCLR      ; R1 CONTAINS BASE DMC11 ADDRESS
2865 017536 005061 000004      CLR      4(R1)      ; MASTER CLEAR DMC11
2866 017542 104414      ROMCLK      ; CLEAR PORT4
2867 017544 122117      122117      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2868 017546 004737 033374      JSR      PC, CLRIO      ; PUT LINE UNIT IN BITSTUFF MODE
2869 017552 012711 004000      MOV      #BIT11, (R1)      ; MUST DO THIS AFTER MODE IS SET
2870 017556 012702 000012      MOV      #12, R2      ; SAVE REG ADDRESS FOR TYPEOUT
2871 017562 004737 032176      JSR      PC, OUTRDY      ; WAIT FOR OUTRDY
2872 017566 012761 000001 000004      MOV      #1, 4(R1)      ; LOAD PORT4
2873 017574 104414      ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2874 017576 122111      122111      ; SET SOM
2875 017600 104414      ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2876 017602 122110      122110      ; LOAD GARBAGE CHAR
2877 017604 012761 000301 000004      MOV      #301, 4(R1)      ; LOAD PORT4
2878 017612 104414      ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2879 017614 122110      122110      ; LOAD OUT DATA
2880 017616 004737 032044      JSR      PC, OCOR      ; WAIT FOR OCOR
2881 017622 104415 000023      DATA CLK,      23      ; SINGLE CLOCK THE DATA
2882 017626 104414      ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2883 017630 021244      021244      ; PORT4+LU-12
2884 017632 016104 000004      MOV      4(R1), R4      ; PUT "FOUND" IN R4
2885 017636 042704 000277      BIC      #277, R4      ; CLEAR UNWANTED BITS
2886 017642 012705 000100      MOV      #BIT6, R5      ; PUT "EXPECTED" IN R5
2887 017646 120504      CMPB      R5, R4      ; IS IN ACTIVE SET?
2888 017650 001401      BEQ      1$      ;
2889 017652 104040      HLT      40      ; ERROR IN ACTIVE NOT SET
2890 017654 104400      1$: SCOPE      ; SCOPE THIS TEST
2891
2892
2893 ;***** TEST 32 *****
2894 ;*BITSTUFF IN ACTIVE TEST

```

```

2895      ;*SET LU LOOP, SINGLE STEP 2 FLAGS AND A NON-FLAG (301)
2896      ;*VERIFY THAT IN ACTIVE IS SET
2897      ;*****
2898
2899      : TEST 32
2900      :-----
2901      017656 012737 000032 001226      TST32: MOV      #32,TSTNO
2902      017664 012737 017762 001216      MOV      #TST33,NEXT
2903
2904      017672 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
2905      017674 005061 000004      CLR      4(R1)      ;MASTER CLEAR DMC11
2906      017700 104414      ROMCLK      ;CLEAR PORT4
2907      017702 122117      122117      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2908      017704 004737 033374      JSR      PC,CLR10      ;PUT LINE UNIT IN BITSTUFF MODE
2909      017710 012711 004000      MOV      #BIT11,(R1)      ;DO THIS AFTER MODE IS SET
2910      017714 012702 000012      MOV      #12,R2      ;SET LU LOOP
2911      017720 004737 032062      JSR      PC,SYNC      ;SAVE LU REG FOR TYPEOUT
2912      017724 000002      2          ;SINGLE CLOCK 2 SYNC CHARACTERS
2913      017726 104415 000033      DATACLK, 33
2914      017732 104414      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2915      017734 021244      021244      ;PORT4+LU12
2916      017736 016104 000004      MOV      4(R1),R4      ;PUT "FOUND" IN R4
2917      017742 042704 000277      BIC      #277,R4      ;CLEAR UNWANTED BITS
2918      017746 012705 000100      MOV      #BIT6,R5      ;PUT "EXPECTED" IN R5
2919      017752 120504      CMPB      R5,R4      ;IS ACTIVE SET?
2920      017754 001401      BEQ      1$      ;BR IF YES
2921      017756 104040      HLT      40      ;ERROR ACTIVE IS NOT SET
2922      017760 104400      1$: SCOPE      ;SCOPE THIS TEST
2923
2924
2925      ;***** TEST 33 *****
2926      ;*IN CLEAR TEST
2927      ;*SYNC UP RECEIVER AND TRANSMIT A CHARACTER
2928      ;*WAIT FOR IN RDY THEN SET IN CLEAR
2929      ;*VERIFY THAT IN ACTIVE AND IN RDY ARE CLEARED
2930      ;*****
2931
2932      : TEST 33
2933      :-----
2934      017762 012737 000033 001226      TST33: MOV      #33,TSTNO
2935      017770 012737 020166 001216      MOV      #TST34,NEXT
2936
2937      017776 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
2938      020000 005061 000004      CLR      4(R1)      ;MASTER CLEAR DMC11
2939      020004 104414      ROMCLK      ;CLEAR PORT4
2940      020006 122117      122117      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2941      020010 004737 033374      JSR      PC,CLR10      ;PUT LINE UNIT IN BITSTUFF MODE
2942      020014 012702 000012      MOV      #12,R2      ;DO THIS AFTER MODE IS SET
2943      020020 012711 004000      MOV      #BIT11,(R1)      ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
2944      020024 012761 000001 000004      MOV      #1,4(R1)      ;SET LINE UNIT LOOP
2945      020032 104414      ROMCLK      ;SET BIT0 IN PORT4
2946      020034 122111      122111      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2947      020036 104414      ROMCLK      ;SET SOM!
2948      020040 122110      122110      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2949      020042 004737 032342      JSR      PC,CHARSC      ;LOAD GARBAGE CHAR
2950      020046 000026      26          ;LOAD SILO WITH CHARACTER
2951      ;CHARACTER

```

2951	020050	104415	000033		DATA CLK,	33		; SINGLE CLOCK THE DATA
2952	020054	104416	000002		TIMER,	2		; WAIT FOR INRDY
2953	020060	104414			ROMCLK			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2954	020062	021244			021244			; PORT4+LU 12
2955	020064	016104	000004		MOV	4(R1), R4		; PUT "FOUND" IN R4
2956	020070	042704	000357		BIC	#357, R4		; CLEAR UNWANTED BITS
2957	020074	012705	000020		MOV	#BIT4, R5		; PUT "EXPECTED" IN R5
2958	020100	120504			CMPB	R5, R4		; IS INRDY SET?
2959	020102	001401			BEQ	1\$		
2960	020104	104040			HLT	40		; ERROR, INRDY IS NOT SET
2961	020106			1\$:				
2962	020106	012761	000200	000004	MOV	#BIT7 4(R1)		; LOAD PORT4
2963	020114	104414			ROMCLK			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2964	020116	122112			122112			; SET IN CLEAR
2965	020120	104414			ROMCLK			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2966	020122	021244			021244			; PORT4+LU 12
2967	020124	016104	000004		MOV	4(R1), R4		; PUT "FOUND" IN R4
2968	020130	042704	000277		BIC	#277, R4		; CLEAR UNWANTED BITS
2969	020134	005005			CLR	R5		; PUT "EXPECTED" IN R5
2970	020136	120504			CMPB	R5, R4		; IS IN ACTIVE CLEAR?
2971	020140	001401			BEQ	2\$		
2972	020142	104040			HLT	40		; ERROR, IN ACTIVE IS NOT CLEAR
2973	020144			2\$:				
2974	020144	016104	000004		MOV	4(R1), R4		; PUT "FOUND" IN R4
2975	020150	042704	000357		BIC	#357, R4		; CLEAR UNWANTED BITS
2976	020154	005005			CLR	R5		; PUT "EXPECTED" IN R5
2977	020156	120504			CMPB	R5, R4		; IS INRDY CLEARED?
2978	020160	001401			BEQ	3\$		
2979	020162	104040			HLT	40		; ERROR, INRDY IS NOT CLEARED
2980	020164	104400		3\$:	SCOPE			; SCOPE THIS TEST
2981								
2982								
2983								
2984								
2985								
2986								
2987								
2988								
2989								
2990								
2991	020166	012737	000034	001226	TST34:	MOV	#34, TSTNO	
2992	020174	012737	020334	001216		MOV	#TST35, NEXT	
2993								
2994	020202	104412			MSICLR			; R1 CONTAINS BASE DMC11 ADDRESS
2995	020204	005061	000004		CLR	4(R1)		; MASTER CLEAR DMC11
2996	020210	104414			ROMCLK			; CLEAR PORT4
2997	020212	122117			122117			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2998	020214	004737	033374		JSR	PC, CLRI0		; PUT LINE UNIT IN BITSTUFF MODE
2999	020220	012702	000012		MOV	#12, R2		; DO THIS AFTER MODE IS SET
3000	020224	012711	004000		MOV	#BIT11, (R1)		; SAVE REG ADDRESS IN R2 FOR TYPEOUT
3001	020230	012761	000001	000004	MOV	#1, 4(R1)		; SET LINE UNIT LOOP
3002	020236	104414			ROMCLK			; SET BIT0 IN PORT4
3003	020240	122111			122111			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3004	020242	104414			ROMCLK			; SET SOM!
3005	020244	122110			122110			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3006	020246	004737	032342		JSR	PC, CHARSD		; LOAD GARBAGE CHAR
								; LOAD SILO WITH CHARACTER

```

;***** TEST 34 *****
;BITSTUFF BASIC RECEICER TEST
;SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 0
;VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
;*****

```

```

: TEST 34
:-----

```

PC	Instruction	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418
----	-------------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

3063	020456	104040			15:	HLT	40		;ERROR, INRDY IS NOT SET
3064	020460					ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3065	020460	104414				021204			;PORT4-IN DATA
3066	020462	021204				MOV	4(R1),R4		;PUT "FOUND" IN R4
3067	020464	016104	000004			MOV	#125,R5		;PUT "EXPECTED" IN R5
3068	020470	012705	000125			CMPB	R5,R4		;WAS A 125 RECEIVED
3069	020474	120504				BEQ	25		
3070	020476	001401				HLT	10		;ERROR, RECEIVED DATA IS WRONG
3071	020500	104010			25:	SCOPE			;SCOPE THIS TEST
3072	020502	104400							
3073									
3074									
3075									
3076									
3077									
3078									
3079									
3080									
3081									
3082									
3083	020504	012737	000036	001226	TS36:	MOV	#36,TSTNO		
3084	020512	012737	020654	001216		MOV	#TST37,NEXT		
3085									
3086	020520	104412				MSTCLR			;R1 CONTAINS BASE DMC11 ADDRESS
3087	020522	005061	000004			CLR	4(R1)		;MASTER CLEAR DMC11
3088	020526	104414				ROMCLK			;CLEAR PORT4
3089	020530	122117				122117			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3090	020532	004737	033374			JSR	PC,CLRIO		;PUT LINE UNIT IN BITSTUFF MODE
3091	020536	012702	000012			MOV	#12,R2		;DO THIS AFTER MODE IS SET
3092	020542	012711	004000			MOV	#BIT11,(R1)		;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3093	020546	012761	000001	000004		MOV	#1,4(R1)		;SET LINE UNIT LOOP
3094	020554	104414				ROMCLK			;SET BIT0 IN PORT4
3095	020556	122111				122111			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3096	020560	104414				ROMCLK			;SET SOM!
3097	020562	122110				122110			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3098	020564	004737	032342			JSR	PC,CHARSD		;LOAD GARBAGE CHAR
3099	020570	000252				252			;LOAD SILO WITH CHARACTER
3100	020572	104415	000033			DATACLK,	33		;CHARACTER
3101	020576	104416	000002			TIMER,	2		;SINGLE CLOCK THE DATA
3102	020602	104414				ROMCLK			;WAIT FOR INRDY
3103	020604	021244				021244			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3104	020606	016104	000004			MOV	4(R1),R4		;PORT4+LU 12
3105	020612	042704	000357			BIC	#357,R4		;PUT "FOUND" IN R4
3106	020616	012705	000020			MOV	#BIT4,R5		;CLEAR UNWANTED BITS
3107	020622	120504				CMPB	R5,R4		;PUT "EXPECTED" IN R5
3108	020624	001401				BEQ	15		;IS INRDY SET?
3109	020626	104040				HLT	40		;ERROR, INRDY IS NOT SET
3110	020630				15:				
3111	020630	104414				ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3112	020632	021204				021204			;PORT4-IN DATA
3113	020634	016104	000004			MOV	4(R1),R4		;PUT "FOUND" IN R4
3114	020640	012705	000252			MOV	#252,R5		;PUT "EXPECTED" IN R5
3115	020644	120504				CMPB	R5,R4		;WAS A 252 RECEIVED
3116	020646	001401				BEQ	25		
3117	020650	104010				HLT	10		;ERROR, RECEIVED DATA IS WRONG
3118	020652	104400			25:	SCOPE			;SCOPE THIS TEST

***** TEST 36 *****
 ;BITSTUFF BASIC RECEIVER TEST
 ;SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 252
 ;VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
 ;*****

TEST 36

 TS36: MOV #36,TSTNO
 MOV #TST37,NEXT

;R1 CONTAINS BASE DMC11 ADDRESS
 ;MASTER CLEAR DMC11
 ;CLEAR PORT4
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;PUT LINE UNIT IN BITSTUFF MODE
 ;DO THIS AFTER MODE IS SET
 ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
 ;SET LINE UNIT LOOP
 ;SET BIT0 IN PORT4
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;SET SOM!
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;LOAD GARBAGE CHAR
 ;LOAD SILO WITH CHARACTER
 ;CHARACTER
 ;SINGLE CLOCK THE DATA
 ;WAIT FOR INRDY
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;PORT4+LU 12
 ;PUT "FOUND" IN R4
 ;CLEAR UNWANTED BITS
 ;PUT "EXPECTED" IN R5
 ;IS INRDY SET?

;ERROR, INRDY IS NOT SET

;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;PORT4-IN DATA
 ;PUT "FOUND" IN R4
 ;PUT "EXPECTED" IN R5
 ;WAS A 252 RECEIVED?

;ERROR, RECEIVED DATA IS WRONG
 ;SCOPE THIS TEST

```

3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129 020654 012737 000037 001226
3130 020662 012737 021024 001216
3131
3132 020670 104412
3133 020672 005061 000004
3134 020676 104414
3135 020700 122117
3136 020702 004737 033374
3137 020706 012702 000012
3138 020712 012711 004000
3139 020716 012761 000001 000004
3140 020724 104414
3141 020726 122111
3142 020730 104414
3143 020732 122110
3144 020734 004737 032342
3145 020740 000377
3146 020742 104415 000034
3147 020746 104416 000002
3148 020752 104414
3149 020754 021244
3150 020756 016104 000004
3151 020762 042704 000357
3152 020766 012705 000020
3153 020772 120504
3154 020774 001401
3155 020776 104040
3156 021000
3157 021000 104414
3158 021002 021204
3159 021004 016104 000004
3160 021010 012705 000377
3161 021014 120504
3162 021016 001401
3163 021020 104010
3164 021022 104400
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174

:***** TEST 37 *****
:BITSTUFF BASIC RECEICER TEST
:SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 377
:VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
:*****

: TEST 37
:-----
TST37: MOV #37,TSTNO
MOV #TST40,NEXT

MSTCLR
CLR 4(R1)
ROMCLK
122117
JSR PC,CLR10
MOV #12,R2
MOV #BIT11,(R1)
MOV #1,4(R1)
ROMCLK
122111
ROMCLK
122110
JSR PC,CHARSC
377
DATACLK, 34
TIMER, 2
ROMCLK
021244
MOV 4(R1),R4
BIC #357,R4
MOV #BIT4,R5
CMPB R5,R4
BEQ 1$
HLT 40

1$: ROMCLK
021204
MOV 4(R1),R4
MOV #377,R5
CMPB R5,R4
BEQ 2$
HLT 10

2$: SCOPE

:R1 CONTAINS BASE DMC11 ADDRESS
:MASTER CLEAR DMC11
:CLEAR PORT4
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:PUT LINE UNIT IN BITSTUFF MODE
:DO THIS AFTER MODE IS SET
:SAVE REG ADDRESS IN R2 FOR TYPEOUT
:SET LINE UNIT LOOP
:SET BIT0 IN PORT4
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:SET SOM!
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:LOAD GARBAGE CHAR
:LOAD SILO WITH CHARACTER
:CHARACTER
:SINGLE CLOCK THE DATA
:WAIT FOR INRDY
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:PORT4+LU 12
:PUT "FOUND" IN R4
:CLEAR UNWANTED BITS
:PUT "EXPECTED" IN R5
:IS INRDY SET?
:ERROR, INRDY IS NOT SET
:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
:PORT4+IN DATA
:PUT "FOUND" IN R4
:PUT "EXPECTED" IN R5
:WAS A 377 RECEIVED?
:ERROR, RECEIVED DATA IS WRONG
:SCOPE THIS TEST

:***** TEST 40 *****
:BITSTUFF DATA TEST
:THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
:CHECKING EACH CHARACTER AS IT IS RECEIVED
:*****

: TEST 40
:-----

```

3175	021024	012737	000040	001226	TST40:	MOV	#40,TSTNO	
3176	021032	012737	021200	001216		MOV	#TST41,NEXT	
3177								
3178	021040	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
3179	021042	005061	000004			CLR	4(R1)	;MASTER CLEAR DMC11
3180	021046	104414				ROMCLK		;CLEAR PORT4
3181	021050	122117				122117		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3182	021052	004737	033374			JSR	PC,CLRIO	;PUT LINE UNIT IN BITSTUFF MODE
3183	021056	005037	032646			CLR	SCHAR	;DO THIS AFTER MODE IS SET
3184	021062	005137	032646			COM	SCHAR	;START BINARY COUNT AT ZERO
3185	021066	005037	033612			CLR	BITCON	;IF BITSTUFF SCHAR IS MINUS NUMBER
3186	021072	005037	032650			CLR	STUFLG	;START 1'S COUNT AT 0
3187	021076	005002				CLR	R2	;CLEAR BITSTUFF FLAG
3188	021100	012703	000073			MOV	#73,R3	;R2 IS "EXPECTED" DATA
3189	021104	012711	004000			MOV	#BIT11,(R1)	;R3 IS CHARACTER COUNT
3190	021110	004737	032406			JSR	PC,SILOLD	;SET LINE UNIT LOOP
3191	021114	104415	000023			DATACLK,	23	;LOAD SILO WITH COUNT PATTERN
3192	021120	104415	000730		13:	DATACLK,	730	;SYNC RECEIVER AND GET IT ACTIVE
3193	021124	004737	032652		45:	JSR	PC,INRDY	;CLOCK IN 73 CHARACTERS
3194	021130	104414				ROMCLK		;WAIT FOR INRDY
3195	021132	021204				021204		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3196	021134	016104	000004			MOV	4(R1),R4	;PORT4-IN DATA
3197	021140	010205				MOV	R2,R5	;PUT "FOUND" IN R4
3198	021142	120504				CMPB	R5,R4	;PUT "EXPECTED" IN R5
3199	021144	001401				BEQ	25	;IS DATA CORRECT?
3200	021146	104010				HLT	10	;BR IF YES
3201	021150	005202			25:	INC	R2	;DATA ERROR
3202	021152	022702	000400			CMP	#400,R2	;NEXT CHARACTER
3203	021156	001407				BEQ	35	;ALL DONE?
3204	021160	005303				DEC	R3	;BR IF YES
3205	021162	001360				BNE	45	;DECREMENT CHARACTER COUNT
3206	021164	004737	032406			JSR	PC,SILOLD	;BR IF SILO NOT EMPTY
3207	021170	012703	000073			MOV	#73,R3	;LOAD SILO WITH MORE OF COUNT PATTERN
3208	021174	000751				BR	15	;RELOAD CHARACTER COUNT
3209	021176	104400			35:	SCOPE		;CONTINUE
3210								;SCOPE THIS TEST
3211								
3212								
3213								
3214								
3215								
3216								
3217								
3218								
3219								
3220								
3221								
3222	021200	012737	000041	001226	TST41:	MOV	#41,TSTNO	
3223	021206	012737	021364	001216		MOV	#TST42,NEXT	
3224								
3225	021214	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
3226	021216	005061	000004			CLR	4(R1)	;MASTER CLEAR DMC11
3227	021222	104414				ROMCLK		;CLEAR PORT4
3228	021224	122117				122117		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3229	021226	004737	033374			JSR	PC,CLRIO	;PUT LINE UNIT IN BITSTUFF MODE
3230	021232	005037	032646			CLR	SCHAR	;DO THIS AFTER MODE IS SET
								;START BINARY COUNT AT ZERO

***** TEST 41 *****
 ;*BITSTUFF 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 41

3222	021200	012737	000041	001226	TST41:	MOV	#41,TSTNO	
3223	021206	012737	021364	001216		MOV	#TST42,NEXT	
3224								
3225	021214	104412				MSTCLR		;R1 CONTAINS BASE DMC11 ADDRESS
3226	021216	005061	000004			CLR	4(R1)	;MASTER CLEAR DMC11
3227	021222	104414				ROMCLK		;CLEAR PORT4
3228	021224	122117				122117		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3229	021226	004737	033374			JSR	PC,CLRIO	;PUT LINE UNIT IN BITSTUFF MODE
3230	021232	005037	032646			CLR	SCHAR	;DO THIS AFTER MODE IS SET
								;START BINARY COUNT AT ZERO

3231	021236	005137	032646		COM	SCHAR	; IF BITSTUFF SCHAR IS MINUS NUMBER
3232	021242	005037	033612		CLR	BITCON	; START 1'S COUNT AT 0
3233	021246	005037	032650		CLR	STUFLG	; CLEAR BITSTUFF FLAG
3234	021252	005002			CLR	R2	; R2 IS "EXPECTED" DATA
3235	021254	012703	000073		MOV	#73,R3	; R3 IS CHARACTER COUNT
3236	021260	005011			CLR	(R1)	; CLEAR LU LOOP IN MAINT REG
3237	021262	012761	000040	000004	MOV	#BIT5,4(R1)	; LOAD PORT4
3238	021270	104414			ROMCLK		; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3239	021272	122112			122112		; SET LU LOOP IN LU REG 12
3240	021274	004737	032406		JSR	PC,SILOLD	; LOAD SILO WITH COUNT PATTERN
3241	021300	104415	000023		DATACLK,	23	; SYNC RECEIVER AND GET IT ACTIVE
3242	021304	104415	000730		DATACLK,	730	; CLOCK IN 73 CHARACTERS
3243	021310	004737	032652		JSR	PC,INRDY	; WAIT FOR INRDY
3244	021314	104414			ROMCLK		; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3245	021316	021204			021204		; PORT4+IN DATA
3246	021320	016104	000004		MOV	4(R1),R4	; PUT "FOUND" IN R4
3247	021324	010205			MOV	R2,R5	; PUT "EXPECTED" IN R5
3248	021326	120504			CMPB	R5,R4	; IS DATA CORRECT?
3249	021330	001401			BEQ	2\$	; BR IF YES
3250	021332	104010			HLT	10	; DATA ERROR
3251	021334	005202			INC	R2	; NEXT CHARACTER
3252	021336	022702	000400		CMP	#400,R2	; ALL DONE?
3253	021342	001407			BEQ	3\$	; BR IF YES
3254	021344	005303			DEC	R3	; DECREMENT CHARACTER COUNT
3255	021346	001360			BNE	4\$	; BR IF SILO NOT EMPTY
3256	021350	004737	032406		JSR	PC,SILOLD	; LOAD SILO WITH MORE OF COUNT PATTERN
3257	021354	012703	000073		MOV	#73,R3	; RELOAD CHARACTER COUNT
3258	021360	000751			BR	1\$	; CONTINUE
3259	021362	104400			3\$:	SCOPE	; SCOPE THIS TEST

3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286

***** TEST 42 *****
 ;RECEIVER ABORT TEST
 ;SINGLE CLOCK 3 FLAGS, A 301, ANOTHER 301 AND 10 EXTRA
 ;CLOCK TICKS, VERIFY THAT A 301 AND A BLOCK END
 ;WERE RECEIVED INDICATING THAT THE RECEIVER RECOGNIZED
 ;THE ABORT SEQUENCE (8 CONTIGUOUS 1'S)
 ;*****

; TEST 42

3272	021364	012737	000042	001226	TS*42:	MOV	#42,TSTNO	
3273	021372	012737	021526	001216		MOV	#TST43,NEXT	
3274								
3275	021400	104412				MSTCLR		; R1 CONTAINS BASE DMC11 ADDRESS
3276	021402	005061	000004			CLR	4(R1)	; MASTER CLEAR DMC11
3277	021406	104414				ROMCLK		; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3278	021410	122117				122117		; PUT LINE UNIT IN BITSTUFF MODE
3279	021412	004737	033374			JSR	PC,CLR10	; DO THIS AFTER MODE IS SET
3280	021416	012711	004000			MOV	#BIT11,(R1)	; SET LINE UNIT LOOP
3281	021422	004737	032230			JSR	PC,CHAR	; LOAD SILO WITH 3 FLAGS
3282	021426	000301				301		; AND A 301
3283	021430	004737	032176			JSR	PC,OUTRDY	; WAIT FOR OUTRDY
3284	021434	104414				ROMCLK		; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3285	021436	122110				122110		; LOAD 2ND 301 CHARACTER
3286	021440	104415	000073			DATACLK,	73	; CLOCK THE 301 IN AND 10 EXTRA TICKS

[illegible]

```

3343 021652 016104 000004      MOV      4(R1),R4      ;PUT "FOUND" IN R4
3344 021656 042704 000223      BIC      #223,R4      ;CLEAR UNWANTED BITS
3345 021662 005005              CLR      R5              ;PUT "EXPECTED" IN R5
3346 021664 032737 020000 001366 BIT      #BIT13,STAT1 ;IS LINE UNIT M8202?
3347 021672 001402              BEQ      .+6              ;BR IF NO
3348 021674 052705 000010      BIS      #BIT3,R5      ;MRDY SET ON M8202
3349 021700 120504              CMPB     R5,R4      ;IS MRDY CLEAR?
3350 021702 001401              BEQ      2$              ;
3351 021704 104011              HLT      11              ;ERROR, MRDY NOT CLEAR
3352 021706 104400              2$: SCOPE      ;SCOPE THIS TEST
3353
3354
3355 ;***** TEST 44 *****
3356 ;*CABLE TURNAROUND TEST
3357 ;*CLEAR LINE UNIT LOOP, LOAD OUT DATA SILO
3358 ;*VERIFY THAT ALL MODEM SIGNALS ARE SET
3359 ;*****
3360
3361 ; TEST 44
3362 ;-----
3363 021710 012737 000044 001226 TST44: MOV      #44,TSTNO
3364 021716 012737 022054 001216      MOV      #TST45,NEXT
3365
3366 021724 104412      MSTCLR     ;R1 CONTAINS BASE DMC11 ADDRESS
3367 021726 032737 020000 001366 BIT      #BIT13,STAT1 ;MASTER CLEAR DMC11
3368 021734 001004      BNE      .+12      ;IS LINE UNIT M8202?
3369 021736 032737 040000 001366 BIT      #BIT14,STAT1 ;BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
3370 021744 001442      BEQ      1$      ;IS TURNAROUND CONNECTOR ON?
3371 021746 012711 004000      MOV      #BIT11,(R1) ;SKIP TEST IF NO
3372 021752 012761 000100 000004 MOV      #100, 4(R1) ;SET LINE UNIT LOOP
3373 021760 104414      ROMCLK     ;LOAD PORT4
3374 021762 122113      122113      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3375 021764 104416 000002      TIMER, 2 ;CLEAR ALL MODEM SIGNALS, EXCEPT STR
3376 021770 012761 000001 000004 MOV      #1,4(R1) ;WAIT
3377 021776 104414      ROMCLK     ;LOAD PORT4
3378 022000 122111      122111      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3379 022002 004537 033332      JSR      R5,MESLD ;SET SOM
3380 022006 033614      MESDAT     ;FILL OUT DATA SILO
3381 022010 000100      64.         ;WITH 64 CHARACTERS
3382 022012 012700 000050      MOV      #50,R0 ;PREPARE FOR DELAY
3383 022016 005011      CLR      (R1) ;CLEAR LINE UNIT LOOP
3384 022020
3385 022020 104414      ROMCLK     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3386 022022 021264      021264      ;PORT4-LU13
3387 022024 016104 000004      MOV      4(R1),R4 ;PUT "FOUND" IN R4
3388 022030 042704 000223      BIC      #223,R4 ;CLEAR UNWANTED BITS
3389 022034 012705 000154      MOV      #154,R5 ;PUT "EXPECTED" IN R5
3390 022040 120504      CMPB     R5,R4 ;COMPARE EXPECTED AND FOUND
3391 022042 001403      BEQ      1$      ;BR IF OK
3392 022044 005300      DEC      R0 ;DEC DELAY COUNT
3393 022046 001364      BNE      2$      ;BR IF NOT ZERO
3394 022050 104011      HLT      11 ;ERROR, ALL SIGNALS ARE NOT SET
3395 022052 104400      1$: SCOPE ;SCOPE THIS TEST
3396
3397
3398 ;***** TEST 45 *****

```

```

3399 ;*TEST OF CRC OPERATION
3400 ;*USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3401 ;*0, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3402 ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3403 ;*****
3404
3405 ; TEST 45
3406 -----
3407 022054 012737 000045 001226 TST45: MOV #45,TSTNO
3408 022062 012737 022420 001216 MOV #TST46,NEXT
3409 022070 012737 022124 001220 MOV #64$,LOCK
3410 ;R1 CONTAINS BASE DMC11 ADDRESS
3411 022076 104412 MSTCLR ;MASTER CLEAR DMC11
3412 022100 005061 000004 CLR 4(R1) ;CLEAR PORT4
3413 022104 104414 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3414 022106 122117 122117 ;PUT LINE UNIT IN BITSTUFF MODE
3415 022110 004737 033374 JSR PC,CLRIO ;DO THIS AFTER MODE IS SET
3416 022114 005037 033612 CLR BITCON ;CONSECUTIVE 1'S COUNTER INIT TO 0
3417 022120 012711 004000 MOV #BIT11,(R1) ;SET LU LOOP
3418 022124 004737 033374 64$: JSR PC,CLRIO ;CLEAR BCC REGISTERS
3419 022130 005000 CLR RO ;START SHIFT COUNTER AT ZERO
3420 022132 012737 102010 033030 MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
3421 022140 012737 000000 022204 MOV #0,66$ ;LOAD CHAR FOR SOFTWARE BCC
3422 022146 005037 022206 CLR 67$ ;CLEAR OLD SOFTWARE BCC
3423 022152 005137 022206 COM 67$ ;START AT -1
3424 022156 004737 033034 JSR PC,BCCLD ;LOAD OUT SILO WITH 2 SYNC
3425 022162 000000 0 ;AND THE CHARACTER 0
3426 022164 104415 000021 DATACLK, 21 ;GET TRANSMITTER ACTIVE
3427 022170 104415 000001 65$: DATACLK, 1 ;SHIFT BCC ONCE
3428 022174 005200 INC RO ;BUMP SHIFT COUNT
3429 022176 004537 032706 JSR RS,SIMBCC ;CALCULATE SOFTWARE BCC LSB
3430 022202 000001 1 ;ONE SHIFT
3431 022204 000000 66$: 0 ;DATA CHARACTER
3432 022206 000000 67$: 0 ;OLD BCC
3433 022210 103405 BCS 68$ ;BR IF SOFT BCC LSB IS SET
3434 022212 004737 033146 JSR PC,GETQ0 ;GET HARDWARE TRANSMITTER BCC LSB
3435 022216 103006 BCC 69$ ;BR IF HARD BCC LSB IS CLEAR
3436 022220 104012 HLT 12 ;ERROR, BCC LSB IS SET
3437 022222 000404 BR 69$ ;CONTINUE
3438 022224 004737 033146 68$: JSR PC,GETQ0 ;GET HARDWARE TRANSMITTER BCC LSB
3439 022230 103401 BCS 69$ ;BR IF HARD BCC LSB IS SET
3440 022232 104016 HLT 16 ;ERROR, HARD BCC LSB IS CLEAR
3441 022234 69$:
3442 022234 006037 022204 ROR 66$ ;SHIFT SOFT DATA
3443 022240 013737 033032 022206 MOV CALBCC,67$ ;LOAD OLD SOFT BCC
3444 022246 022700 000010 CMP #10,RO ;DONE YET?
3445 022252 001346 BNE 65$ ;BR IF NOT DONE
3446 022254 104401 SCOPE1 ;SCOPE SUBTEST (SW09=1)
3447 022256 012737 022264 001221 MOV #71$,LOCK ;NEW SCOPE1
3448 022264 004737 033374 71$: JSR PC,CLRIO ;CLEAR BCC REGISTERS
3449 022270 005000 CLR RO ;START SHIFT COUNTER AT ZERO
3450 022272 012737 102010 033030 MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
3451 022300 012737 000000 022344 MOV #0,73$ ;LOAD CHAR FOR SOFTWARE BCC
3452 022306 005037 022346 CLR 74$ ;CLEAR OLD SOFTWARE BCC
3453 022312 005137 022346 COM 74$ ;START AT -1
3454 022316 004737 033034 JSR PC,BCCLD ;LOAD OUT SILO WITH 2 SYNC

```

3455	022322	000000			0		:AND THE CHARACTER 0
3456	022324	104415	000032		DATACLK,	32	:GET RECEIVER ACTIVE
3457	022330	104415	000001		DATACLK,	1	:SHIFT BCC ONCE
3458	022334	005200		72\$:	INC	RO	:BUMP SHIFT COUNT
3459	022336	004537	032706		JSR	RS,SIMBCC	:CALCULATE SOFTWARE BCC LSB
3460	022342	000001			1		:ONE SHIFT
3461	022344	000000		73\$:	0		:DATA CHARACTER
3462	022346	000000		74\$:	0		:OLD BCC
3463	022350	103405			BCS	75\$	:BR IF SOFT BCC LSB IS SET
3464	022352	004737	033160		JSR	PC,GETQI	:GET HARDWARE RECEIVER BCC LSB
3465	022356	103006			BCC	76\$	:BR IF HARD BCC LSB IS CLEAR
3466	022360	104013			HLT	13	:ERROR, BCC LSB IS SET
3467	022362	000404			BR	76\$	:CONTINUE
3468	022364	004737	033160	75\$:	JSR	PC,GETQI	:GET HARDWARE RECEIVER BCC LSB
3469	022370	103401			BCS	76\$	:BR IF HARD BCC LSB IS SET
3470	022372	104017			HLT	17	:ERROR, BCC LSB IS CLEAR
3471	022374			76\$:			
3472	022374	006037	022344		ROR	73\$	:SHIFT SOFT DATA
3473	022400	013737	033032	022346	MOV	CALBCC,74\$	:LOAD OLD SOFT BCC
3474	022406	022700	000010		CMP	#10,RO	:DONE YET?
3475	022412	001346			BNE	72\$	:BR IF NOT DONE
3476	022414	104401			SCOPE1		:SCOPE SUBTEST (SW09=1)
3477	022416	104400		77\$:	SCOPE		:SCOPE THIS TEST
3478							
3479							
3480							:***** TEST 46 *****
3481							:*TEST OF CRC OPERATION
3482							:*USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3483							:*377, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3484							:*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3485							:*****
3486							
3487							
3488							: TEST 46
3489	022420	012737	000046	001226	TST46:	MOV	#46,TSTNO
3490	022426	012737	023012	001216		MOV	#TST47,NEXT
3491	022434	012737	022470	001220		MOV	#64\$,LOCK
3492							
3493	022442	104412				MSTCLR	:R1 CONTAINS BASE DMC11 ADDRESS
3494	022444	005061	000004		CLR	4(R1)	:MASTER CLEAR DMC11
3495	022450	104414			ROMCLK		:CLEAR PORT4
3496	022452	122117			122117		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3497	022454	004737	033374		JSR	PC,CLRIO	:PUT LINE UNIT IN BITSTUFF MODE
3498	022460	005037	033612		CLR	BITCON	:DO THIS AFTER MODE IS SET
3499	022464	012711	004000		MOV	#BIT11,(R1)	:CONSECUTIVE 1'S COUNTER INIT TO 0
3500	022470	004737	033374	64\$:	JSR	PC,CLRIO	:SET LU LOOP
3501	022474	005000			CLR	RO	:CLEAR BCC REGISTERS
3502	022476	012737	102010	033030	MOV	#CRC.CCITT,XPOLY	:START SHIFT COUNTER AT ZERO
3503	022504	012737	000377	022570	MOV	#377,66\$	:LOAD POLYNOMIAL FOR SOFTWARE BCC
3504	022512	005037	022572		CLR	67\$	:LOAD CHAR FOR SOFTWARE BCC
3505	022516	005137	022572		COM	67\$	:CLEAR OLD SOFTWARE BCC
3506	022522	004737	033034		JSR	PC,BCCLO	:START AT -1
3507	022526	000377			377		:LOAD OUT SILO WITH 2 SYNCs
3508	022530	104415	000021		DATACLK,	21	:AND THE CHARACTER 377
3509	022534	005037	033612		CLR	BITCON	:GET TRANSMITTER ACTIVE
3510	022540	005037	022554		CLR	60\$	:CLEAR BIT COUNTER

3511	022544	104415	000001	65\$:	DATACLK,	1	;SHIFT BCC ONCE
3512	022550	004537	033474		JSR	R5,STFFCR	;CHECK FOR STUFFING ZEROS
3513	022554	000000		60\$:	0		;CHARACTER
3514	022556	000001			1		;SHIFT COUNT
3515	022560	005200			INC	R0	;BUMP SHIFT COUNT
3516	022562	004537	032706		JSR	R5,SIMBCC	;CALCULATE SOFTWARE BCC LSB
3517	022566	000001			1		;ONE SHIFT
3518	022570	000000		66\$:	0		;DATA CHARACTER
3519	022572	000000		67\$:	0		;OLD BCC
3520	022574	103405			BCS	68\$	;BR IF SOFT BCC LSB IS SET
3521	022576	004737	033146		JSR	PC,GETQ0	;GET HARDWARE TRANSMITTER BCC LSB
3522	022602	103006			BCC	69\$	;BR IF HARD BCC LSB IS CLEAR
3523	022604	104012			HLT	12	;ERROR, BCC LSB IS SET
3524	022606	000404			BR	69\$	;CONTINUE
3525	022610	004737	033146	68\$:	JSR	PC,GETQ0	;GET HARDWARE TRANSMITTER BCC LSB
3526	022614	103401			BCS	69\$	;BR IF HARD BCC LSB IS SET
3527	022616	104016			HLT	16	;ERROR, HARD BCC LSB IS CLEAR
3528	022620			69\$:			
3529	022620	013737	022570	022554	MOV	66\$,60\$	
3530	022626	006037	022570		ROR	66\$	;SHIFT SOFT DATA
3531	022632	013737	033032	022572	MOV	CALBCC,67\$	;LOAD OLD SOFT BCC
3532	022640	022700	000010		CMP	#10,R0	;DONE YET?
3533	022644	001337			BNE	65\$	;BR IF NOT DONE
3534	022646	104401			SCOPI		;SCOPE SUBTEST (SW09=1)
3535	022650	012737	022656	001220	MOV	#71\$,LOCK	;NEW SCOPE1
3536	022656	004737	033374		JSR	PC,CLRIO	;CLEAR BCC REGISTERS
3537	022662	005000			CLR	R0	;START SHIFT COUNTER AT ZERO
3538	022664	012737	102010	033030	MOV	#CRC.CCITT,XPOLY	;LOAD POLYNOMIAL FOR SOFTWARE BCC
3539	022672	012737	000377	022736	MOV	#377,73\$	;LOAD CHAR FOR SOFTWARE BCC
3540	022700	005037	022740		CLR	74\$	;CLEAR OLD SOFTWARE BCC
3541	022704	005137	022740		COM	74\$	;START AT -1
3542	022710	004737	033034		JSR	PC,BCCLD	;LOAD OUT SLD WITH 2 SYNC
3543	022714	000377			377		;AND THE CHARACTER 377
3544	022716	104415	000033		DATACLK,	33	;GET RECEIVER ACTIVE
3545	022722	104415	000001	72\$:	DATACLK,	1	;SHIFT BCC ONCE
3546	022726	005200			INC	R0	;BUMP SHIFT COUNT
3547	022730	004537	032706		JSR	R5,SIMBCC	;CALCULATE SOFTWARE BCC LSB
3548	022734	000001			1		;ONE SHIFT
3549	022736	000000		73\$:	0		;DATA CHARACTER
3550	022740	000000		74\$:	0		;OLD BCC
3551	022742	103405			BCS	75\$	;BR IF SOFT BCC LSB IS SET
3552	022744	004737	033160		JSR	PC,GETQI	;GET HARDWARE RECEIVER BCC LSB
3553	022750	103006			BCC	76\$	;BR IF HARD BCC LSB IS CLEAR
3554	022752	104013			HLT	13	;ERROR, BCC LSB IS SET
3555	022754	000404			BR	76\$	;CONTINUE
3556	022756	004737	033160	75\$:	JSR	PC,GETQI	;GET HARDWARE RECEIVER BCC LSB
3557	022762	103401			BCS	76\$	;BR IF HARD BCC LSB IS SET
3558	022764	104017			HLT	17	;ERROR, BCC LSB IS CLEAR
3559	022766			76\$:			
3560	022766	006037	022736		ROR	73\$	;SHIFT SOFT DATA
3561	022772	013737	033032	022740	MOV	CALBCC,74\$	;LOAD OLD SOFT BCC
3562	023000	022700	000010		CMP	#10,R0	;DONE YET?
3563	023004	001346			BNE	72\$	;BR IF NOT DONE
3564	023006	104401			SCOPI		;SCOPE SUBTEST (SW09=1)
3565	023010	104400		77\$:	SCOPE		;SCOPE THIS TEST
3566							

```

3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577 023012 012737 000047 001226
3578 023020 012737 023356 001216
3579 023026 012737 023062 001220
3580
3581 023034 104412
3582 023036 005061 000004
3583 023042 104414
3584 023044 122117
3585 023046 004737 033374
3586 023052 005037 033612
3587 023056 012711 004000
3588 023062 004737 033374
3589 023066 005000
3590 023070 012737 102010 033030
3591 023076 012737 000125 023142
3592 023104 005037 023144
3593 023110 005137 023144
3594 023114 004737 033034
3595 023120 000125
3596 023122 104415 000021
3597 023126 104415 000001
3598 023132 005200
3599 023134 004537 032706
3600 023140 000001
3601 023142 000000
3602 023144 000000
3603 023146 103405
3604 023150 004737 033146
3605 023154 103006
3606 023156 104012
3607 023160 000404
3608 023162 004737 033146
3609 023166 103401
3610 023170 104016
3611 023172
3612 023172 006037 023142
3613 023176 013737 033032 023144
3614 023204 022700 000010
3615 023210 001346
3616 023212 104401
3617 023214 012737 023222 001220
3618 023222 004737 033374
3619 023226 005000
3620 023230 012737 102010 033030
3621 023236 012737 000125 023302
3622 023244 005037 023304

```

```

***** TEST 47 *****
*TEST OF CRC OPERATION
*USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
*125. VERIFY THE LSB OF THE BCC ON EACH SHIFT
*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
*****

; TEST 47
-----
TST47: MOV #47,TSTNO
MOV #TST50,NEXT
MOV #64$,LOCK

MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
CLR 4(R1) ;MASTER CLEAR DMC11
ROMCLK ;CLEAR PORT4
122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=53C4
JSR PC,CLRIO ;PUT LINE UNIT IN BITSTUFF MODE
CLR BITCON ;DO THIS AFTER MODE IS SET
MOV #BIT11,(R1) ;CONSECUTIVE 1'S COUNTER INIT TO 0
;SET LU LOOP
64$: JSR PC,CLRIO ;CLEAR BCC REGISTERS
CLR RO ;START SHIFT COUNTER AT ZERO
MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
MOV #125.66$ ;LOAD CHAR FOR SOFTWARE BCC
CLR 67$ ;CLEAR OLD SOFTWARE BCC
COM 67$ ;START AT -1
JSR PC,BCCLD ;LOAD OUT SILO WITH 2 SYNCs
125 ;AND THE CHARACTER 125
DATACLK, 21 ;GET TRANSMITTER ACTIVE
65$: DATACLK, 1 ;SHIFT BCC ONCE
INC RO ;BUMP SHIFT COUNT
JSR RS,SIMBCC ;CALCULATE SOFTWARE BCC LSB
1 ;ONE SHIFT
66$: 0 ;DATA CHARACTER
67$: 0 ;OLD BCC
BCS 68$ ;BR IF SOFT BCC LSB IS SET
JSR PC,GETQ0 ;GET HARDWARE TRANSMITTER BCC LSB
BCC 69$ ;BR IF HARD BCC LSB IS CLEAR
HLT 12 ;ERROR, BCC LSB IS SET
BR 69$ ;CONTINUE
68$: JSR PC,GETQ0 ;GET HARDWARE TRANSMITTER BCC LSB
BCS 69$ ;BR IF HARD BCC LSB IS SET
HLT 16 ;ERROR, HARD BCC LSB IS CLEAR
69$: ROR 66$ ;SHIFT SOFT DATA
MOV CALBCC,67$ ;LOAD OLD SOFT BCC
CMP #10,RO ;DONE YET?
BNE 65$ ;BR IF NOT DONE
SCOPE1 ;SCOPE SUBTEST (SW09=1)
MOV #71$,LOCK ;NEW SCOPE1
71$: JSR PC,CLRIO ;CLEAR BCC REGISTERS
CLR RO ;START SHIFT COUNTER AT ZERO
MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE BCC
MOV #125.73$ ;LOAD CHAR FOR SOFTWARE BCC
CLR 74$ ;CLEAR OLD SOFTWARE BCC

```

3623	023250	005137	023304		COM	74\$		; START AT -1
3624	023254	004737	033034		JSR	PC, BCCLD		; LOAD OUT SILO WITH 2 SYNCs
3625	023260	000125			125			; AND THE CHARACTER 125
3626	023262	104415	000032		DATA CLK,	32		; GET RECEIVER ACTIVE
3627	023266	104415	000001	72\$:	DATA CLK,	1		; SHIFT BCC ONCE
3628	023272	005200			INC	RO		; BUMP SHIFT COUNT
3629	023274	004537	032706		JSR	RS, SIMBCC		; CALCULATE SOFTWARE BCC LSB
3630	023300	000001			1			; ONE SHIFT
3631	023302	000000		73\$:	0			; DATA CHARACTER
3632	023304	000000		74\$:	0			; OLD BCC
3633	023306	103405			BCC	75\$		; BR IF SOFT BCC LSB IS SET
3634	023310	004737	033160		JSR	PC, GETQI		; GET HARDWARE RECEIVER BCC LSB
3635	023314	103006			BCC	76\$		; BR IF HARD BCC LSB IS CLEAR
3636	023316	104013			HLT	13		; ERROR, BCC LSB IS SET
3637	023320	000404			BR	76\$		; CONTINUE
3638	023322	004737	033160	75\$:	JSR	PC, GETQI		; GET HARDWARE RECEIVER BCC LSB
3639	023326	103401			BCC	76\$		; BR IF HARD BCC LSB IS SET
3640	023330	104017			HLT	17		; ERROR, BCC LSB IS CLEAR
3641	023332			76\$:				
3642	023332	006037	023302		ROR	73\$		; SHIFT SOFT DATA
3643	023336	013737	033032	023304	MOV	CALBCC, 74\$		; LOAD OLD SOFT BCC
3644	023344	022700	000010		CMP	#10, RO		; DONE YET?
3645	023350	001346			BNE	72\$		; BR IF NOT DONE
3646	023352	104401			SCOPI			; SCOPE SUBTEST (SW09=1)
3647	023354	10440C		77\$:	SCOPE			; SCOPE THIS TEST
3648								
3649								
3650								
3651								
3652								
3653								
3654								
3655								
3656								
3657								
3658								
3659	023356	012737	000050	001226	TST50:	MOV	#50, TSTNO	
3660	023364	012737	023722	001216		MOV	#TST51, NEXT	
3661	023372	012737	023426	001220		MOV	#64\$, LOCK	
3662								
3663	023400	104412			MSTCLR			; R1 CONTAINS BASE DMC11 ADDRESS
3664	023402	005061	000004		CLR	4(R1)		; MASTER CLEAR DMC11
3665	023406	104414			ROMCLK			; CLEAR PORT4
3666	023410	122117			122117			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3667	023412	004737	033374		JSR	PC, CLR10		; PUT LINE UNIT IN BITSTUFF MODE
3668	023416	005037	033612		CLR	BITCON		; DO THIS AFTER MODE IS SET
3669	023422	012711	004000		MOV	#BIT11, (R1)		; CONSECUTIVE 1'S COUNTER INIT TO 0
3670	023426	004737	033374	54\$:	JSR	PC, CLR10		; SET LU LOOP
3671	023432	005000			CLR	RO		; CLEAR BCC REGISTERS
3672	023434	012737	102010	033030	MOV	#CRC.CCITT, XPOLY		; START SHIFT COUNTER AT ZERO
3673	023442	012737	000252	023506	MOV	#252, 66\$;		; LOAD POLYNOMIAL FOR SOFTWARE BCC
3674	023450	005037	023510		CLR	67\$		; LOAD CHAR FOR SOFTWARE BCC
3675	023454	005137	023510		COM	67\$		; CLEAR OLD SOFTWARE BCC
3676	023460	004737	033034		JSR	PC, BCCLD		; START AT -1
3677	023464	000252			252			; LOAD OUT SILO WITH 2 SYNCs
3678	023466	104415	000021		DATA CLK,	21		; AND THE CHARACTER 252
								; GET TRANSMITTER ACTIVE

***** TEST 50 *****
 ; *TEST OF CRC OPERATION
 ; *USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
 ; *252, VERIFY THE LSB OF THE BCC ON EACH SHIFT
 ; *TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
 ; *****

; TEST 50

 MOV #50, TSTNO
 MOV #TST51, NEXT
 MOV #64\$, LOCK

MSTCLR
 CLR 4(R1)
 ROMCLK
 122117
 JSR PC, CLR10
 CLR BITCON
 MOV #BIT11, (R1)
 JSR PC, CLR10

; R1 CONTAINS BASE DMC11 ADDRESS
 ; MASTER CLEAR DMC11
 ; CLEAR PORT4
 ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ; PUT LINE UNIT IN BITSTUFF MODE
 ; DO THIS AFTER MODE IS SET
 ; CONSECUTIVE 1'S COUNTER INIT TO 0
 ; 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
 ; START AT -1
 ; LOAD OUT SILO WITH 2 SYNCs
 ; AND THE CHARACTER 252
 ; GET TRANSMITTER ACTIVE

3679	023472	104415	000001	65\$:	DATACLK,	1	:SHIFT BCC ONCE
3680	023476	005200			INC	RO	:BUMP SHIFT COUNT
3681	023500	004537	032706		JSR	R5,SIMBCC	:CALCULATE SOFTWARE BCC LSB
3682	023504	000001			1		:ONE SHIFT
3683	023506	000000		66\$:	0		:DATA CHARACTER
3684	023510	000000		67\$:	0		:OLD BCC
3685	023512	103405			BCS	68\$	:BR IF SOFT BCC LSB IS SET
3686	023514	004737	033146		JSR	PC,GETQ0	:GET HARDWARE TRANSMITTER BCC LSB
3687	023520	103006			BCC	69\$	:BR IF HARD BCC LSB IS CLEAR
3688	023522	104012			HLT	12	:ERROR, BCC LSB IS SET
3689	023524	000404			BR	69\$	:CONTINUE
3690	023526	004737	033146	68\$:	JSR	PC,GETQ0	:GET HARDWARE TRANSMITTER BCC LSB
3691	023532	103401			BCS	69\$	:BR IF HARD BCC LSB IS SET
3692	023534	104016			HLT	16	:ERROR, HARD BCC LSB IS CLEAR
3693	023536			69\$:			
3694	023536	006037	023506		ROR	66\$	:SHIFT SOFT DATA
3695	023542	013737	033032	023510	MOV	CALBCC,67\$	:LOAD OLD SOFT BCC
3696	023550	022700	000010		CMP	#10,RO	:DONE YET?
3697	023554	001346			BNE	65\$	:BR IF NOT DONE
3698	023556	104401			SCOPE1		:SCOPE SUBTEST (SW09=1)
3699	023560	012737	023566	001220	MOV	#71\$,LOCK	:NEW SCOPE1
3700	023566	004737	033374	71\$:	JSR	PC,CLRIO	:CLEAR BCC REGISTERS
3701	023572	005000			CLR	RO	:START SHIFT COUNTER AT ZERO
3702	023574	012737	102010	033030	MOV	#CRC.CCITT,XPOLY	:LOAD POLYNOMIAL FOR SOFTWARE ECC
3703	023602	012737	000252	023646	MOV	#252,73\$;	:LOAD CHAR FOR SOFTWARE BCC
3704	023610	005037	023650		CLR	74\$	:CLEAR OLD SOFTWARE BCC
3705	023614	005137	023650		COM	74\$	:START AT -1
3706	023620	004737	033034		JSR	PC,BCCLD	:LOAD OUT SILO WITH 2 SYNC
3707	023624	000252			252		:AND THE CHARACTER 252
3708	023626	104415	000032		DATACLK,	32	:GET RECEIVER ACTIVE
3709	023632	104415	000001	72\$:	DATACLK,	1	:SHIFT BCC ONCE
3710	023636	005200			INC	RO	:BUMP SHIFT COUNT
3711	023640	004537	032706		JSR	R5,SIMBCC	:CALCULATE SOFTWARE BCC LSB
3712	023644	000001			1		:ONE SHIFT
3713	023646	000000		73\$:	0		:DATA CHARACTER
3714	023650	000000		74\$:	0		:OLD BCC
3715	023652	103405			BCS	75\$	:BR IF SOFT BCC LSB IS SET
3716	023654	004737	033160		JSR	PC,GETQ1	:GET HARDWARE RECEIVER BCC LSB
3717	023660	103006			BCC	76\$	:BR IF HARD BCC LSB IS CLEAR
3718	023662	104013			HLT	13	:ERROR, BCC LSB IS SET
3719	023664	000404			BR	76\$	:CONTINUE
3720	023666	004737	033160	75\$:	JSR	PC,GETQ1	:GET HARDWARE RECEIVER BCC LSB
3721	023672	103401			BCS	76\$	:BR IF HARD BCC LSB IS SET
3722	023674	104017			HLT	17	:ERROR, BCC LSB IS CLEAR
3723	023676			76\$:			
3724	023676	006037	023646		ROR	73\$	:SHIFT SOFT DATA
3725	023702	013737	033032	023650	MOV	CALBCC,74\$	:LOAD OLD SOFT BCC
3726	023710	022700	000010		CMP	#10,RO	:DONE YET?
3727	023714	001346			BNE	72\$	:BR IF NOT DONE
3728	023716	104401			SCOPE1		:SCOPE SUBTEST (SW09=1)
3729	023720	104400		77\$:	SCOPE		:SCOPE THIS TEST
3730							
3731							
3732							
3733							
3734							

:***** TEST 51 *****
 :*TRANSMITTER CRC TEST
 :*USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK A BINARY

```

3735                                     ;*COUNT PATTERN, VERIFY THE LSB OF THE TRANSMITTER BCC ON EACH SHIFT
3736                                     ;:*****
3737                                     ;
3738                                     ; TEST 51
3739                                     ;-----
3740 023722 012737 000051 001226          TST51: MOV      #51,TSTNO
3741 023730 012737 024244 001216          MOV      #TST52,NEXT
3742                                     ;
3743 023736 104412                        MSTCLR
3744 023740 005061 000004                  CLR      4(R1)
3745 023744 104414                        ROMCLK
3746 023746 122117                        122117
3747 023750 004737 033374                  JSR      PC,CLRIO
3748 023754 005037 033612                  CLR      BITCON
3749 023760 012711 004000                  MOV      #BIT11,(R1)
3750 023764 005003                        CLR      R3
3751 023766 005004                        CLR      R4
3752 023770 005005                        CLR      R5
3753 023772 005037 024120                  CLR      4$
3754 023776 005137 024120                  COM      4$
3755 024002 012737 102010 033030          MOV      #CRC.CCITT,XPOLY
3756 024010 004737 033176                  JSR      PC,SYNLD
3757 024014 010461 000004                  MOV      R4,4(R1)
3758 024020 104414                        ROMCLK
3759 024022 122110                        122110
3760 024024 005204                        INC      R4
3761 024026 010461 000004                  MOV      R4,4(R1)
3762 024032 104414                        ROMCLK
3763 024034 122110                        122110
3764 024036 005204                        INC      R4
3765 024040 010461 000004                  MOV      R4,4(R1)
3766 024044 104414                        ROMCLK
3767 024046 122110                        122110
3768 024050 004737 032044                  JSR      PC,OCOR
3769 024054 104415 000021                  DATACLK,21
3770 024060 010537 024104                  MOV      R5,10$
3771 024064 012700 000001                  MOV      #1,R0
3772 024070 010537 024116                  1$: MOV      R5,3$
3773 024074 104415 000001                  2$: DATACLK,1
3774 024100 004537 033474                  JSR      R5,STFFCK
3775 024104 000000                        10$: 0
3776 024106 000001                        1
3777 024110 004537 032706                  JSR      R5,SIMBCC
3778 024114 000001                        1
3779 024116 000000                        3$: 0
3780 024120 000000                        4$: 0
3781 024122 103405                        BCS      5$
3782 024124 004737 033146                  JSR      PC,GETQ0
3783 024130 103006                        BCC      6$
3784 024132 104020                        HLT      20
3785 024134 000404                        BR      6$
3786 024136 004737 033146                  5$: JSR      PC,GETQ0
3787 024142 103401                        BCS      6$
3788 024144 104021                        HLT      21
3789                                     ;
3790                                     ;
3791                                     ;
3792                                     ;
3793                                     ;
3794                                     ;
3795                                     ;
3796                                     ;
3797                                     ;
3798                                     ;
3799                                     ;
3800                                     ;
3801                                     ;
3802                                     ;
3803                                     ;
3804                                     ;
3805                                     ;
3806                                     ;
3807                                     ;
3808                                     ;
3809                                     ;
3810                                     ;
3811                                     ;
3812                                     ;
3813                                     ;
3814                                     ;
3815                                     ;
3816                                     ;
3817                                     ;
3818                                     ;
3819                                     ;
3820                                     ;
3821                                     ;
3822                                     ;
3823                                     ;
3824                                     ;
3825                                     ;
3826                                     ;
3827                                     ;
3828                                     ;
3829                                     ;
3830                                     ;
3831                                     ;
3832                                     ;
3833                                     ;
3834                                     ;
3835                                     ;
3836                                     ;
3837                                     ;
3838                                     ;
3839                                     ;
3840                                     ;
3841                                     ;
3842                                     ;
3843                                     ;
3844                                     ;
3845                                     ;
3846                                     ;
3847                                     ;
3848                                     ;
3849                                     ;
3850                                     ;
3851                                     ;
3852                                     ;
3853                                     ;
3854                                     ;
3855                                     ;
3856                                     ;
3857                                     ;
3858                                     ;
3859                                     ;
3860                                     ;
3861                                     ;
3862                                     ;
3863                                     ;
3864                                     ;
3865                                     ;
3866                                     ;
3867                                     ;
3868                                     ;
3869                                     ;
3870                                     ;
3871                                     ;
3872                                     ;
3873                                     ;
3874                                     ;
3875                                     ;
3876                                     ;
3877                                     ;
3878                                     ;
3879                                     ;
3880                                     ;
3881                                     ;
3882                                     ;
3883                                     ;
3884                                     ;
3885                                     ;
3886                                     ;
3887                                     ;
3888                                     ;
3889                                     ;
3890                                     ;
3891                                     ;
3892                                     ;
3893                                     ;
3894                                     ;
3895                                     ;
3896                                     ;
3897                                     ;
3898                                     ;
3899                                     ;
3900                                     ;
3901                                     ;
3902                                     ;
3903                                     ;
3904                                     ;
3905                                     ;
3906                                     ;
3907                                     ;
3908                                     ;
3909                                     ;
3910                                     ;
3911                                     ;
3912                                     ;
3913                                     ;
3914                                     ;
3915                                     ;
3916                                     ;
3917                                     ;
3918                                     ;
3919                                     ;
3920                                     ;
3921                                     ;
3922                                     ;
3923                                     ;
3924                                     ;
3925                                     ;
3926                                     ;
3927                                     ;
3928                                     ;
3929                                     ;
3930                                     ;
3931                                     ;
3932                                     ;
3933                                     ;
3934                                     ;
3935                                     ;
3936                                     ;
3937                                     ;
3938                                     ;
3939                                     ;
3940                                     ;
3941                                     ;
3942                                     ;
3943                                     ;
3944                                     ;
3945                                     ;
3946                                     ;
3947                                     ;
3948                                     ;
3949                                     ;
3950                                     ;
3951                                     ;
3952                                     ;
3953                                     ;
3954                                     ;
3955                                     ;
3956                                     ;
3957                                     ;
3958                                     ;
3959                                     ;
3960                                     ;
3961                                     ;
3962                                     ;
3963                                     ;
3964                                     ;
3965                                     ;
3966                                     ;
3967                                     ;
3968                                     ;
3969                                     ;
3970                                     ;
3971                                     ;
3972                                     ;
3973                                     ;
3974                                     ;
3975                                     ;
3976                                     ;
3977                                     ;
3978                                     ;
3979                                     ;
3980                                     ;
3981                                     ;
3982                                     ;
3983                                     ;
3984                                     ;
3985                                     ;
3986                                     ;
3987                                     ;
3988                                     ;
3989                                     ;
3990                                     ;
3991                                     ;
3992                                     ;
3993                                     ;
3994                                     ;
3995                                     ;
3996                                     ;
3997                                     ;
3998                                     ;
3999                                     ;
4000                                     ;

```

3791	024146	006037	024104		ROR	10\$	:SHIFT CHAR FOR STUFF CHECK
3792	024152	005300			DEC	R0	:DEC STUFF CHECK SHIFT COUNT
3793	024154	001004			BNE	11\$	:BR IF NOT DONE THIS CHARACTER
3794	024156	012700	000010		MOV	#10,R0	:RESET BIT COUNT TO 10
3795	024162	010537	024104		MOV	R5,10\$	:LOAD NEXT CHAR FOR STUFF CHECK
3796	024166			11\$:			
3797	024166	006037	024116		ROR	3\$	:SHIFT SOFT DATA
3798	024172	013737	033032	024120	MOV	CALBCC,4\$	:LOAD OLD SOFT BCC
3799	024200	005203			INC	R3	:INCREMENT BIT COUNTER
3800	024202	022703	000010		CMP	#10,R3	:DONE A FULL CHARACTER YET?
3801	024206	001332			BNE	2\$	:BR IF NO
3802	024210	005003			CLR	R3	:RESTART BIT COUNTER
3803	024212	005204			INC	R4	:INCREMENT DATA FOR SILO
3804	024214	022704	000400		CMP	#400,R4	:DONE BINARY COUNT YET?
3805	024220	003404			BLE	9\$	:BR IF YES
3806	024222	010461	000004		MOV	R4,4(R1)	:PORT4+DATA
3807	024226	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3808	024230	122110			122110		:LOAD OUT DATA
3809	024232	005205			INC	R5	:INCREMENT DATA
3810	024234	022705	000400		CMP	#400,R5	:DONE BINARY PATTERN YET?
3811	024240	001313			BNE	1\$	:BR IF NO
3812	024242	104400			7\$:	SCOPE	:SCOPE THIS TEST

***** TEST 52 *****
 :RECEIVER CRC TEST
 :USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK A BINARY
 :COUNT PATTERN, VERIFY THE LSB OF THE RECEIVER BCC ON EACH SHIFT
 :*****

: TEST 52

3823	024244	012737	000052	001226	TST52:	MOV	#52,TSTNO	
3824	024252	012737	024602	001216		MOV	#TST53,NEXT	
3825								
3826	024260	104412				MSTCLR		:R1 CONTAINS BASE DMC11 ADDRESS
3827	024262	005061	000004			CLR	4(R1)	:MASTER CLEAR DMC11
3828	024266	104414				ROMCLK		:CLEAR PORT4
3829	024270	122117				122117		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3830	024272	004737	033374			JSR	PC,CLR10	:PUT LINE UNIT IN BITSTUFF MODE
3831	024276	005037	033612			CLR	BITCON	:DO THIS AFTER MODE IS SET
3832	024302	012711	004000			MOV	#BIT11,(R1)	:CONSECUTIVE 1'S COUNTER INIT TO C
3833	024306	005003				CLR	R3	:SET LINE UNIT LOOP
3834	024310	005004				CLR	R4	:ZERO BIT COUNT
3835	024312	005005				CLR	R5	:R4 CONTAINS CHAR TO BE LOADED IN SILO
3836	024314	005037	024446			CLR	4\$	:R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
3837	024320	005137	024446			COM	4\$	:CLEAR SOFT BCC
3838	024324	012737	102010	033030		MOV	#CRC.CCITT,XPOLY	:START AT -1
3839	024332	004737	033176			JSR	PC,SYNLD	:LOAD POLYNOMIAL
3840	024336	010461	000004			MOV	R4,4(R1)	:LOAD SILO WITH 2 SYNCs, SOM SET
3841	024342	104414				ROMCLK		:PORT4+CHAR
3842	024344	122110				122110		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3843	024346	005204				INC	R4	:LOAD OUT DATA
3844	024350	010461	000004			MOV	R4,4(R1)	:INCREMENT TO NEXT CHARACTER
3845	024354	104414				ROMCLK		:PORT4+CHAR
3846	024356	122110				122110		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

:LOAD OUT DATA

3847	024360	005204		INC	R4		; INCREMENT TO NEXT CHARACTER
3848	024362	010461	000004	MOV	R4,4(R1)		; PORT4+CHAR
3849	024366	104414		ROMCLK			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3850	024370	122110		122110			; LOAD OUT DATA
3851	024372	004737	032044	JSR	PC,OCOR		; WAIT FOR OCOR
3852	024376	104415	000032	DATACLK,	32		; CLOCK DATA
3853	024402	010537	024432	MOV	R5,10\$		; START WITH ZERO
3854	024406	005237	024432	INC	10\$		; TRANSMITTER IS ONE CHAR AHEAD
3855	024412	012700	000010	MOV	#10,R0		; R0 = CHAR COUNT
3856	024416	010537	024444	MOV	R5,3\$		; LOAD CHAR FOR SOFT CRC
3857	024422	104415	000001	DATACLK,	1		; SHIFT BCC ONCE
3858	024426	004537	033474	JSR	R5,STFFCK		; CHECK BIT STUFFING
3859	024432	000000		0			; CHARACTER
3860	024434	000001		1			; SHIFT COUNT
3861	024436	004537	032706	JSR	R5,SIMBCC		; CALCULATE SOFT BCC
3862	024442	000001		1			; SOFT SHIFT COUNT
3863	024444	000000		0			; SOFT CHARACTER
3864	024446	000000		0			; OLD SOFT BCC
3865	024450	103405		BCC	5\$		; BR IF SOFT BCC LSB IS SET
3866	024452	004737	033160	JSR	PC,GETQI		; GET HARDWARE RECEIVER BCC _LSB
3867	024456	103006		BCC	6\$		; BR IF OK (CLEARED)
3868	024460	104022		HLT	22		; ERROR, BCC LSB WAS SET
3869	024462	000404		BR	6\$		; CONTINUE WITH TEST
3870	024464	004737	033160	JSR	PC,GETQI		; GET HARDWARE RECEIVER BCC _LSB
3871	024470	103401		BCC	6\$		; BR IF OK (SET)
3872	024472	104023		HLT	23		; ERROR, BCC LSB WAS CLEAR
3873							
3874	024474						
3875	024474	006037	024432	ROR	10\$		; SHIFT CHAR FOR STUFF CHECK
3876	024500	005300		DEC	R0		; DEC STUFF CHECK SHIFT COUNT
3877	024502	001010		BNE	11\$		; BR IF NOT DONE THIS CHARACTER
3878	024504	012700	000010	MOV	#10,R0		; RESET BIT COUNT TO 10
3879	024510	010537	024432	MOV	R5,10\$		; LOAD NEXT CHAR FOR STUFF CHECK
3880	024514	005237	024432	INC	10\$		; TRANSMITTER IS 2 CHAR AHEAD
3881	024520	005237	024432	INC	10\$		
3882	024524						
3883	024524	006037	024444	ROR	3\$		; SHIFT SOFT DATA
3884	024530	013737	033032	MOV	CALBCC,4\$		; LOAD OLD SOFT BCC
3885	024536	005203		INC	R3		; INCREMENT BIT COUNTER
3886	024540	022703	000010	CMP	#10,R3		; DONE A FULL CHARACTER YET?
3887	024544	001326		BNE	2\$		; BR IF NO
3888	024546	005003		CLR	R3		; RESTART BIT COUNTER
3889	024550	005204		INC	R4		; INCREMENT DATA FOR SILO
3890	024552	022704	000400	CMP	#400,R4		; DONE BINARY COUNT YET?
3891	024556	003404		BLE	9\$		; BR IF YES
3892	024560	010461	000004	MOV	R4,4(R1)		; PORT4+DATA
3893	024564	104414		ROMCLK			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3894	024566	122110		122110			; LOAD OUT DATA
3895	024570	005205		INC	R5		; INCREMENT DATA
3896	024572	022705	000400	CMP	#400,R5		; DONE BINARY PATTERN YET?
3897	024576	001307		BNE	1\$		; BR IF NO
3898	024600	104400		SCOPE			; SCOPE THIS TEST
3899							
3900							
3901							
3902							

;***** TEST 53 *****
 ;*TRANSMITTER BITSTUFF CRC TEST

```

3903 ;*THIS TEST TRANSMITS A FOUR CHARACTER MESSAGE WITH CRC
3904 ;*BOTH DATA AND THE BCC ARE VERIFIED IN THE BIT
3905 ;*WINDOW. THE FOUR CHARACTERS ARE 0,125,252,377
3906 ;*THE TRANSMITTER IS CHECKED FOR GOING TO A MARK STATE AFTER THE BCC
3907 ;*****
3908
3909 ; TEST 53
3910 ;-----
3911 024602 012737 000053 001226 TST53: MOV #53,TSTNO
3912 024610 012737 025304 001216 MOV #TST54,NEXT
3913
3914 024616 104412 MSTCLR ;R1 CONTAINS BASE DMC11 ADDRESS
3915 024620 005061 000004 CLR 4(R1) ;MASTER CLEAR DMC11
3916 024624 104414 ROMCLK ;CLEAR PORT4
3917 024626 122117 122117 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3918 024630 004737 033374 JSR PC,CLAIO ;PUT LINE UNIT IN BITSTUFF MODE
3919 024634 005037 033612 CLR BITCON ;DO THIS AFTER MODE IS SET
3920 ;CONSECUTIVE 1'S COUNTER INIT TO 3
3921
3922 ;LOAD OUT DATA SILO
3923 024640 012711 004000 MOV #BIT11,(R1) ;SET LINE UNIT LOOP
3924 024644 012704 033614 MOV #MESDAT,R4 ;LOAD POINTER TO DATA
3925 024650 005037 024760 CLR 10$ ;CLEAR SOFT BCC
3926 024654 005137 024760 COM 10$ ;START AT -1
3927 024660 012700 000004 MOV #4,R0 ;LOAD CHARACTER COUNT
3928 024664 004737 033176 JSR PC,SYNLD ;LOAD 2 FLAG CHARACTERS IN OUT SILO
3929 024670 004737 032176 JSR PC,OUTRDY ;WAIT FOR OUTRDY
3930 024674 004537 033332 JSR R5,MESLD ;LOAD SILO WITH 4 CHAR MESS
3931 024700 033614 MESDAT ;ADDRESS OF MESSAGE
3932 024702 000004 4 ;NUMBER OF CHARACTERS
3933 024704 004737 033306 JSR PC,EOM ;LOAD GARBAGE CHARACTER, WITH EOM SET
3934 024710 004737 033306 JSR PC,EOM
3935 024714 004737 032044 JSR PC,OCOR ;WAIT FOR OCOR
3936 024720 005003 CLR R3 ;CLEAR BIT COUNTER
3937 024722 104415 000022 DATACLK,22 ;CLOCK DATA
3938 024726 112405 12$: MOVB (R4)+,R5 ;LOAD R5 WITH CHAR
3939 024730 010502 MOV R5,R2 ;LOAD R2 WITH CHAR
3940
3941 ;CHECK FIRST FOUR CHARACTER MESSAGE
3942 ;IN THE BIT WINDOW (0,125,252,377)
3943
3944 024732 010537 025026 MOV R5,71$ ;LOAD FOR STUFF CHECK
3945 024736 012737 102010 033030 MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL
3946 024744 010537 024756 MOV R5,67$ ;LOAD SOFT CHAR FOR BCC
3947 024750 004537 032706 JSR R5,SIMBCC ;CALCULATE SOFT BCC
3948 024754 000010 10 ;SHIFT COUNT
3949 024756 000000 67$: 0 ;CHARACTER
3950 024760 000000 10$: 0 ;OLD BCC
3951 024762 013737 033032 024760 MOV CALBCC,10$ ;LOAD SOFT BCC FOR NEXT SHIFT
3952 024770 104415 000001 64$: DATACLK,1 ;SHIFT DATA IN TO BIT WINDOW
3953 024774 106002 PORB R2 ;SHIFT SOFT DATA
3954 024776 103005 BCC 65$ ;BR IF A SPACE
3955 025000 004737 032012 JSR PC,GETSI ;LOOK AT BIT WINDOW
3956 025004 103406 BCS 66$ ;BR IF OK (MARK)
3957 025006 104006 HLT 6 ;ERROR, BIT WINDOW WAS A SPACE
3958 025010 000404 BR 66$ ;CONTINUE

```

3959	025012	004737	032012	65\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
3960	025016	103001			BCC	66\$	;BR IF OK (SPACE)
3961	025020	104006			HLT	6	;ERROR, BIT WINDOW WAS A MARK
3962	025022			66\$:			
3963	025022	004537	033474		JSR	R5,STFFCK	
3964	025026	000000		71\$:	0		
3965	025030	000001			1		
3966	025032	110237	025026		MOV8	R2,71\$	;SHIFT FOR NEXT STUFF CHECK
3967	025036	005203			INC	R3	;BUMP BIT COUNTER
3968	025040	022703	000010		CMP	#10,R3	;DONE FULL 8 BITS YET
3969	025044	001351			BNE	64\$	;BR IF NO
3970	025046	005003			CLR	R3	;CLEAR BIT COUNTER
3971	025050	005300			DEC	R0	;DEC CHARACTER COUNT
3972	025052	001325			BNE	12\$	;BR IF NOT DONE YET
3973							
3974							;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
3975							
3976	025054	005137	033032		COM	CALBCC	;ADJUST BCC FOR SDLC
3977	025060	013700	033032		MOV	CALBCC,R0	;PUT BCC IN R0
3978	025064	010037	025126		MOV	R0,72\$	;LOAD BCC FOR STUFF CHECK
3979	025070	104415	000001	69\$:	DATACLK,1		;SHIFT HARDWARE BCC
3980	025074	006000			ROR	R0	;SHIFT SOFT BCC
3981	025076	103005			BCC	69\$	;BR IF CARRY CLEAR
3982	025100	004737	032012		JSR	PC,GETSI	;LOOK AT BIT WINDOW
3983	025104	103406			BCS	70\$	;BR IF OK (MARK)
3984	025106	104014			HLT	14	;ERROR, CRC WRONG (SPACE)
3985	025110	000404			BR	70\$	;CONTINUE
3986	025112	004737	032012	69\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
3987	025116	103001			BCC	70\$	;BR IF OK (SPACE)
3988	025120	104014			HLT	14	;ERROR, CRC WRONG (MARK)
3989	025122			70\$:			
3990	025122	004537	033474		JSR	R5,STFFCK	;CHECK BCC CHAR FOR ZERO STUFFS
3991	025126	000000		72\$:	0		;CHARACTER
3992	025130	000001			1		;SHIFT COUNT
3993	025132	010037	025126		MOV	R0,72\$	;SHIFT SOFTBCC ONCE
3994	025136	005203			INC	R3	;BUMP BIT COUNTER
3995	025140	022703	000020		CMP	#20,R3	;FINISHED BCC YET?
3996	025144	001351			BNE	68\$	;BR IF NO
3997	025146	005003			CLR	R3	;CLEAR BIT COUNTER
3998							
3999							;CHECK FOR FLAG TO FOLLOW BCC
4000							
4001	025150	012737	000176	001252	MOV	#1B'01111110',TEMP3	;PUT FLAG CHARACTER IN TEMP3
4002	025156	104415	000001		DATACLK,1		;CLOCK FLAG ONCE
4003	025162	106037	001252	73\$:	RORB	TEMP3	;SHIFT SOFT FLAG
4004	025166	103405			BCS	74\$	;BR IF BIT IS MARK
4005	025170	004737	032012		JSR	PC,GETSI	;LOOK AT BIT WINDOW
4006	025174	103006			BCC	75\$	;BR IF OK
4007	025176	104026			HLT	26	;ERROR IN FLAG CHAR
4008	025200	000404			BR	75\$	
4009	025202	004737	032012	74\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
4010	025206	103401			BCS	75\$	;BR IF OK
4011	025210	104026			HLT	26	;ERROR IN FLAG CHAR
4012	025212	005203		75\$:	INC	R3	;INC BIT COUNT
4013	025214	022703	000010		CMP	#10,R3	;FLAG DONE YET?
4014	025220	001356			BNE	73\$	;BR IF NO

```

4015 025222 005003          CLR    R3          ;CLEAR BIT COUNT
4016
4017          ;CHECK TO SEE IF TRANSMITTER IS MARKING
4018
4019 025224 104415 000001    2$:  DATACLK, 1          ;CLOCK TRANSMITTER
4020 025230 004737 032012    JSR    PC,GETSI 1        ;LOOK AT WINDOW
4021 025234 103401          BCS    3$          ;IT SHOULD BE MARKING
4022 025236 104024          HLT    24          ;ERROR, BIT WAS A SPACE
4023 025240 005203          3$:  INC    R3          ;BUMP BIT COUNTER
4024 025242 022703 000007    CMP    #7,R3        ;DONE YET
4025 025246 001366          BNE    2$          ;BR IF NO
4026 025250 104415 000010    DATACLK, 10        ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
4027 025254 005003          CLR    R3          ;CLEAR BIT COUNTER
4028 025256 104415 000001    4$:  DATACLK, 1        ;SHIFT OUT NEXT BIT
4029 025262 004737 032012    JSR    PC,GETSI 1        ;LOOK AT BIT WINDOW
4030 025266 103401          BCS    +4          ;BR IF IT IS A MARK
4031 025270 104024          HLT    24          ;ERROR, TRANSMITTER IS NOT MARKING
4032 025272 005203          INC    R3          ;INC BIT COUNT
4033 025274 022703 000020    CMP    #20,R3       ;DONE YET?
4034 025300 001366          BNE    4$          ;BR IF NO
4035 025302 104400          5$:  SCOPE          ;SCOPE THIS TEST
4036
4037
4038          ;***** TEST 54 *****
4039          ;*RECEIVER BITSTUFF CRC TEST
4040          ;*THIS TEST CLOCKS A FOUR CHARACTER MESSAGE WITH BCC
4041          ;*AND VERIFYS CORRECT DATA RECEPTION AND BCC MATCH
4042          ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4043          ;*****
4044
4045          ; TEST 54
4046          ;-----
4047 025304 012737 000054 001226    TST54:  MOV    #54,TSTNO
4048 025312 012737 025526 001216    MOV    #TST55,NEXT
4049
4050 025320 104412          MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
4051 025322 005061 000004          CLR    4(R1)        ;MASTER CLEAR DMC11
4052 025326 104414          ROMCLK          ;CLEAR PORT4
4053 025330 122117          122117          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4054 025332 004737 033374          JSR    PC,CLRIO      ;PUT LINE UNIT IN BITSTUFF MODE
4055 025336 012711 004000          MOV    #BIT11,(R1)    ;DO THIS AFTER MODE IS SET
4056 025342 012702 033614          MOV    #MESDAT,R2    ;SET LINE UNIT LOOP
4057 025346 012700 000004          MOV    #4,R0         ;LOAD POINTER TO DATA
4058 025352 004737 033176          JSR    PC,SYNLD      ;LOAD CHARACTER COUNT
4059 025356 004737 032176          JSR    PC,OUTRDY     ;LOAD 2 FLAG CHARACTERS IN OUT SILO
4060 025362 004537 033332          JSR    R5,MESLD      ;WAIT FOR OUTRDY
4061 025366 033614          MESDAT          ;LOAD SILO WITH 4 CHAR MESS
4062 025370 000004          4          ;ADDRESS OF MESSAGE
4063 025372 004737 033306          JSR    PC,EOM        ;NUMBER OF CHARACTERS
4064 025376 004737 033306          JSR    PC,EOM        ;LOAD GARBAGE CHARACTER, WITH ECM SET
4065 025402 004737 032044          JSR    PC,OCOR
4066 025406 104415 000115          DATACLK,11$       ;WAIT FOR OCOR
4067 025412 004737 032652          3$:  JSR    PC,INRDY    ;CLOCK DATA
4068 025416 104414          ROMCLK          ;WAIT FOR INRDY
4069 025420 021204          021204          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4070 025422 016104 000004          MOV    4(R1),R4      ;GET IN DATA
          ;PUT "FOUND" IN R4

```

4071	025426	112205			MOVB	(R2)+,R5	:PUT "EXPECTED" IN R5
4072	025430	120504			CMPB	R5,R4	:COMPARE RECEIVED DATA
4073	025432	001401			BEQ	1\$	:BR IF OK
4074	025434	104010			HLT	10	:DATA ERROR
4075	025436	005300			DEC	RC	:DEC CHARACTER COUNT
4076	025440	001364			BNE	3\$	:BR IF NOT DONE YET
4077							
4078							:CHECK TO SEE THAT IN BCC MATCH IS SET
4079							
4080	025442	004737	032652		JSR	PC,INRDY	:WAIT FOR INRDY
4081	025446	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4082	025450	021204			021204		:GET FIRST HALF OF CRC
4083	025452	116137	000004	001252	MOVB	4(R1),TEMP3	:PUT IN TEMP3
4084	025460	042737	177400	001252	BIC	#177400,TEMP3	:CLEAR HI BYTE
4085	025466	004737	032652		JSR	PC,INRDY	:WAIT FOR INRDY
4086	025472	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4087	025474	021244			021244		
4088	025476	016104	000004		MOV	4(R1),R4	:PUT "FOUND" IN R4
4089	025502	042704	000374		BIC	#374,R4	:CLEAR UNWANTED BITS
4090	025506	0:3705	000003		MOV	#3,R5	:PUT "EXPECTED" IN R5
4091	025512	120504			CMPB	R5,R4	:ARE IN BCC MATCH AND BLOCK END SET?
4092	025514	001401			BEQ	25\$	
4093	025516	104042			HLT	42	:IN BCC MATCH ERROR
4094	025520						
4095	025520	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4096	025522	021204			021204		:GET LAST HALF
4097	025524	104400			SCOPE		:SCOPE THIS TEST
4098							
4099							
4100							:***** TEST 55 *****
4101							:*BITSTUFF EOM FUNCTION TEST
4102							:*THIS TEST LOADS OUT SILO WITH: 2 FLAGS, 4 CHAR MESSAGE, EOM
4103							:*4 CHARACTER MESS, EOM. THE DATA STREAM IS CHECKED TO BE
4104							:*4 CHAR, BCC, FLAG, 4 CHAR, BCC, FLAG, MARKS. THIS TEST VERIFYS THAT
4105							:*THE CHARACTERS LOADED WITH EOM SET ARE LOST
4106							:*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
4107							:*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4108							:*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
4109							:*****
4110							
4111							: TEST 55
4112							
4113	025526	012737	000055	001226	TST55:	MOV	#55,TSTNO
4114	025534	012737	027126	001216		MOV	#TST56,NEXT
4115							
4116	025542	104412			MSTCLR		:R1 CONTAINS BASE DMC11 ADDRESS
4117	025544	005061	000004		CLR	4(R1)	:MASTER CLEAR DMC11
4118	025550	104414			ROMCLK		:CLEAR PORT4
4119	025552	122117			122117		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4120	025554	004737	033374		JSR	PC,CLR10	:PUT LINE UNIT IN BITSTUFF MODE
4121	025560	005037	033612		CLR	BITCON	:DO THIS AFTER MODE IS SET
4122							:CONSECUTIVE 1'S COUNTER INIT TO 0
4123							
4124							:LOAD OUT DATA SILO
4125	025564	012711	004000		MOV	#BIT11,(R1)	:SET LINE UNIT LOOP
4126	025570	012704	033614		MOV	#MESCAT,R4	:LOAD PCINTER TO DATA

```

4127 025574 005037 025724 CLR 10$ ;CLEAR SOFT BCC
4128 025600 005137 025724 COM 10$ ;START AT -1
4129 025604 012700 000004 MOV #4,R0 ;LOAD CHARACTER COUNT
4130 025610 004737 033176 JSR PC,SYNLD ;LOAD 2 FLAG CHARACTERS IN OUT SILC
4131 025614 004737 032176 JSR PC,OUTRDY ;WAIT FOR OUTRDY
4132 025620 004537 033332 JSR R5,MESLD ;LOAD SILO WITH 4 CHAR MESS
4133 025624 033614 MESDAT ;ADDRESS OF MESSAGE
4134 025626 000004 4 ;NUMBER OF CHARACTERS
4135 025630 004737 033306 JSR PC,EOM ;LOAD GARBAGE CHARACTER, WITH EOM SET
4136 025634 004737 033306 JSR PC,EOM
4137 025640 004537 033332 JSR R5,MESLD ;LOAD FOUR MORE CHARACTERS
4138 025644 033614 MESDAT ;ADDRESS OF MESSAGE
4139 025646 000004 4 ;NUMBER OF CHACTERS
4140 025650 004737 033306 JSR PC,EOM ;SET EOM
4141 025654 004737 033306 JSR PC,EOM ;SET EOM
4142 025660 004737 032044 JSR PC,OCOR ;WAIT FOR OCOR
4143 025664 005003 CLR R3 ;CLEAR BIT COUNTER
4144 025666 104415 000022 DATACLK,22 ;CLOCK DATA
4145 025672 112405 12$: MOVB (R4)+,R5 ;LOAD R5 WITH CHAR
4146 025674 010502 MOV R5,R2 ;LOAD R2 WITH CHAR

;CHECK FIRST FOUR CHARACTER MESSAGE
;IN THE BIT WINDOW (0,125,252,377)

4151 025676 010537 025772 MOV R5,71$ ;LOAD FOR STUFF CHECK
4152 025702 012737 102010 333030 MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL
4153 025710 010537 025722 MOV R5,67$ ;LOAD SOFT CHAR FOR BCC
4154 025714 004537 032706 JSR R5,SIMBCC ;CALCULATE SOFT BCC
4155 025720 000010 10 ;SHIFT COUNT
4156 025722 000000 67$: 0 ;CHARACTER
4157 025724 000000 10$: 0 ;OLD BCC
4158 025726 013737 033032 025724 MOV CALBCC,10$ ;LOAD SOFT BCC FOR NEXT SHIFT
4159 025734 104415 000001 64$: DATACLK,1 ;SHIFT DATA IN TO BIT WINDOW
4160 025740 106002 RORB R2 ;SHIFT SOFT DATA
4161 025742 103005 BCC 65$ ;BR IF A SPACE
4162 025744 004737 032012 JSR PC,GETSI ;LOOK AT BIT WINDOW
4163 025750 103406 BCS 66$ ;BR IF OK (MARK)
4164 025752 104006 HLT 6 ;ERROR, BIT WINDOW WAS A SPACE
4165 025754 000404 BR 66$ ;CONTINUE
4166 025756 004737 032012 65$: JSR PC,GETSI ;LOOK AT BIT WINDOW
4167 025762 103001 BCC 66$ ;BR IF OK (SPACE)
4168 025764 104006 HLT 6 ;ERROR, BIT WINDOW WAS A MARK
4169 025766 66$:
4170 025766 004537 033474 JSR R5,STFFCK
4171 025772 000000 71$: 0
4172 025774 000001 1
4173 025776 110237 025772 MOVB R2,71$ ;SHIFT FOR NEXT STUFF CHECK
4174 026002 005203 INC R3 ;BUMP BIT COUNTER
4175 026004 022703 000010 CMP #10,R3 ;DONE FULL 8 BITS YET
4176 026010 001351 BNE 64$ ;BR IF NO
4177 026012 005003 CLR R3 ;CLEAR BIT COUNTER
4178 026014 005300 DEC R0 ;DEC CHARACTER COUNT
4179 026016 001325 BNE 12$ ;BR IF NOT DONE YET

;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW

```

4183	026020	005137	033032		COM	CALBCC	;ADJUST BCC FOR SDLC
4184	026024	013700	033032		MOV	CALBCC,R0	;PUT BCC IN R0
4185	026030	010037	026072		MOV	R0,72\$	;LOAD BCC FOR STUFF CHECK
4186	026034	104415	000001	68\$:	DATACLK,	1	;SHIFT HARDWARE BCC
4187	026040	006000			ROR	R0	;SHIFT SOFT BCC
4188	026042	103005			BCC	69\$	;BR IF CARRY CLEAR
4189	026044	004737	032012		JSR	PC,GETSI	;LOOK AT BIT WINDOW
4190	026050	103406			BCS	70\$	;BR IF OK (MARK)
4191	026052	104014			HLT	14	;ERROR, CRC WRONG (SPACE)
4192	026054	000404			BR	70\$	;CONTINUE
4193	026056	004737	032012	69\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
4194	026062	03001			BCC	70\$	;BR IF OK (SPACE)
4195	026064	04014			HLT	14	;ERROR, CRC WRONG (MARK)
4196	026066			70\$:			
4197	026066	004537	033474		JSR	R5,STFFCK	;CHECK BCC CHAR FOR ZERO STUFFS
4198	026072	000000		72\$:	0		;CHARACTER
4199	026074	000001			1		;SHIFT COUNT
4200	026076	010037	026072		MOV	R0,72\$	;SHIFT SOFTBCC ONCE
4201	026102	005203			INC	R3	;BUMP BIT COUNTER
4202	026104	022703	000020		CMP	#20,R3	;FINISHED BCC YET?
4203	026110	001351			BNE	68\$	;BR IF NO
4204	026112	005003			CLR	R3	;CLEAR BIT COUNTER
4205							
4206							;CHECK FOR FLAG TO FOLLOW BCC
4207							
4208	026114	012737	000176	001252	MOV	#1B<01111110>,TEMP3	;PUT FLAG CHARACTER IN TEMP3
4209	026122	104415	000001		DATACLK,	1	;CLOCK FLAG ONCE
4210	026126	106037	001252	73\$:	RORB	TEMP3	;SHIFT SOFT FLAG
4211	026132	103405			BCS	74\$	;BR IF BIT IS MARK
4212	026134	004737	032012		JSR	PC,GETSI	;LOOK AT BIT WINDOW
4213	026140	103006			BCC	75\$	;BR IF OK
4214	026142	104026			HLT	26	;ERROR IN FLAG CHAR
4215	026144	000404			BR	75\$	
4216	026146	004737	032012	74\$:	JSR	PC,GETSI	;LOOK AT BIT WINDOW
4217	026152	103401			BCS	75\$	;BR IF OK
4218	026154	104026			HLT	26	;ERROR IN FLAG CHAR
4219	026156	005203		75\$:	INC	R3	;INC BIT COUNT
4220	026160	022703	000010		CMP	#10,R3	;FLAG DONE YET?
4221	026164	001356			BNE	73\$	;BR IF NO
4222	026166	005003			CLR	R3	;CLEAR BIT COUNT
4223	026170	012700	000004		MOV	#4,R0	;RESET CHARACTER COUNTER
4224	026174	012704	033014		MOV	#MESDAT,R4	;LOAD MESSAGE POINTER
4225	026200	005037	026242		CLR	11\$	;CLR SOFT BCC
4226	026204	005137	026242		COM	11\$	;ADJUST TO -1 FOR SDLC
4227	026210	112405		13\$:	MOVB	(R4)+,R5	;LOAD CHAR IN R5
4228	026212	010502			MOV	R5,R2	;LOAD CHAR IN R2
4229							
4230							;CHECK SECOND MESSAGE IN THE BIT WINDOW (0.125.252.377)
4231							
4232	026214	010537	026310		MOV	R5,83\$	;LOAD FOR STUFF CHECK
4233	026220	012737	102010	033030	MOV	#CRC.CCITT,XPOLY	;LOAD POLYNOMIAL
4234	026226	010537	026240		MOV	R5,79\$	;LOAD SOFT CHAR FOR BCC
4235	026232	004537	032706		JSR	R5,SIMBCC	;CALCULATE SOFT BCC
4236	026236	000010			10		;SHIFT COUNT
4237	026240	000000		79\$:	0		;CHARACTER
4238	026242	000000		11\$:	0		;OLD BCC

4239	026244	013737	033032	026242	76\$:	MOV	CALBCC,11\$	:LOAD SOFT BCC FOR NEXT SHIFT
4240	026252	104415	000001			DATACLK,	1	:SHIFT DATA IN TC BIT WINDOW
4241	026256	106002				RORB	R2	:SHIFT SOFT DATA
4242	026260	103005				BCC	77\$	:BR IF A SPACE
4243	026262	004737	032012			JSR	PC,GETSI	:LOOK AT BIT WINDOW
4244	026266	103406				BCS	78\$	:BR IF OK (MARK)
4245	026270	104006				HLT	6	:ERROR, BIT WINDOW WAS A SPACE
4246	026272	000404				BR	78\$	:CONTINUE
4247	026274	004737	032012		77\$:	JSR	PC,GETSI	:LOOK AT BIT WINDOW
4248	026300	103001				BCC	78\$	:BR IF OK (SPACE)
4249	026302	104006				HLT	6	:ERROR, BIT WINDOW WAS A MARK
4250	026304				78\$:			
4251	026304	004537	033474			JSR	R5,STFFCK	
4252	026310	000000			83\$:	0		
4253	026312	000001				1		
4254	026314	110237	026310			MOVB	R2,83\$	:SHIFT FOR NEXT STUFF CHECK
4255	026320	005203				INC	R3	:BUMP BIT COUNTER
4256	026322	022703	000010			CMP	#10,R3	:DONE FULL 8 BITS YET
4257	026326	001351				BNE	76\$	:BR IF NO
4258	026330	005003				CLR	R3	:CLEAR BIT COUNTER
4259	026332	005300				DEC	R0	:DEC CHARACTER COUNT
4260	026334	001325				BNE	13\$	:BR IF NOT DONE YET
4261								
4262								:CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4263								
4264	026336	005137	033032			COM	CALBCC	:ADJUST BCC FOR SDLC
4265	026342	013700	033032			MOV	CALBCC,R0	:PUT BCC IN R0
4266	026346	010037	026410			MOV	R0,84\$	:LOAD BCC FOR STUFF CHECK
4267	026352	104415	000001		80\$:	DATACLK,	1	:SHIFT HARDWARE BCC
4268	026356	006000				ROR	R0	:SHIFT SOFT BCC
4269	026360	103005				BCC	81\$	:BR IF CARRY CLEAR
4270	026362	004737	032012			JSR	PC,GETSI	:LOOK AT BIT WINDOW
4271	026366	103406				BCS	82\$	:BR IF OK (MARK)
4272	026370	104014				HLT	14	:ERROR, CRC WRONG (SPACE)
4273	026372	000404				BR	82\$	:CONTINUE
4274	026374	004737	032012		81\$:	JSR	PC,GETSI	:LOOK AT BIT WINDOW
4275	026400	103001				BCC	82\$	:BR IF OK (SPACE)
4276	026402	104014				HLT	14	:ERROR, CRC WRONG (MARK)
4277	026404				82\$:			
4278	026404	004537	033474			JSR	R5,STFFCK	:CHECK BCC CHAR FOR ZERO STUFFS
4279	026410	000000			84\$:	0		:CHARACTER
4280	026412	000001				1		:SHIFT COUNT
4281	026414	010037	026410			MOV	R0,84\$	:SHIFT SOFTBCC ONCE
4282	026420	005203				INC	R3	:BUMP BIT COUNTER
4283	026422	022703	000020			CMP	#20,R3	:FINISHED BCC YET?
4284	026426	001351				BNE	80\$	:BR IF NO
4285	026430	005003				CLR	R3	:CLEAR BIT COUNTER
4286								
4287								:CHECK FOR FLAG TO FOLLOW BCC
4288								
4289	026432	012737	000176	001252	85\$:	MOV	#18 01111110,TEMP3	:PUT FLAG CHARACTER IN TEMP3
4290	026440	104415	000001			DATACLK,	1	:CLOCK FLAG ONCE
4291	026444	106037	001252			RORB	TEMP3	:SHIFT SOFT FLAG
4292	026450	103405				BCS	86\$	:BR IF BIT IS MARK
4293	026452	004737	032012			JSR	PC,GETSI	:LOOK AT BIT WINDOW
4294	026456	103006				BCC	87\$	:BR IF OK

4295	026460	104026			HLT	26		:ERROR IN FLAG CHAR
4296	026462	000404			BR	87\$		
4297	026464	004737	032012	86\$:	JSR	PC,GETSI		:LOOK AT BIT WINDOW
4298	026470	103401			BCS	87\$		:BR IF OK
4299	026472	104026			HLT	26		:ERROR IN FLAG CHAR
4300	026474	005203		87\$:	INC	R3		:INC BIT COUNT
4301	026476	022703	000010		CMP	#10,R3		:FLAG DONE YET?
4302	026502	001356			BNE	85\$		:BR IF NO
4303	026504	005003			CLR	R3		:CLEAR BIT COUNT
4304								
4305								:CHECK TO SEE IF TRANSMITTER IS MARKING
4306								
4307	026506	104415	000001	2\$:	DATACLK,	1		:CLOCK TRANSMITTER
4308	026512	004737	032012		JSR	PC,GETSI		:LOOK AT WINDOW
4309	026516	103401			BCS	3\$		:IT SHOULD BE MARKING
4310	026520	104024			HLT	24		:ERROR, BIT WAS A SPACE
4311	026522	005203		3\$:	INC	R3		:BUMP BIT COUNTER
4312	026524	022703	000007		CMP	#7,R3		:DONE YET
4313	026530	001366			BNE	2\$		:BR IF NO
4314	026532	104415	000010		DATACLK,	10		:GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
4315	026536	005003			CLR	R3		:CLEAR BIT COUNTER
4316	026540	104415	000001	4\$:	DATACLK,	1		:SHIFT OUT NEXT BIT
4317	026544	004737	032012		JSR	PC,GETSI		:LOOK AT BIT WINDOW
4318	026550	103401			BCS	.+4		:BR IF IT IS A MARK
4319	026552	104024			HLT	24		:ERROR, TRANSMITTER IS NOT MARKING
4320	026554	005203			INC	R3		:INC BIT COUNT
4321	026556	022703	000020		CMP	#20,R3		:DONE YET?
4322	026562	001366			BNE	4\$		:BR IF NO
4323								
4324								:CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
4325								:WAS RECEIVED CORRECTLY (0,125,252,377)
4326								
4327	026564	104415	000001		DATACLK,	1		:GET LAST BIT IN RECEIVER
4328	026570	012703	000004		MOV	#4,R3		:R3=CHARACTER COUNT
4329	026574	012702	033614		MOV	#MESDAT,R2		:LOAD MESSAGE POINTER IN R2
4330	026600	004737	032652	40\$:	JSR	PC,INRDY		:WAIT FOR INRDY
4331	026604	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4332	026606	021204			021204			
4333	026610	016104	000004		MOV	4(R1),R4		:PUT "FOUND" IN R4
4334	026614	112205			MOV	(R2)+,R5		:PUT "EXPECTED" IN R5
4335	026616	120504			CMP	R5,R4		:IS RECEIVED DATA CORRECT?
4336	026620	001401			BEG	41\$		:BR IF YES
4337	026622	104010			HLT	10		:RECEIVE DATA ERROR
4338	026624	005303		41\$:	DEC	R3		:DEC CHARACTER COUNT
4339	026626	001364			BNE	40\$		:BR IF NOT DONE YET
4340								
4341								:CHECK TO SEE THAT IN BCC MATCH IS SET
4342								:AND THAT THE BCC WAS RECEIVED CORRECTLY
4343								
4344	026630	004737	032652		JSR	PC,INRDY		:WAIT FOR INRDY
4345	026634	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4346	026636	021204			021204			:GET FIRST HALF OF CRC
4347	026640	116137	000004	001252	MOV	4(R1),TEMP3		:PUT IN TEMP3
4348	026646	042737	177400	001252	BIC	#177400,TEMP3		:CLEAR HI BYTE
4349	026654	004737	032652		JSR	PC,INRDY		:WAIT FOR INRDY
4350	026660	104414			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

4351	026662	021244			021244		
4352	026664	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
4353	026670	042704	000374		BIC	#374,R4	;CLEAR UNWANTED BITS
4354	026674	012705	000003		MOV	#3,R5	;PUT "EXPECTED" IN R5
4355	026700	120504			CMPB	R5,R4	;ARE IN BCC MATCH AND BLOCK END SET?
4356	026702	001401			BEQ	50\$	
4357	026704	104042			HLT	42	;IN BCC MATCH ERROR
4358	026706			50\$:			
4359	026706	104414			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4360	026710	021204			021204		;GET LAST HALF
4361	026712	116137	000004	001251	MOVB	4(R1),TEMP2+1	;PUT IN TEMP2
4362	026720	042737	000377	001250	BIC	#377,TEMP2	;CLEAR LO BYTE
4363	026726	053737	001250	001252	BIS	TEMP2,TEMP3	;16 BIT BCC NOW IN TEMP3
4364	026734	023737	033032	001252	CMP	CALBCC,TEMP3	;IS IT CORRECT?
4365	026742	001401			BEQ	42\$	;BR IF OK
4366	026744	104027			HLT	27	
4367							
4368							
4369							;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
4370							;WAS RECEIVED CORRECTLY (0,125,252,377)
4371	026746	012703	000004		42\$:	MOV	#4,R3
4372	026752	012702	033614				;R3=CHARACTER COUNT
4373	026756	004737	032652		43\$:	MOV	#MESDAT,R2
4374	026762	104414				JSR	PC,INRDY
4375	026764	021204					;LOAD MESSAGE POINTER IN R2
4376	026766	016104	000004			ROMCLK	;WAIT FOR INRDY
4377	026772	112205				021204	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4378	026774	120504				MOV	4(R1),R4
4379	026776	001401				MOVB	(R2)+,R5
4380	027000	104010				CMPB	R5,R4
4381	027002	005303				BEQ	44\$
4382	027004	001364				HLT	10
4383					44\$:	DEC	R3
4384						BNE	43\$
4385							;R3=CHARACTER COUNT
4386							;LOAD MESSAGE POINTER IN R2
4387	027006	004737	032652				;WAIT FOR INRDY
4388	027012	104414				ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4389	027014	021204				021204	;GET FIRST HALF OF CRC
4390	027016	116137	000004	001252		MOVB	4(R1),TEMP3
4391	027024	042737	177400	001252		BIC	#177400,TEMP3
4392	027032	004737	032652			JSR	PC,INRDY
4393	027036	104414				ROMCLK	;WAIT FOR INRDY
4394	027040	021244				021244	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4395	027042	016104	000004			MOV	4(R1),R4
4396	027046	042704	000374			BIC	#374,R4
4397	027052	012705	000003			MOV	#3,R5
4398	027056	120504				CMPB	R5,R4
4399	027060	001401				BEQ	51\$
4400	027062	104042				HLT	42
4401	027064						;IN BCC MATCH ERROR
4402	027064	104414			51\$:	ROMCLK	;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4403	027066	021204				021204	;GET LAST HALF
4404	027070	116137	000004	001251		MOVB	4(R1),TEMP2+1
4405	027076	042737	000377	001250		BIC	#377,TEMP2
4406	027104	053737	001250	001252		BIS	TEMP2,TEMP3

```

4407 027112 023737 033032 001252      CMP      CALBCC,TEMP3      ;IS IT CORRECT?
4408 027120 001401                      BEQ      55              ;BR IF OK
4409 027122 104027                      HLT      27
4410 027124 104400                      55:      SCOPE              ;SCOPE THIS TEST
4411
4412
4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427 027126 012737 000056 001226      TST56:  MOV      #56,TSTNO
4428 027134 012737 030606 001216      MOV      #TST57,NEXT
4429
4430 027142 104412                      MSTCLR
4431 027144 005061 000004                      CLR      4(R1)
4432 027150 104414                      ROMCLK
4433 027152 122117 122117                      122117
4434 027154 004737 033374                      JSR      PC,CLRIO
4435 027160 005037 033612                      CLR      BITCON
4436
4437
4438
4439 027164 012711 004000                      ;LOAD OUT DATA SILO
4440 027170 012704 033614                      MOV      #BIT11,(R1)
4441 027174 005037 027330                      MOV      #MESDAT,R4
4442 027200 005137 027330                      CLR      10$
4443 027204 012700 000004                      COM      10$
4444 027210 004737 033176                      MOV      #4,R0
4445 027214 004737 032176                      JSR      PC,SYNLD
4446 027220 004537 033332                      JSR      PC,OUTRDY
4447 027224 033614                      JSR      R5,MESLD
4448 027226 000004                      MESDAT
4449 027230 004737 033306                      4
4450 027234 004737 033306                      JSR      PC,EOM
4451 027240 004737 033256                      JSR      PC,EOM
4452 027244 004537 033332                      JSR      PC,SOM
4453 027250 033614                      JSR      R5,MESLD
4454 027252 000004                      MESDAT
4455 027254 004737 033306                      4
4456 027260 004737 033306                      JSR      PC,EOM
4457 027264 004737 032044                      JSR      PC,EOM
4458 027270 005003                      JSR      PC,OCOR
4459 027272 104415 000022                      CLR      R3
4460 027276 112405                      DATA CLK,22
4461 027300 010502                      12$:  MOV      (R4)+,R5
4462                                MOV      R5,R2

```

***** TEST 56 *****

*BITSTUFF EOM FUNCTION TEST

*THIS TEST LOADS OUT SILO WITH: 2 FLAGS, 4 CHAR MESSAGE, EOM

*SOM, 4 CHAR MESS, EOM. THE DATA STREAM IS CHECKED TO BE

*4 CHAR BCC FLAG, 4 CHAR BCC FLAG 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 56

MOV #56,TSTNO

MOV #TST57,NEXT

R1 CONTAINS BASE DMC11 ADDRESS

MASTER CLEAR DMC11

CLEAR PORT4

NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

PUT LINE UNIT IN BITSTUFF MODE

DO THIS AFTER MODE IS SET

CONSECUTIVE 1'S COUNTER INIT TO 3

SET LINE UNIT LOOP

LOAD POINTER TO DATA

CLEAR SOFT BCC

START AT -1

LOAD CHARACTER COUNT

LOAD 2 FLAG CHARACTERS 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

SET EOM

WAIT FOR OCOR

CLEAR BIT COUNTER

CLOCK DATA

LOAD R5 WITH CHAR

LOAD R2 WITH CHAR

```

4463                                     :CHECK FIRST FOUR CHARACTER MESSAGE
4464                                     ;IN THE BIT WINDOW (0,125,252,377)
4465
4466 027302 010537 027376                MOV     R5,71$      ;LOAD FOR STUFF CHECK
4467 027306 012737 102010 033030        MOV     #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL
4468 027314 010537 027326                MOV     R5,67$      ;LOAD SOFT CHAR FOR BCC
4469 027320 004537 032706                JSR     R5,SIMBCC    ;CALCULATE SOFT BCC
4470 027324 000010                      IO             ;SHIFT COUNT
4471 027326 000000                      67$: 0             ;CHARACTER
4472 027330 000000                      10$: 0             ;OLD BCC
4473 027332 013737 033032 027330        MOV     CALBCC,10$   ;LOAD SOFT BCC FOR NEXT SHIFT
4474 027340 104415 000001                64$: DATACLK, 1   ;SHIFT DATA IN TO BIT WINDOW
4475 027344 106002                      RORB      R2         ;SHIFT SOFT DATA
4476 027346 103005                      BCC       65$        ;BR IF A SPACE
4477 027350 004737 032012                JSR     PC,GETSI    ;LOOK AT BIT WINDOW
4478 027354 103406                      BCS       66$        ;BR IF OK (MARK)
4479 027356 104006                      HLT       6          ;ERROR, BIT WINDOW WAS A SPACE
4480 027360 000404                      BR        66$        ;CONTINUE
4481 027362 004737 032012                65$: JSR     PC,GETSI ;LOOK AT BIT WINDOW
4482 027366 103001                      BCC       66$        ;BR IF OK (SPACE)
4483 027370 104006                      HLT       6          ;ERROR, BIT WINDOW WAS A MARK
4484 027372                                66$:
4485 027372 004537 033474                JSR     R5,STFFCK
4486 027376 000000                      71$: 0
4487 027400 000001                      1
4488 027402 110237 027376                MOVB     R2,71$      ;SHIFT FOR NEXT STUFF CHECK
4489 027406 005203                      INC       R3         ;BUMP BIT COUNTER
4490 027410 022703 000010                CMP      #10,R3     ;DONE FULL 8 BITS YET
4491 027414 001351                      BNE      64$        ;BR IF NO
4492 027416 005003                      CLR       R3         ;CLEAR BIT COUNTER
4493 027420 005300                      DEC       R0         ;DEC CHARACTER COUNT
4494 027422 001325                      BNE      12$        ;BR IF NOT DONE YET
4495
4496                                     :CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4497
4498 027424 005137 033032                COM       CALBCC      ;ADJUST BCC FOR SDLC
4499 027430 013700 033032                MOV     CALBCC,R0   ;PUT BCC IN R0
4500 027434 010037 027476                MOV     R0,72$      ;LOAD BCC FOR STUFF CHECK
4501 027440 104415 000001                68$: DATACLK, 1   ;SHIFT HARDWARE BCC
4502 027444 006000                      ROR       R0         ;SHIFT SOFT BCC
4503 027446 103005                      BCC       69$        ;BR IF CARRY CLEAR
4504 027450 004737 032012                JSR     PC,GETSI    ;LOOK AT BIT WINDOW
4505 027454 103406                      BCS       70$        ;BR IF OK (MARK)
4506 027456 104014                      HLT       14        ;ERROR, CRC WRONG (SPACE)
4507 027460 000404                      BR        70$        ;CONTINUE
4508 027462 004737 032012                69$: JSR     PC,GETSI ;LOOK AT BIT WINDOW
4509 027466 103001                      BCC       70$        ;BR IF OK (SPACE)
4510 027470 104014                      HLT       14        ;ERROR, CRC WRONG (MARK)
4511 027472                                70$:
4512 027472 004537 033474                JSR     R5,STFFCK
4513 027476 000000                      72$: 0
4514 027500 000001                      1
4515 027502 010037 027476                MOV     R0,72$      ;CHECK BCC CHAR FOR ZERO STUFFS
4516 027506 005203                      INC       R3         ;CHARACTER
4517 027510 022703 000020                CMP      #20,R3     ;SHIFT COUNT
4518 027514 001351                      BNE      68$        ;SHIFT SOFTBCC ONCE
                                     ;BUMP BIT COUNTER
                                     ;FINISHED BCC YET?
                                     ;BR IF NO

```

4519	027516	005003		CLR	R3	:CLEAR BIT COUNTER
4520						
4521						:CHECK FOR FLAG TO FOLLOW BCC
4522						
4523	027520	012737	000176 001252	73:	MOV	#18(01111110),TEMP3 :PUT FLAG CHARACTER IN TEMP3
4524	027526	104415	000001		DATACLK,	1 :CLOCK FLAG ONCE
4525	027532	106037	001252		RORB	TEMP3 :SHIFT SOFT FLAG
4526	027536	103405			BCS	74\$:BR IF BIT IS MARK
4527	027540	004737	032012		JSR	PC,GETSI :LOOK AT BIT WINDOW
4528	027544	103006			BCC	75\$:BR IF OK
4529	027546	104026			HLT	26 :ERROR IN FLAG CHAR
4530	027550	000404			BR	75\$
4531	027552	004737	032012	74:	JSR	PC,GETSI :LOOK AT BIT WINDOW
4532	027556	103401			BCS	75\$:BR IF OK
4533	027560	104026			HLT	26 :ERROR IN FLAG CHAR
4534	027562	005203		75:	INC	R3 :INC BIT COUNT
4535	027564	022703	000010		CMP	#10,R3 :FLAG DONE YET?
4536	027570	001356			BNE	73\$:BR IF NO
4537	027572	005003			CLR	R3 :CLEAR BIT COUNT
4538						
4539						:CHECK FOR ANOTHER FLAG CAUSED BY THE SOM
4540						
4541	027574	012737	000176 001252	76:	MOV	#18(01111110),TEMP3 :PUT FLAG CHARACTER IN TEMP3
4542	027602	104415	000001		DATACLK,	1 :CLOCK FLAG ONCE
4543	027606	106037	001252		RORB	TEMP3 :SHIFT SOFT FLAG
4544	027612	103405			BCS	77\$:BR IF BIT IS MARK
4545	027614	004737	032012		JSR	PC,GETSI :LOOK AT BIT WINDOW
4546	027620	103006			BCC	78\$:BR IF OK
4547	027622	104026			HLT	26 :ERROR IN FLAG CHAR
4548	027624	000404			BR	78\$
4549	027626	004737	032012	77:	JSR	PC,GETSI :LOOK AT BIT WINDOW
4550	027632	103401			BCS	78\$:BR IF OK
4551	027634	104026			HLT	26 :ERROR IN FLAG CHAR
4552	027636	005203		78:	INC	R3 :INC BIT COUNT
4553	027640	022703	000010		CMP	#10,R3 :FLAG DONE YET?
4554	027644	001356			BNE	76\$:BR IF NO
4555	027646	005003			CLR	R3 :CLEAR BIT COUNT
4556	027650	012700	000004		MOV	#4,R0 :RESL CHARACTER COUNTER
4557	027654	012704	033614		MOV	#MESDAT,R4 :LOAD MESSAGE POINTER
4558	027660	005037	027722		CLR	11\$:CLR SOFT BCC
4559	027664	005137	027722		COM	11\$:ADJUST TO -1 FOR SDLC
4560	027670	112405		13:	MOVB	(R4)+,R5 :LOAD CHAR IN R5
4561	027672	010502			MOV	R5,R2 :LOAD CHAR IN R2
4562						
4563						:CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
4564						
4565	027674	010537	027770		MOV	R5,86\$:LOAD FOR STUFF CHECK
4566	027700	012737	102010 033030		MOV	#CRC.CCITT,XPOLY :LOAD POLYNOMIAL
4567	027706	010537	027720		MOV	R5,82\$:LOAD SOFT CHAR FOR BCC
4568	027712	004537	032706		JSR	R5,SIMBCC :CALCULATE SOFT BCC
4569	027716	000010			10	:SHIFT COUNT
4570	027720	000000		82:	0	:CHARACTER
4571	027722	000000		11:	0	:OLD BCC
4572	027724	013737	033032 027722		MOV	CALBCC,11\$:LOAD SOFT BCC FOR NEXT SHIFT
4573	027732	104415	000001	79:	DATACLK,	1 :SHIFT DATA IN TO BIT WINDOW
4574	027736	106002			RORB	R2 :SHIFT SOFT DATA

4575	027740	103005		BCC	80\$	;BR IF A SPACE
4576	027742	004737	032012	JSR	PC,GETSI	;LOOK AT BIT WINDOW
4577	027746	103406		BCS	81\$	;BR IF OK (MARK)
4578	027750	104006		HLT	6	;ERROR, BIT WINDOW WAS A SPACE
4579	027752	000404		BR	81\$	;CONTINUE
4580	027754	004737	032012	80\$: JSR	PC,GETSI	;LOOK AT BIT WINDOW
4581	027760	103001		BCC	81\$	;BR IF OK (SPACE)
4582	027762	104006		HLT	6	;ERROR, BIT WINDOW WAS A MARK
4583	027764			81\$: JSR	RS,STFFCK	
4584	027764	004537	033474	86\$: 0		
4585	027770	000000		1		
4586	027772	000001		MOV8	R2,86\$	;SHIFT FOR NEXT STUFF CHECK
4587	027774	110237	027770	INC	R3	;BUMP BIT COUNTER
4588	030000	005203		CMP	#10,R3	;DONE FULL 8 BITS YET
4589	030002	022703	000010	BNE	79\$	;BR IF NO
4590	030006	001351		CLR	R3	;CLEAR BIT COUNTER
4591	030010	005003		DEC	RO	;DEC CHARACTER COUNT
4592	030012	005300		BNE	13\$	;BR IF NOT DONE YET
4593	030014	001325				
4594						
4595						;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4596						
4597	030016	005137	033032	COM	CALBCC	;ADJUST BCC FOR SOLC
4598	030022	0137C0	033032	MOV	CALBCC,RO	;PUT BCC IN RO
4599	030026	010037	030070	MOV	RO,87\$	;LOAD BCC FOR STUFF CHECK
4600	030032	104415	000001	93\$: DATACLK,1		;SHIFT HARDWARE BCC
4601	030036	006000		ROR	RO	;SHIFT SOFT BCC
4602	030040	103005		BCC	84\$	;BR IF CARRY CLEAR
4603	030042	004737	032012	JSR	PC,GETSI	;LOOK AT BIT WINDOW
4604	030046	103406		BCS	85\$	;BR IF OK (MARK)
4605	030050	104014		HLT	14	;ERROR, CRC WRONG (SPACE)
4606	030052	000404		BR	85\$	;CONTINUE
4607	030054	004737	032012	84\$: JSR	PC,GETSI	;LOOK AT BIT WINDOW
4608	030060	103001		BCC	85\$	;BR IF OK (SPACE)
4609	030062	104014		HLT	14	;ERROR, CRC WRONG (MARK)
4610	030064			85\$: JSR	RS,STFFCK	;CHECK BCC CHAR FOR ZERO STUFFS
4611	030064	004537	033474	87\$: 0		;CHARACTER
4612	030070	000000		1		;SHIFT COUNT
4613	030072	000001		MOV	RO,87\$	;SHIFT SOFTBCC ONCE
4614	030074	010037	030070	INC	R3	;BUMP BIT COUNTER
4615	030100	005203		CMP	#20,R3	;FINISHED BCC YET?
4616	030102	022703	000020	BNE	83\$	;BR IF NO
4617	030106	001351		CLR	R3	;CLEAR BIT COUNTER
4618	030110	005003				
4619						
4620						;CHECK FOR FLAG TO FOLLOW BCC
4621						
4622	030112	012737	000176	MOV	#18<01111110>,TEMP3	;PUT FLAG CHARACTER IN TEMP3
4623	030120	104415	000001	88\$: DATACLK,1		;CLOCK FLAG ONCE
4624	030124	106037	001252	RORB	TEMP3	;SHIFT SOFT FLAG
4625	030130	103405		BCS	89\$	;BR IF BIT IS MARK
4626	030132	004737	032012	JSR	PC,GETSI	;LOOK AT BIT WINDOW
4627	030136	103006		BCC	90\$	;BR IF OK
4628	030140	104026		HLT	26	;ERROR IN FLAG CHAR
4629	030142	000404		BR	90\$	
4630	030144	004737	032012	89\$: JSR	PC,GETSI	;LOOK AT BIT WINDOW

4631	030150	103401			BCS	90\$		;BR IF OK
4632	030152	104026			HLT	26		;ERROR IN FLAG CHAR
4633	030154	005203		90\$	INC	R3		;INC BIT COUNT
4634	030156	022703	000010		CMP	#10,R3		;FLAG DONE YET?
4635	030162	001356			BNE	88\$		;BR IF NO
4636	030164	005003			CLR	R3		;CLEAR BIT COUNT
4637								
4638								;CHECK TO SEE IF TRANSMITTER IS MARKING
4639								
4640	030166	104415	000001	2\$:	DATACLK,	1		;CLOCK TRANSMITTER
4641	030172	004737	032012		JSR	PC,GETSI		;LOOK AT WINDOW
4642	030176	103401			BCS	3\$		;IT SHOULD BE MARKING
4643	030200	104024			HLT	24		;ERROR, BIT WAS A SPACE
4644	030202	005203		3\$:	INC	R3		;BUMP BIT COUNTER
4645	030204	022703	000007		CMP	#7,R3		;DONE YET
4646	030210	001366			BNE	2\$		;BR IF NO
4647	030212	104415	000010		DATACLK,	10		;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
4648	030216	005003			CLR	R3		;CLEAR BIT COUNTER
4649	030220	104415	000001	4\$:	DATACLK,	1		;SHIFT OUT NEXT BIT
4650	030224	004737	032012		JSR	PC,GETSI		;LOOK AT BIT WINDOW
4651	030230	103401			BCS	+4		;BR IF IT IS A MARK
4652	030232	104024			HLT	24		;ERROR, TRANSMITTER IS NOT MARKING
4653	030234	005203			INC	R3		;INC BIT COUNT
4654	030236	022703	000020		CMP	#20,R3		;DONE YET?
4655	030242	001366			END	4\$		;BR IF NO
4656								
4657								;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
4658								;WAS RECEIVED CORRECTLY (0,125,252,377)
4659								
4660	030244	104415	000001		DATACLK,	1		;GET LAST BIT IN RECEIVER
4661	030250	012703	000004		MOV	#4,R3		;R3=CHARACTER COUNT
4662	030254	012702	033614		MOV	#MESDAT,R2		;LOAD MESSAGE POINTER IN R2
4663	030260	004737	032652	40\$:	JSR	PC,INRDY		;WAIT FOR INRDY
4664	030264	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4665	030266	021204			021204			
4666	030270	016104	000004		MOV	4(R1),R4		;PUT "FOUND" IN R4
4667	030274	112205			MOVB	(R2)+,R5		;PUT "EXPECTED" IN R5
4668	030276	120504			CMPB	R5,R4		;IS RECEIVED DATA CORRECT?
4669	030300	001401			BEQ	41\$		;BR IF YES
4670	030302	104010			HLT	10		;RECEIVE DATA ERROR
4671	030304	005303		41\$:	DEC	R3		;DEC CHARACTER COUNT
4672	030306	001364			BNE	40\$		;BR IF NOT DONE YET
4673								
4674								;CHECK TO SEE THAT IN BCC MATCH IS SET
4675								;AND THAT THE BCC WAS RECEIVED CORRECTLY
4676								
4677	030310	004737	032652		JSR	PC,INRDY		;WAIT FOR INRDY
4678	030314	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4679	030316	021204			021204			;GET FIRST HALF OF CRC
4680	030320	116137	000004	001252	MOVB	4(R1),TEMP3		;PUT IN TEMP3
4681	030326	042737	177400	001252	BIC	#177400,TEMP3		;CLEAR HI BYTE
4682	030334	004737	032652		JSR	PC,INRDY		;WAIT FOR INRDY
4683	030340	104414			ROMCLK			;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4684	030342	021244			021244			
4685	030344	016104	000004		MOV	4(R1),R4		;PUT "FOUND" IN R4
4686	030350	042704	000374		BIC	#374,R4		;CLEAR UNWANTED BITS

```

4687 030354 012705 000003      MOV      #3,R5      ;PUT "EXPECTED" IN R5
4688 030360 120504      CMPB     R5,R4      ;ARE IN BCC MATCH AND BLOCK END SET?
4689 030362 001401      BEQ      50$      ;IN BCC MATCH ERROR
4690 030364 104042      HLT      42
4691 030366      50$:
4692 030366 104414      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4693 030370 021204      021204    ;GET LAST HALF
4694 030372 116137 000004 001251  MOVB     4(R1),TEMP2+1 ;PUT IN TEMP2
4695 030400 042737 000377 001250  BIC      #377,TEMP2    ;CLEAR LO BYTE
4696 030406 053737 001250 001252  BIS      TEMP2,TEMP3    ;16 BIT BCC NOW IN TEMP3
4697 030414 023737 033032 001252  CMP      CALBCC,TEMP3   ;IS IT CORRECT?
4698 030422 001401      BEQ      42$      ;BR IF OK
4699 030424 104027      HLT      27
4700
4701      ;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
4702      ;WAS RECEIVED CORRECTLY (0,125,252,377)
4703
4704 030426 012733 000004      42$: MOV      #4,R3      ;R3=CHARACTER COUNT
4705 030432 012732 033614      MOV      #MESDAT,R2    ;LOAD MESSAGE POINTER IN R2
4706 030436 004737 032652      43$: JSR      PC,INRDY    ;WAIT FOR INRDY
4707 030442 104414      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4708 030444 021204      021204
4709 030446 016104 000004      MOV      4(R1),R4      ;PUT "FOUND" IN R4
4710 030452 112205      MOVB     (R2)+,R5    ;PUT "EXPECTED" IN R5
4711 030454 120504      CMPB     R5,R4      ;IS RECEIVED DATA CORRECT?
4712 030456 001401      BEQ      44$      ;BR IF YES
4713 030460 104010      HLT      10      ;RECEIVE DATA ERROR
4714 030462 005303      44$: DEC      R3      ;DEC CHARACTER COUNT
4715 030464 001364      BNE      43$      ;BR IF NOT DONE YET
4716
4717      ;CHECK TO SEE THAT IN BCC MATCH IS SET
4718      ;AND THAT THE BCC WAS RECEIVED CORRECTLY
4719
4720 030466 004737 032652      JSR      PC,INRDY    ;WAIT FOR INRDY
4721 030472 104414      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4722 030474 021204      021204    ;GET FIRST HALF OF CRC
4723 030476 116137 000004 001252  MOVB     4(R1),TEMP3    ;PUT IN TEMP3
4724 030504 042737 177400 001252  BIC      #177400,TEMP3 ;CLEAR HI BYTE
4725 030512 004737 032652      JSR      PC,INRDY    ;WAIT FOR INRDY
4726 030516 104414      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4727 030520 021244      021244
4728 030522 016104 000004      MOV      4(R1),R4      ;PUT "FOUND" IN R4
4729 030526 042704 000374      BIC      #374,R4      ;CLEAR UNWANTED BITS
4730 030532 012705 000003      MOV      #3,R5      ;PUT "EXPECTED" IN R5
4731 030536 120504      CMPB     R5,R4      ;ARE IN BCC MATCH AND BLOCK END SET?
4732 030540 001401      BEQ      51$      ;IN BCC MATCH ERROR
4733 030542 104042      HLT      42
4734 030544      51$:
4735 030544 104414      ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4736 030546 021204      021204    ;GET LAST HALF
4737 030550 116137 000004 001251  MOVB     4(R1),TEMP2+1 ;PUT IN TEMP2
4738 030556 042737 000377 001250  BIC      #377,TEMP2    ;CLEAR LO BYTE
4739 030564 053737 001250 001252  BIS      TEMP2,TEMP3    ;16 BIT BCC NOW IN TEMP3
4740 030572 023737 033032 001252  CMP      CALBCC,TEMP3   ;IS IT CORRECT?
4741 030600 001401      BEQ      5$      ;BR IF OK
4742 030602 104027      HLT      27

```

```

4743 030604 104400          SS:  SCOPE          ;SCOPE THIS TEST
4744
4745
4746          ;***** TEST 57 *****
4747          ;*EMPTY SILO TEST
4748          ;*LOAD SILO WITH 2 SYNCs, 4 CHAR MESSAGE, SINGLE CLOCK
4749          ;*UNTIL THE SILO IS EMPTY, LOAD 4 MORE CHARACTERS IN THE
4750          ;*SILO. GIVE MORE TICKS, AND VERIFY THAT ONLY THE FIRST
4751          ;*4 CHARACTERS AND A BLOCK END WERE RECEIVED, AND IN ACTIVE IS CLEAR
4752          ;*****
4753
4754          : TEST 57
4755          :-----
4756 030606 012737 000057 001226      TST57: MOV      #57,TSTNO
4757 030614 012737 031026 001216      MOV      #TST60,NEXT
4758
4759 030622 104412                MSTCLR          ;R1 CONTAINS BASE DMC11 ADDRESS
4760 030624 005061 000004          CLR      4(R1)      ;MASTER CLEAR DMC11
4761 030630 104414                ROMCLK          ;CLEAR PORT4
4762 030632 122117                122117          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=53C4
4763 030634 004737 033374          JSR      PC,CLR10    ;PUT LU IN BITSTUFF MODE
4764 030640 012711 004000          MOV      #BIT11,(R1) ;DO THIS AFTER MODE IS SET
4765 030644 012702 033614          MOV      #MESDAT,R2  ;SET LINE UNIT LOOP
4766 030650 012700 000003          MOV      #3,R0       ;R2 POINTS TO MESSAGE
4767 030654 004737 033176          JSR      PC,SYNLD    ;R0 = CHAR COUNT
4768 030660 004737 032176          JSR      PC,OUTRDY   ;LOAD SILO WITH TWO FLAGS
4769 030664 004537 033332          JSR      R5,MESLD    ;WAIT FOR OUTRDY
4770 030670 033614                MESDAT          ;LOAD MESSAGE IN SILO
4771 030672 000004                4              ;START OF MESSAGE
4772 030674 004737 032044          JSR      PC,OCOR      ;CHARACTER COUNT
4773 030700 104415 000065          DATACLK,        65  ;WAIT FOR OCOR
4774 030704 004537 033332          JSR      R5,MESLD    ;CLOCK DATA (EMPTY SILO)
4775 030710 033614                MESDAT          ;PUT MORE CHARACTERS IN SILO
4776 030712 000004                4
4777 030714 004737 032044          JSR      PC,OCOR
4778 030720 104415 000006          DATACLK,        6    ;CLOCK UNTIL RTS IS CLEARED
4779 030724 104414                ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=53C4
4780 030726 021264                021264          ;GET RTS
4781 030730 032761 000040 000004  BIT      #BIT5,4,(R1) ;IS IT CLEAR?
4782 030736 001401                BEQ      5$          ;BR IF YES
4783 030740 104034                HLT      34          ;ERROR, RTS NOT CLEAR
4784 030742 104415 000041          DATACLK,        41  ;CLOCK XMITTER SOME MORE
4785 030746 004737 032652          JSR      PC,INRDY   ;OK LETS CHECK WHAT WAS RECEIVED
4786 030752 104414                ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=53C4
4787 030754 021204                021204          ;GET RECEIVE DATA
4788 030756 016104 000004          MOV      4(R1),R4    ;PUT IT IN R4
4789 030762 112205                MOV      (R2)+,R5    ;R5 = "EXPECTED"
4790 030764 120504                CMP      R5,R4       ;IS DATA CORRECT?
4791 030766 001401                BEQ      2$          ;BR IF OK
4792 030770 104010                HLT      10          ;DATA ERROR
4793 030772 005300                DEC      R0         ;DEC CHAR COUNT
4794 030774 001364                BNE      1$          ;BR IF NOT DONE YET
4795 030776 004737 032652          JSR      PC,INRDY   ;WAIT FOR INRDY
4796 031002 104414                ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=53C4
4797 031004 021244                021244          ;READ LU-12
4798 031006 016104 000004          MOV      4(R1),R4    ;PUT "FOUND" IN R4

```

```

4799 031012 012705 000022      MOV      #22,R5      ;PUT "EXPECTED" IN R5
4800 031016 120504      CMPB     R5,R4      ;ARE BLOCK END AND IN RDY SET?
4801                                ;AND IN ACTIVE AND IN BCC MATCH CLEAR?
4802 031020 001401      BEQ       6$      ;BR IF YES
4803 031022 104032      HLT        32      ;ERROR, BLOCK END NOT SET
4804                                ;OR IN BCC MATCH NOT CLEAR
4805                                ;OR IN ACTIVE NOT CLEAR
4806 031024      6$:      SCOPE      ;SCOPE THIS TEST
4807 031024 104400
4808
4809
4810
4811      ;***** TEST 60 *****
4812      ;*BITSTUFF CABLE DATA TEST
4813      ;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
4814      ;*2 FLAGS, 16 CHAR EOM, 16 CHAR EOM, 16 CHAR EOM
4815      ;*THE 16 CHARACTERS INCLUDE A FLOATING ONE AND ZERO
4816      ;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOC
4817      ;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
4818      ;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
4819      ;*****
4820      ; TEST 60
4821      ;-----
4822 031026 012737 000060 301226  *ST60: MOV      #60,TSTNO
4823 031034 012737 031450 001216  MOV      #TST61,NEXT
4824
4825 031042 104412      MSTCLR      ;R1 CONTAINS BASE DMC11 ADDRESS
4826 031044 032737 040000 001366  BIT      #BIT14,STAT1 ;MASTER CLEAR DMC11
4827 031052 001575      BEQ       3$      ;SKIP TEST IF NO
4828 031054 005061 000004      CLR      4(R1) ;LOOPBACK CONNECTOR ON
4829 031060 104414      ROMCLK      ;CLEAR PORT4
4830 031062 122117 122117      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4831 031064 004737 033374      JSR      PC,CLR10 ;PUT LINE UNIT IN BITSTUFF MODE
4832 031070 012711 004000      MOV      #BIT11,(R1) ;DO THIS AFTER MODE IS SET
4833 031074 004737 033176      JSR      PC,SYNLD ;SET LINE UNIT LOOP
4834 031100 012737 102010 033030  MOV      #CRC.CCITT,XPOLY ;LOAD TWO FLAGS
4835 031106 005037 031142      CLR      6$      ;LOAD POLYNOMIAL FOR SOFT CRC CALC
4836 031112 005137 031142      COM      6$      ;CLEAR OLD BCC
4837 031116 012703 000020      MOV      #16,R3 ;ADJUST TO -1 FOR SLOC
4838 031122 012702 033620      MOV      #FLT0AT,R2 ;CHARACTER COUNT
4839 031126 112237 031140      MOVB     (R2)+,5$ ;R2= POINTER
4840 031132 004537 032706      JSR      R5,SIMBCC ;LOAD CHAR FOR SOFT BCC CALC.
4841 031136 000010      LD         10 ;CALC SOFT BCC
4842 031140 000000      5$:      0 ;SHIFT COUNT
4843 031142 000000      6$:      0 ;CHARACTER
4844 031144 013737 033032 031142  MOV      CALBCC,6$ ;OLD BCC
4845 031152 005303      DEC      R3 ;LOAD OLD BCC
4846 031154 001364      BNE      7$      ;DEC COUNT
4847 031156 005137 033032      COM      CALBCC ;BR IF NOT DONE YET
4848 031162 004537 033332      JSR      R5,MESLD ;ADJUST CALBCC FOR SLOC
4849 031166 033620      FLT0AT ;LOAD SILO
4850 031170 000020      16. ;MESSAGE ADDRESS
4851 031172 004737 033306      JSR      PC,EOM ;CHARACTER COUNT
4852 031176 004737 033306      JSR      PC,EOM ;LOAD AN EOM
4853 031202 004537 033332      JSR      R5,MESLD ;LOAD SILO
4854 031206 033620      FLT0AT ;MESSAGE ADDRESS

```

4855	031210	000020		16.		: CHARACTER COUNT
4856	031212	004737	033306	JSR	PC,EOM	: LOAD AN EOM
4857	031216	004737	033306	JSR	PC,EOM	
4858	031222	004537	033332	JSR	R5,MESLD	: LOAD SILO
4859	031226	033620		FLTDAT		: MESSAGE ADDRESS
4860	031230	000020		16.		: CHARACTER COUNT
4861	031232	004737	033306	JSR	PC,EOM	: LOAD AN EOM
4862	031236	004737	033306	JSR	PC,EOM	
4863	031242	004737	032044	JSR	PC,OCOR	: WAIT FOR OCOR
4864	031246	005011		CLR	(R1)	: CLEAR LINE UNIT LOOP
4865	031250	012700	000003	MOV	#3,R0	: R0 = MESSAGE COUNT
4866	031254	012703	000020	MOV	#16.,R3	: R3= CHARACTER COUNT
4867	031260	012702	033620	MOV	#FLTDAT,R2	: LOAD MESSAGE POINTER IN R2
4868	031264	004737	032652	JSR	PC,INRDY	: WAIT FOR INRDY
4869	031270	104414		ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4870	031272	021204		021204		: GET DATA FROM IN SILO
4871	031274	016104	000004	MOV	4(R1),R4	: PUT CHARACTER IN "FOUND"
4872	031300	112205		MOVB	(R2)+,R5	: PUT "EXPECTED" IN R5
4873	031302	120504		CMPB	R5,R4	: IS RECEIVED DATA CORRECT
4874	031304	001401		BEQ	2\$	: BR IF OK
4875	031306	104025		HLT	25	: DATA ERROR
4876	031310					
4877	031310	005303		DEC	R3	: DEC CHARACTER COUNT
4878	031312	001364		BNE	1\$	: BR IF NOT DONE THIS MESSAGE
4879	031314	012703	000020	MOV	#16.,R3	: RESET CHARACTER COUNT
4880						
4881						: CHECK TO SEE THAT IN BCC MATCH IS SET
4882						: AND THAT THE BCC WAS RECEIVED CORRECTLY
4883						
4884	031320	004737	032652	JSR	PC,INRDY	: WAIT FOR INRDY
4885	031324	104414		ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4886	031326	021204		021204		: GET FIRST HALF OF CRC
4887	031330	116137	000004	MOVB	4(R1),TEMP3	: PUT IN TEMP3
4888	031336	042737	177400	BIC	#177400,TEMP3	: CLEAR HI BYTE
4889	031344	004737	032652	JSR	PC,INRDY	: WAIT FOR INRDY
4890	031350	104414		ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4891	031352	021244		021244		
4892	031354	016104	000004	MOV	4(R1),R4	: PUT "FOUND" IN R4
4893	031360	042704	000374	BIC	#374,R4	: CLEAR UNWANTED BITS
4894	031364	012705	000003	MOV	#3,R5	: PUT "EXPECTED" IN R5
4895	031370	120504		CMPB	R5,R4	: ARE IN BCC MATCH AND BLOCK END SET?
4896	031372	001401		BEQ	25\$	
4897	031374	104042		HLT	42	: IN BCC MATCH ERROR
4898	031376					
4899	031376	104414		ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4900	031400	021204		021204		: GET LAST HALF
4901	031402	116137	000004	MOVB	4(R1),TEMP2+1	: PUT IN TEMP2
4902	031410	042737	000377	BIC	#377,TEMP2	: CLEAR LO BYTE
4903	031416	053737	001250	BIS	TEMP2,TEMP3	: 16 BIT BCC NOW IN TEMP3
4904	031424	023737	033032	CMP	CALBCC,TEMP3	: IS IT CORRECT?
4905	031432	001401		BEQ	4\$	: BR IF OK
4906	031434	104027		HLT	27	
4907	031436	012702	033620	MOV	#FLTDAT,R2	: RESET MESSAGE POINTER
4908	031442	005300		DEC	R0	: DECREMENT COUNTER
4909	031444	001307		BNE	1\$	: BR IF NOT DONE
4910	031446	104400		SCOPE		: SCOPE THIS TEST

```

4911
4912
4913
4914
4915
4916
4917
4918
4919
4920
4921
4922
4923
4924 031450 012737 000061 001226
4925 031456 012737 003364 001216
4926
4927 031464 104412
4928 031466 032737 040000 001366
4929 031474 001545
4930 031476 005061 000004
4931 031502 104414
4932 031504 122117
4933 031506 004737 033374
4934 031512 012711 004000
4935 031516 004737 033176
4936 031522 012737 102010 03303C
4937 031530 005037 031564
4938 031534 005137 031564
4939 031540 012703 000073
4940 031544 012702 033614
4941 031550 112237 031562
4942 031554 004537 032706
4943 031560 000010
4944 031562 000000
4945 031564 000000
4946 031566 013737 033032 031564
4947 031574 005303
4948 031576 001364
4949 031600 005137 033032
4950 031604 004537 033332
4951 031610 033614
4952 031612 000073
4953 031614 004737 033306
4954 031620 004737 033306
4955 031624 004737 032044
4956 031630 005011
4957 031632 012700 000073
4958 031636 012702 033614
4959 031642 004737 032652
4960 031646 104414
4961 031650 021204
4962 031652 016104 000004
4963 031656 112205
4964 031660 120504
4965 031662 001401
4966 031664 104025

;*****-- TEST 61 *****
;BITSTUFF CABLE DATA TEST
;THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
;2 FLAGS, 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
;*****

; TEST 61
-----
TST61: MOV #61, TSTNO
MOV #.EOP, NEXT
MSTCLR
BIT #BIT14, STAT1
BEG 3$
CLR 4(R1)
ROMCLK
122117
JSR PC, CLRIO
MOV #BIT11, (R1)
JSR PC, SYNLD
MOV #CRC.CCITT, XPOLY
CLR 6$
COM 6$
MOV #59, R3
MOV #MESDAT, R2
7$: MOVB (R2)+, 5$
JSR R5, SIMBCC
10
5$: 0
6$: 0
MOV CALBCC, 6$
DEC R3
BNE 7$
COM CALBCC
JSR R5, MESLD
MESDAT
59
JSR PC, EOM
JSR PC, EOM
JSR PC, OCOR
CLR (R1)
MOV #59, R0
MOV #MESDAT, R2
1$: JSR PC, INRDY
ROMCLK
021204
MOV 4(R1), R4
MOVB (R2)+, R5
CMPB R5, R4
BEG 2$
HLT 2$

;R1 CONTAINS BASE DMC11 ADDRESS
;MASTER CLEAR DMC11
;SKIP TEST IF NO
;LOOPBACK CONNECTOR ON
;CLEAR PORT4
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PUT LINE UNIT IN BITSTUFF MODE
;DO THIS AFTER MODE IS SET
;SET LINE UNIT LOOP
;LOAD TWO FLAGS
;LOAD POLYNOMIAL FOR SOFT CRC CALC
;CLEAR OLD BCC
;ADJUST TO -1 FOR SDLC
;CHARACTER COUNT
;R2= POINTER
;LOAD CHAR FOR SOFT BCC CALC.
;CALC SOFT BCC
;SHIFT COUNT
;CHARACTER
;OLD BCC
;LOAD OLD BCC
;DEC COUNT
;BR IF NOT DONE YET
;ADJUST CALBCC FOR SDLC
;LOAD SILO
;MESSAGE ADDRESS
;CHARACTER COUNT
;LOAD AN EOM
;WAIT FOR OCOR
;CLEAR LINE UNIT LOOP
;R0= CHARACTER COUNT
;LOAD MESSAGE POINTER IN R2
;WAIT FOR INRDY
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;GET DATA FROM IN SILO
;PUT CHARACTER IN "FOUND"
;PUT "EXPECTED" IN R5
;IS RECEIVED DATA CORRECT
;BR IF OK
;DATA ERROR

```

```

4967 031666
4968 031666 005300
4969 031670 001364
4970
4971
4972
4973
4974 031672 004737 032652
4975 031676 104414
4976 031700 021204
4977 031702 116137 000004 001252
4978 031710 042737 177400 001252
4979 031716 004737 032652
4980 031722 104414
4981 031724 021244
4982 031726 016104 000004
4983 031732 042704 000374
4984 031736 012705 000003
4985 031742 120504
4986 031744 001401
4987 031746 104042
4988 031750
4989 031750 104414
4990 031752 021204
4991 031754 116137 000004 001251
4992 031762 042737 000377 001250
4993 031770 053737 001250 001252
4994 031776 023737 033032 001252
4995 032004 001401
4996 032006 104027
4997 032010 104400
4998
4999
5000
5001
5002
5003 032012
5004
5005
5006
5007 032012 104414
5008 032014 021364
5009 032016 017737 147370 032042 01300
5010 032024 106137 032042 01400
5011 032030 106137 032042 01500
5012 032034 106137 032042 01600
5013 032040 000207 01700
5014 032042 000000 01800
5015
5016
5017 032044
5018
5019
5020 032044 104414
5021 032046 021364
5022 032050 032777 000020 147334 02600 02700

```

25: DEC RO ; DECREMENT COUNTER
BNE 15 ; BR IF NOT DONE

; CHECK TO SEE THAT IN BCC MATCH IS SET
; AND THAT THE BCC WAS RECEIVED CORRECTLY

JSR PC, INRDY ; WAIT FOR INRDY
ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021204 ; GET FIRST HALF OF CRC
MOVB 4(R1), TEMP3 ; PUT IN TEMP3
BIC #177400, TEMP3 ; CLEAR HI BYTE
JSR PC, INRDY ; WAIT FOR INRDY
ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021244
MOV 4(R1), R4 ; PUT "FOUND" IN R4
BIC #374, R4 ; CLEAR UNWANTED BITS
MOV #3, R5 ; PUT "EXPECTED" IN R5
CMPB R5, R4 ; ARE IN BCC MATCH AND BLOCK END SET?
BEQ 25\$
HLT 42 ; IN BCC MATCH ERROR

25\$: ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021204 ; GET LAST HALF
MOVB 4(R1), TEMP2+1 ; PUT IN TEMP2
BIC #377, TEMP2 ; CLEAR LO BYTE
BIS TEMP2, TEMP3 ; 16 BIT BCC NOW IN TEMP3
CMP CALBCC, TEMP3 ; IS IT CORRECT?
BEQ 3\$; BR IF OK
HLT 27

3\$: SCOPE ; SCOPE THIS TEST

; SUBROUTINES
;-----
GETSI: ; THIS SUBROUTINE READS LU 17, AND PUTS IT INTO NITCH.
; NITCH IS ROTATED LEFT UNTILL THE SI BIT IS IN CARRY

ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021364 ; PORT4+LU 17
MOV @DMP04, NITCH ; STORE LU 17
ROLB NITCH
ROLB NITCH
ROLB NITCH ; PUT SI IN THE CARRY BIT
RTS PC

NITCH: 0

OCOR: ; THIS SUBROUTINE SPINS ON OCOR

ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021364 ; PORT4+LU 17
BIT #BIT4, @DMP04 ; IS OCOR SET?

5023	032056	001272		02800		REQ	OCOR	:BR IF NO
5024	032060	000207		02900		RTS	PC	:OK OCOR IS SET, GO BACK
5025				03000				
5026				03100				
5027	032062			03200				
5028				03300				
5029				03400				
5030				03500				
5031				03600				
5032	032062	013637	001246	03700		MOV	2(SP)+,TEMP1	:GET COUNT
5033	032066	062746	000002	03800		ADD	#2,-(SP)	:ADJUST STACK
5034	032072	012761	000026	03900	000004	MOV	#26,4(R1)	:LOAD PORT4
5035	032100	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5036	032102	122114		04100		122114		:LOAD SYNC REGISTER
5037	032104	004737	032176	04200		JSR	PC,OUTRDY	:WAIT FOR OUTRDY
5038	032110	012761	000001	04300	000004	MOV	#1,4(R1)	:LOAD PORT4
5039	032116	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5040	032120	122111		04500		122111		:SET SOM
5041	032122	012761	000026	04600	000004	MOV	#26,4(R1)	:LOAD PORT4
5042	032130	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5043	032132	122110		04800		122110		:LOAD OUT DATA
5044	032134	005337	001246	04900		DEC	TEMP1	:ALL DONE?
5045	032140	001361		05000		BNE	1\$	:BR IF NOT
5046	032142	004737	032176	05100		JSR	PC,OUTRDY	:WAIT FOR OUTRDY
5047	032146	005061	000004	05200		CLR	4(R1)	:LOAD PORT4
5048	032152	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5049	032154	122111		05400		122111		:SET SOM
5050	032156	012761	000301	05500	000004	MOV	#301,4(R1)	:LOAD PORT4
5051	032164	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5052	032166	122110		05700		122110		:LOAD OUT DATA
5053	032170	004737	032044	05800		JSR	PC,OCOR	:WAIT FOR OCOR
5054	032174	000207		05900		RTS	PC	
5055				06000				
5056				06100				
5057	032176			06200				
5058				06300				
5059				06400				
5060	032176	005037	001256	06500		CLR	TEMPS	:CLEAR TIMER
5061	032202			06600				
5062	032202	104414				ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5063	032204	021224		06800		021224		:PORT4+LUI1
5064	032206	032777	000020	06900	147176	BIT	#BIT4,DZDMF04	:IS OUT RDY SET?
5065	032214	001004		07000		BNE	2\$	:BR IF YES
5066	032216	005237	001256	07100		INC	TEMPS	:INC TIMER
5067	032222	001367		07200		BNE	1\$	:KEEP CHECKING IF NOT DONE
5068	032224	104036		07300		HLT	36	:ERROR, OUT READY NOT SET
5069	032226	000207		07400		2\$: RTS	PC	
5070				07500				
5071				07600				
5072	032230			07700				
5073				07800				
5074				07900				
5075				08000				
5076	032230	013637	001250	08100		MOV	2(SP)+,TEMP2	:GET CHARACTER
5077	032234	062746	000002	08200		ADD	#2,-(SP)	:ADJUST STACK
5078	032240	012737	000003	08300	001246	MOV	#3,TEMP1	:SET FOR 3 SYNC

SYNC:

;THIS SUBROUTINE LOADS THE SILO WITH THE NUMBER OF SYNC
 ;CHARACTERS PASSED TO IT IN THE WORD AFTER THE JSR CALL
 ;AND A NON-SYNC CHARACTER (301)

1\$:

OUTRDY:

;THIS SUBROUTINE SPINS ON OUT READY

1\$:

2\$:

CHAR:

;THIS SUBROUTINE LOADS THE SILO WITH 3 SYNC
 ;AND THE CHARACTER PASSED TO IT.

5079	032246	012761	000026	000004	08400
5080	032254	104414			
5081	032256	122114			08600
5082	032260	004737	032176		08700
5083	032264	012761	000001	000004	08800
5084	032272	104414			
5085	032274	122111			09000
5086	032276	012761	000026	000004	09100
5087	032304	104414			
5088	032306	122110			09300
5089	032310	005337	001246		09400
5090	032314	001361			09500
5091	032316	004737	032176		09600
5092	032322	013761	001250	000004	09700
5093	032330	104414			
5094	032332	122110			09900
5095	032334	004737	032044		10000
5096	032340	000207			10100
5097					10200
5098					10300
5099	032342				10400
5100					10500
5101					10600
5102	032342	013637	001250		10700
5103	032346	062746	000002		10800
5104	032352	004737	032176		10900
5105	032356	013761	001250	000004	11000
5106	032364	104414			
5107	032366	122110			11200
5108	032370	004737	032176		11300
5109	032374	104414			
5110	032376	122110			11500
5111	032400	004737	032044		11600
5112	032404	000207			11700
5113					11800
5114					11900
5115	032406				12000
5116					12100
5117					12200
5118					12300
5119	032406	012737	000073	001250	12400
5120	032414	005737	032646		12500
5121	032420	100470			12600
5122	032422	001032			12700
5123	032424	062737	000002	001250	12800
5124	032432	012737	000003	001246	12900
5125	032440	012761	000026	000004	13000
5126	032446	104414			
5127	032450	122114			13200
5128	032452	004737	032176		13300
5129	032456	012761	000001	000004	13400
5130	032464	104414			
5131	032466	122111			13600
5132	032470	012761	000026	000004	13700
5133	032476	104414			
5134	032500	122110			13900

1\$:

CHARSD:

SILOLD:

1\$:

MOV	#26,4(R1)	:LOAD PORT4
ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122114		:LOAD SYNC REGISTER
JSR	PC,OUTRDY	:WAIT FOR OUTRDY
MOV	#1,4(R1)	:LOAD PORT4
ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122111		:SET SOM
MOV	#26,4(R1)	:LOAD PORT4
ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110		:LOAD OUT DATA
DEC	TEMP1	:ALL DONE?
BNE	1\$	:BR IF NOT
JSR	PC,OUTRDY	:WAIT FOR OUTRDY
MOV	TEMP2,4(R1)	:LOAD PORT4
ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110		:LOAD OUT DATA
JSR	PC,OCOR	:WAIT FOR OCOR
RTS	PC	
:THIS SUBROUTINE LOADS THE SILO WITH THE CHARACTER PASSED TO IT.		
MOV	2(SP)+,TEMP2	:GET CHARACTER
ADD	#2,-(SP)	:ADJUST STACK
JSR	PC,OUTRDY	:WAIT FOR OUTRDY
MOV	TEMP2,4(R1)	:LOAD PORT4
ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110		:LOAD OUT DATA
JSR	PC,OUTRDY	:WAIT FOR OUTRDY
ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110		:LOAD GARBAGE CHAR
JSR	PC,OCOR	:WAIT FOR OCOR
RTS	PC	
:THIS SUBROUTINE FILLS THE OUT SILO		
: WITH A BINARY COUNT PATTERN		
MOV	#73,TEMP2	:LOAD COUNT
TST	SCHAR	:FIRST TIME HERE?
BMI	4\$	:BR IF BITSTUFF
BNE	2\$	:BR IF NC
ADD	#2,TEMP2	:ADD 2 TO CHARACTER COUNT
MOV	#3,TEMP1	:SET FOR 3 SYNCs
MOV	#26,4(R1)	:LOAD PORT4
ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122114		:LOAD SYNC REGISTER
JSR	PC,OUTRDY	:WAIT FOR OUTRDY
MOV	#1,4(R1)	:LOAD PORT4
ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122111		:SET SOM
MOV	#26,4(R1)	:LOAD PORT4
ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110		:LOAD OUT DATA

5135	032502	005337	001246	14000	DEC	TEMP1	; ALL DONE?
5136	032506	001361		14100	BNE	15	; BR IF NOT
5137	032510	004737	032176	14200	JSR	PC, OUTRDY	; WAIT FOR OUTRDY
5138	032514	013761	032646	14300	MOV	SCHAR, 4(R1)	; LOAD PORT4
5139	032522	104414			ROMCLK		; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5140	032524	122110		14500	122110		; LOAD OUT DATA
5141	032526	005737	032650	14600	TST	STUFLG	; BITSTUFF???
5142	032532	001407		14700	BEQ	65	; BR IF NO
5143	032534	013737	032646	14800	MOV	SCHAR, 55	; IT IS SOLD SO CHECK BITSTUFFING
5144	032542	004537	033414	14900	JSR	RS, STFFCL	; ADD ANY BIT STUFF CLOCK TICKS
5145	032546	000000		15000	0		; CHARACTER
5146	032550	000010		15100	10		; SHIFT COUNT
5147	032552	005237	032646	15200	65:	INC SCHAR	; NEXT CHARACTER
5148	032556	022737	000400	15300	CMP	#400, SCHAR	; ALL DONE?
5149	032564	001403		15400	BEQ	35	
5150	032566	005337	001250	15500	DEC	TEMP2	; DECREMENT COUNT
5151	032572	001346		15600	BNE	25	; BR IF NOT DONE
5152	032574	004737	032044	15700	35:	JSR PC, OCOR	; WAIT FOR OCOR
5153	032600	000207		15800	RTS	PC	
5154	032602	005037	032646	15900	45:	CLR SCHAR	; START PATTERN AT ZERO
5155	032606	012737	177777	16000	MOV	#-1, STUFLG	; SET BITSTUFF FLAG
5156	032614	005037	033612	16100	CLR	BITCON	; CLEAR STUFF COUNT
5157	032620	062737	000002	16200	ADD	#2, TEMP2	; ADD 2 TO CHARACTER COUNT
5158	032626	012761	000001	16300	MOV	#1, 4(R1)	; SET BIT0 IN PORT4
5159	032634	104414			ROMCLK		; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5160	032636	122111		16500	122111		; SET SGM!
5161	032640	104414			ROMCLK		; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5162	032642	122110		16700	122110		; LOAD JARBAGE CHAR
5163	032644	000721		16800	BR	25	; GO LOAD SILO
5164	032646	000000		16900	SCHAR:	0	
5165	032650	000000		17000	STJFLG:	0	
5166				17100			
5167				17200			
5168	032652			17300	INRDY:		
5169				17400			; THIS SUBROUTINE SPINS ON INRDY
5170				17500			; IF INRDY FAILS TO SET THE DELAY TIMES OUT AND AN
5171				17600			; ERROR IS REPORTED. FOR BETTER SCOPE LOOPS THIS
5172				17700			; DELAY CAN BE MADE SHORTER BY ALTERING THE NUMBER
5173				17800			; INITIALLY LOADED INTO TEMP1, THE SMALLER THE NUMBER
5174				17900			; THE SHORTER THE DELAY. 0 IS THE LONGEST DELAY.
5175				18000			
5176	032652	012737	000000	18100	MOV	#0, TEMP1	; SET UP DELAY COUNTER
5177	032660		001246	18200	15:		
5178	032660	104414			ROMCLK		; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5179	032662	021244		18400	021244		; PORT4+LUI2
5180	032664	032777	000020	18500	BIT	#BIT4, JOMP04	; IS INRDY SET?
5181	032672	001004		18600	BNE	25	; BR IF YES
5182	032674	005237	001246	18700	INC	TEMP1	; INC DELAY
5183	032700	001367		18800	BNE	15	; TRY AGAIN
5184	032702	104037		18900	HLT	37	; ERROR, NO INRDY
5185	032704	000207		19000	25:	RTS PC	; RETURN
5186				19100			
5187				19200			
5188	032706				SIMBCC:		
5189							; THIS SUBROUTINE CALCULATES THE CRC USING POLYNOMIAL GIVEN
5190							; IN XPOLY. THE CORRECT CRC IS RETURNED IN CALBCC, AND THE

; STATE OF THE LSB OF THE BCC IS RETURNED IN THE C BIT.

5191
 5192
 5193 032706 010046
 5194 032710 012537 001246
 5195 032714 012537 001250
 5196 032720 012537 033032
 5197 032724 013700 033032
 5198 032730 000241
 5199 032732 006037 033032
 5200 032736 006037 001250
 5201 032742 005500
 5202 032744 006000
 5203 032746 103011
 5204 032750 013700 033030
 5205 032754 043700 033032
 5206 032760 043737 033030 033032
 5207 032766 050037 033032
 5208 032772 005337 001246
 5209 032776 001352
 5210 033000 012737 000001 001246
 5211 033006 013700 033032
 5212 033012 006000
 5213 033014 005537 001246
 5214 033020 006037 001246
 5215 033024 012600
 5216 033026 000205
 5217 033030 000000
 5218 033032 000000
 5219 000200
 5220 120001
 5221 102010
 5222
 5223
 5224 033034 19800
 5225 19900
 5226 20000
 5227 20100
 5228 20200
 5229 033034 013637 001250 20300
 5230 033040 062746 000002 001246 20400
 5231 033044 012737 000002 000004 20500
 5232 033052 012761 000026 000004 20600
 5233 033060 104414
 5234 033062 122114 20800
 5235 033064 004737 032176 20900
 5236 033070 012761 000001 000004 21000
 5237 033076 104414
 5238 033100 122111 21200
 5239 033102 012761 000026 000004 21300
 5240 033110 104414
 5241 033112 122110 21500
 5242 033114 005337 001246 21600
 5243 033120 001361 21700
 5244 033122 004737 032176 21800
 5245 033126 013761 001250 000004 21900
 5246 033134 104414

1\$: MOV R0, -(SP) ;SAVE R0 ON STACK
 MOV (R5)+, TEMP1 ;TEMP1 = SHIFT COUNT
 MOV (R5)+, TEMP2 ;TEMP2 = CHARACTER
 MOV (R5)+, CALBCC ;CALBCC = OLD BCC
 MOV CALBCC, R0 ;PUT OLD BCC IN R0
 CLC
 ROR CALBCC ;SHIFT OLD BCC
 ROR TEMP2 ;SHIFT CHARACTER
 ADC R0 ;ADD CHAR CARRY TO OLD BCC
 ROR R0 ;PUT BIT0 TO CARRY BIT
 BCC 2\$;CARRY IS FEEDBACK BIT
 MOV XPOLY, R0 ;IF FEEDBACK = 1
 BIC CALBCC, R0 ;EXCLUSIVLY OR XPOLY TO CALBCC
 BIC XPOLY, CALBCC
 BIS R0, CALBCC
 2\$: DEC TEMP1 ;DEC SHIFT COUNT
 BNE 1\$;BR IF NOT DONE
 MOV #1, TEMP1 ;GET SET TO INVERT BIT0
 MOV CALBCC, R0 ;PUT RESULT IN R0
 ROR R0 ;SHIFT BIT0 TO CARRY
 ADC TEMP1 ;INVERT CARRY TO BIT0 OF TEMP1
 ROR TEMP1 ;PUT INVERTED BIT IN CARRY
 MOV (SP)+, R0 ;RESTORE R0
 RTS R5 ;RETURN
 XPOLY: 0
 CALBCC: 0
 LRC8=200
 CRC16=120001
 CRC.CCITT=102010
 BCCLD: ;THIS SUBROUTINE LOADS THE OUT SILO WITH 2 SYNCs
 ;WITH SOM SET AND ONE CHARACTER PASSED TO IT
 ;WITH THE SOM BIT CLEAR (ENABLE CRC)
 MOV 2(SP)+, TEMP2 ;GET CHARACTER
 ADD #2, -(SP) ;ADJUST STACK
 MOV #2, TEMP1 ;SET FOR 2 SYNCs
 MOV #26, 4(R1) ;LOAD PORT4
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 122114 ;LOAD SYNC REGISTER
 JSR PC, OUTPDY ;WAIT FOR OUTRDY
 MOV #1, 4(R1) ;LOAD PORT4
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 122111 ;SET SOM
 MOV #26, 4(R1) ;LOAD PORT4
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 122110 ;LOAD OUT DATA
 DEC TEMP1 ;ALL DONE?
 BNE 1\$;BR IF NOT
 JSR PC, OUTRDY ;WAIT FOR OUTRDY
 MOV TEMP2, 4(R1) ;LOAD PORT4
 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

5247	033136	122110		22100	122110		:LOAD OUT DATA
5248	033140	004737	032044	22200	JSR	PC,JCOP	:WAIT FOR JCOP
5249	033144	000207		22300	RTS	PC	
5250				22400			
5251				22500			
5252	033146				GETQ0:		
5253							:THIS SUBROUTINE READS THE STATE OF THE TRANSMIT
5254							:BCC LSB AND PUTS IT IN THE CARRY BIT
5255							
5256	033146	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5257	033150	021364			021364		:PORT4+LU-17
5258	033152	106177	146234		ROLB	QDMP04	:PUT Q0 IN CARRY
5259	033156	000207			RTS	PC	:RETURN
5260							
5261							
5262	033160				GETQ1:		
5263							:THIS SUBROUTINE READS THE STATE OF THE RECEIVE
5264							:BCC LSB AND PUTS IT IN THE CARRY BIT
5265							
5266	033160	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5267	033162	021364			021364		:PORT4+LU-17
5268	033164	106177	146222		ROLB	QDMP04	:PUT Q0 IN CARRY
5269	033170	106177	146216		ROLB	QDMP04	:PUT Q1 IN CARRY
5270	033174	000207			RTS	PC	:RETURN
5271							
5272							
5273	033176			22800	SYNLD:		
5274				22900			:THIS SUBROUTINE LOADS OUT SILO WITH
5275				23000			:2 SYNC CHARACTERS WITH SOM SET
5276				23100			
5277	033176	012737	000002	001246	MOV	#2,TEMP1	:LOAD COUNTER FOR 2 SYNC
5278	033204	012761	000026	000004	MOV	#26,4(R1)	:PORT4+26
5279	033212	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5280	033214	122114		23500	122114		:LOAD SYNC REG
5281	033216	004737	032176		JSR	PC,OUTRDY	:WAIT FOR OUTRDY
5282	033222	012761	000001	000004	MOV	#1,4(R1)	:LOAD PORT4
5283	033230	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5284	033232	122111		23900	122111		:SET SOM
5285	033234	012761	000026	000004	MOV	#26,4(R1)	:PORT+26
5286	033242	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5287	033244	122110		24200	122110		:LOAD OUT DATA WITH SYNC
5288	033246	005337	001246	24300	DEC	TEMP1	:DECREMENT COUNTER
5289	033252	001361		24400	BNE	1\$	:BR IF NOT DONE
5290	033254	000207		24500	RTS	PC	:RETURN
5291				24600			
5292				24700			
5293	033256			24800	SOM:		
5294				24900			:THIS SUBROUTINE LOADS SOM AND OUT DATA WITH A
5295				25000			:GARBAGE CHARACTER (0)
5296				25100			
5297	033256	004737	032176	000001	JSR	PC,OUTRDY	:WAIT FOR OUTRDY
5298	033262	012761	000001	000004	MOV	#1,4(R1)	:PORT4+1
5299	033270	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5300	033272	122111		25500	122111		:SET SOM
5301	033274	005061	000004	25600	CLR	4(R1)	:CLEAR DATA CHAR
5302	033280	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

5303	033302	122110	25800		122110		:LOAD GARBAGE CHARACTER
5304	033304	000207	25900		RTS	PC	:RETURN
5305			26000				
5306			26100				
5307	033306		26200	EOM:			
5308			26300				:THIS SUBROUTINE LOADS EOM AND OUT DATA WITH A
5309			26400				:GARBAGE CHARACTER (2) TO ENABLE TRANSMISSION OF BCC
5310			26500				
5311	033306	004737	26600	032176	JSR	PC,OUTRDY	:WAIT FOR OUTRDY
5312	033312	012761	26700	000004	MOV	#2,4(R1)	:PORT4+2
5313	033320	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5314	033322	122111	26900		122111		:SET EOM
5315	033324	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5316	033326	122110	27100		122110		:LOAD GARBAGE CHARACTER
5317	033330	000207	27200		RTS	PC	:RETURN
5318			27300				
5319			27400				
5320	033332		27500	MESLD:			
5321			27600				:THIS SUBROUTINE LOADS SILO WITH MESSAGE
5322			27700				:THE FIRST ARGUMENT IS THE ADDRESS OF THE MESSAGE
5323			27800				:THE SECOND ARGUMENT IS THE NUMBER OF CHARACTERS IN THE MESSAGE
5324			27900				
5325	033332	010046	28000		MOV	RO, -(SP)	:SAVE RO
5326	033334	012500	28100		MOV	(R5)+, RO	:RO=MESSAGE POINTER
5327	033336	012537	28200		MOV	(R5)+, TEMP1	:TEMP1=CHARACTER COUNT
5328	033342	004737	28300	15:	JSR	PC,OUTRDY	:WAIT FOR OUT RDY
5329	033346	112061	28400		MOVB	(R0)+, 4(R1)	:LOAD PORT4 WITH CHARACTER
5330	033352	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5331	033354	122110	28600		122110		:LOAD OUT DATA SILO
5332	033356	005337	28700		DEC	TEMP1	:DEC CHAR COUNT
5333	033362	001367	28800		BNE	15	:BR IF NOT DONE
5334	033364	004737	28900	032044	JSR	PC,OCOR	:WAIT FOR OCOR
5335	033370	012600	29000		MOV	(SP)+, RO	:RESTORE RO
5336	033372	000205	29100		RTS	RS	:RETURN
5337			29200				
5338			29300				
5339	033374		29400	CLRIC:			
5340			29500				:THIS SUBROUTINE SETS IN CLR AND OUT CLR TO
5341			29600				:CLEAR THE TRANSMIT AND RECEIVE BCC REGISTERS
5342			29700				
5343	033374	012761	29800		MOV	#BIT7, 4(R1)	:LOAD PORT4
5344	033402	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5345	033404	122112	30000		122112		:SET IN CLR!
5346	033406	104414			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5347	033410	122111	30200		122111		:SET OUT CLR!
5348	033412	000207	30300		RTS	PC	:RETURN
5349			30400				
5350			30500				
5351	033414		30600	STFFCL:			
5352			30700				:THIS SUBROUTINE ADDS ANY NECESSARY BIT STUFF CLOCK TIME
5353			30800				:FIRST ARGUMENT IS CHAR, SECOND ARGUMENT IS SHIFT COUNT.
5354			30900				
5355	033414	010046	31000		MOV	RO, -(SP)	:SAVE RO
5356	033416	012500	31100		MOV	(R5)+, RO	:PUT CHAR IN RO
5357	033420	012537	31200		MOV	(R5)+, TEMP3	:PUT SHIFT COUNT IN TEMP3
5358	033424	106000	31300	15:	PCRB	RO	:LOOK AT NEXT BIT

5359	033426	103403		31400	BCS	2\$		:BR IF A MARK
5360	033430	005037	033612	31500	CLR	BITCON		:IT WAS A SPACE, CLEAR I'S COUNTER
5361	033434	000412		31600	BR	3\$		:CONTINUE
5362	033436	0052	033612	31700	INC	BITCON		:INC CONSECUTIVE I'S COUNTER
5363	033442	022737	000005	31800	CMP	#5,BITCON		:IS IT 5 YET?
5364	033450	001004	033612	31900	BNE	3\$		:BR IF NO
5365	033452	005037	033612	32000	CLR	BITCON		:YES! SO START AGAIN
5366	033456	104415	000001	32100	DATACLK,	1		:GIVE EXTRA TICK TO STUFF ZERO
5367	033462	005337	001252	32200	DEC	TEMP3		:DEC SHIFT COUNT
5368	033466	001356		32300	BNE	1\$		:BR IF NOT DONE
5369	033470	012600		32400	MOV	(SP)+,RO		:RESTORE RO
5370	033472	000205		32500	RTS	R5		:RETURN
5371				32600				
5372				32700				
5373	033474			32800	STFFCK:			
5374				32900				:THIS SUBROUTINE CHECKS TO SEE IF TRANSMITTER
5375				33000				:IS STUFFING ZEROS WHEN IT SHOULD. FIRST ARGUMENT
5376				33100				:IS THE CHARACTER, SECOND ARGUMENT IS SHIFT COUNT.
5377				33200				
5378	033474	010046		33300	MOV	RO, -(SP)		:SAVE RO
5379	033476	012500		33400	MOV	(R5)+,RO		:PUT CHAR IN RO
5380	033500	012537	001252	33500	MOV	(R5)+,TEMP3		:PUT SHIFT COUNT IN TEMP3
5381	033504	106000		33600	RORB	RO		:SHIFT OUT NEXT BIT
5382	033506	103403		33700	BCS	2\$		:BR IF IT IS A MARK
5383	033510	005037	033612	33800	CLR	BITCON		:IT WAS A SPACE, CLEAR I'S COUNTER
5384	033514	000416		33900	BR	3\$		:CONTINUE
5385	033516	005237	033612	34000	INC	BITCON		:INC CONSECUTIVE I'S COUNTER
5386	033522	022737	000005	34100	CMP	#5,BITCON		:5 IN A ROW YET?
5387	033530	001010	033612	34200	BNE	3\$		:BR IF NO
5388	033532	005037	033612	34300	CLR	BITCON		:YES, SO START OVER
5389	033536	104415	000001	34400	DATACLK,	1		:EXTRA TICK TO STUFF ZERO
5390	033542	004737	032012	34500	JSR	PC,GETSI		:LOOK AT WINDOW
5391	033546	103001		34600	BCC	3\$		:IS IT A ZERO, BR IF YES
5392	033550	104030		34700	HLT	30		:NO, ERROR ZERO WAS NOT STUFFED
5393	033552	005337	001252	34800	DEC	TEMP3		:DEC SHIFT COUNT
5394	033556	001352		34900	BNE	1\$		:BR IF NOT DONE
5395	033560	012600		35000	MOV	(SP)+,RO		:RESTORE RO
5396	033562	000205		35100	RTS	R5		:RETURN
5397				35200				
5398				35300				
5399	033564			35400	CTSDLY:			
5400				35500				:THIS SUBROUTINE WASTES TIME UNTIL CTS SETS.
5401				35600				:BUT HOPEFULLY NOT SO LONG THAT THE SILO RUNS OUT
5402				35700				
5403	033564	010046		35800	MOV	RO, -(SP)		:SAVE RO
5404	033566	012700	000032	35900	MOV	#32,RO		:LOAD RO WITH COUNT
5405	033572	027777	145406	36000	CMP	ATKCSR,ATKCSR		:WASTE TIME
5406	033600	005300		36100	DEC	RO		:DECREMENT COUNTER
5407	033602	001373		36200	BNE	1\$		:DO IT AGAIN IF NOT - 0
5408	033604	012600		36300	MOV	(SP)+,RO		:RESTORE RO
5409	033606	000207		36400	RTS	PC		:RETURN
5410				36500				
5411				36600				
5412	033610	000176		36700	FLAG:	#B'01111110		:FLAG CHARACTER
5413	033612	000000		36800	BITCON:	0		
5414	033614	000	125	36900	MESDAT:	.BYTE 0,125,252,377		

5415	033617	377							
5416	033620	001	002	004	37000	FLTDAT: .BYTE	1,2,4,10,20,40,100,200,376,375,373,367,357,337,277,177		
5417	033623	010	020	040					
5418	033626	100	200	376					
5419	033631	375	373	367					
5420	033634	357	337	277					
5421	033637	177							
5422	033640	100	140	160	37100	STUFDT: .BYTE	100,140,160,170,3,300,174,176,177,1		
5423	033643	170	003	300					
5424	033646	174	176	177					
5425	033651	001							
5426	033652	363	347	317	37200	.BYTE	363,347,317,200,0,377,377,377,200,37		
5427	033655	200	000	377					
5428	033660	377	377	200					
5429	033663	037							
5430					37300	.EVEN			
5431	033664	046377	047111	020105	00300	EM1: .ASCIZ	<377>/LINE UNIT INITIALIZATION TEST/		
	033722	046377	047111	020105	00400	EM2: .ASCIZ	<377>/LINE UNIT REGISTER READ/ONLY TEST/		
	033765	377	044514	042516	00500	EM3: .ASCIZ	<377>/LINE UNIT REGISTER WRITE/READ TEST/		
	034031	377	044514	042516	00600	EM4: .ASCIZ	<377>/LINE UNIT INTERNAL CLOCK FAILURE		
	034073	377	051124	047101	00700	EM5: .ASCIZ	<377>/TRANSMITTER DATA ERROR/		
	034123	377	042522	042503	00800	EM6: .ASCIZ	<377>/RECEIVER TEST/		
	034142	051377	041505	044505	00900	EM7: .ASCIZ	<377>/RECEIVER DATA ERROR/		
	034167	377	047015	042504	01000	EM10: .ASCIZ	<377>/MODEM SIGNAL ERROR/		
	034213	377	051124	047101	01100	EM11: .ASCIZ	<377>/TRANSMITTER CRC ERROR/		
	034242	051377	041505	044505	01200	EM12: .ASCIZ	<377>/RECEIVER CRC ERROR/		
	034266	044777	020116	041502	01300	EM13: .ASCIZ	<377>/IN BCC MATCH ERROR (LU REG 12)/		
	034326	052377	040522	051516	01400	EM14: .ASCIZ	<377>/TRANSMITTER FAILED TO GO TO MARK STATE		
	034376	041777	041101	042514	01500	EM15: .ASCIZ	<377>/CABLE DATA TEST/		
	034417	377	046106	043501	01600	EM16: .ASCIZ	<377>/FLAG ERROR/		
	034433	377	051124	047101	01700	EM17: .ASCIZ	<377>/TRANSMITTER FAILED TO STUFF A ZERO		
	034477	377	053523	052111	01800	EM20: .ASCIZ	<377>/SWITCH PAC TEST/		
	034520	040777	047502	052122	01900	EM21: .ASCIZ	<377>/ABORT ERROR/		
	034535	377	051124	047101	02000	EM22: .ASCIZ	<377>/TRANSMITTER ERROR/		
	034560	044377	046101	020106	02100	EM23: .ASCIZ	<377>/HALF DUPLEX TEST/		
	034602	047777	052125	051040	02200	EM24: .ASCIZ	<377>/OUT READY NOT SET/		
	034625	377	047111	051040	02300	EM25: .ASCIZ	<377>/IN READY NOT SET/		
					02400				
	034647	377	054105	042520	02500	DH1: .ASCIZ	<377>/EXPECTED FOUND/		
	034670	042777	050130	041505	02600	DH2: .ASCIZ	<377>/EXPECTED FOUND LU-REGISTER/		
	034726	041777	040510	040522	02700	DH3: .ASCIZ	<377>/CHARACTER BIT THAT FAILED/		
	034764	041777	051117	042522	02800	DH4: .ASCIZ	<377>/CORRECT CRC BIT THAT FAILED		
	035024	042777	050130	041505	02900	DH5: .ASCIZ	<377>/EXPECTED FOUND SHIFT/		
	035056	042777	050130	041505	03000	DH6: .ASCIZ	<377>/EXPECTED FOUND CHARACTER SHIFT		
	035124	041377	047514	045503	03100	DH7: .ASCIZ	<377>/BLOCK END NOT SET/		
	035147	377	052122	020123	03200	DH10: .ASCIZ	<377>/RTS DID NOT CLEAR/		
					03300	.EVEN			
					03400				
	035172	000002			03500	DT1: 2			
	035174	003	007		03600	.BYTE	3,7		
	035176	001272			03700	SAVR5			
	035200	003	002		03800	.BYTE	3,2		
	035202	001270			03900	SAVR4			
	035204	000003			04000	DT2: 3			
	035206	003	007		04100	.BYTE	3,7		
	035210	001272			04200	SAVR5			

035212	003	010	04300	.BYTE	3,10
035214	001270		04400	SAVR4	
035216	003	002	04500	.BYTE	3,2
035220	001264		04600	SAVR2	
035222	000002		04700	2	
035224	003	017	04800	.BYTE	3,17
035226	001272		04900	SAVR5	
035230	002	002	05000	.BYTE	2,2
035232	001266		05100	SAVR3	
035234	000002		05200	2	
035236	006	021	05300	.BYTE	6,21
035240	033032		05400	CALBCC	
035242	002	002	05500	.BYTE	2,2
035244	001266		05600	SAVR3	
035246	000003		05700	3	
035250	001	011	05800	.BYTE	1,11
035252	001300		05900	ZERO	
035254	001	011	06000	.BYTE	1,11
035256	001302		06100	ONE	
035260	002	002	06200	.BYTE	2,2
035262	001260		06300	SAVR0	
035264	000003		06400	3	
035266	001	011	06500	.BYTE	1,11
035270	001302		06600	ONE	
035272	001	011	06700	.BYTE	1,11
035274	001300		06800	ZERO	
035276	002	002	06900	.BYTE	2,2
035300	001260			SAVR0	
035302	000004			4	
035304	001	011		.BYTE	1,11
035306	001300		07300	ZERO	
035310	001	011	07400	.BYTE	1,11
035312	001302		07500	ONE	
035314	003	007	07600	.BYTE	3,7
035316	001272		07700	SAVR5	
035320	002	001	07800	.BYTE	2,1
035322	001266		07900	SAVR3	
035324	000004		08000	4	
035326	001	011	08100	.BYTE	1,11
035330	001302		08200	ONE	
035332	001	011	08300	.BYTE	1,11
035334	001300		08400	ZERO	
035336	003	007	08500	.BYTE	3,7
035340	001272		08600	SAVR5	
035342	002	001	08700	.BYTE	2,1
035344	001266		08800	SAVR3	
035346	000002		08900	2	
035350	003	007	09000	.BYTE	3,7
035352	033610		09100	FLAG	
035354	002	002	09200	.BYTE	2,2
035356	001266		09300	SAVR3	
035360	000002		09400	2	
035362	006	004	09500	.BYTE	6,4
035364	033032		09600	CALBCC	
035366	006	002	09700	.BYTE	6,2
035370	001252		09800	TEMP3	

DDMMF MACY11 30.1046: 11-JUL-77 11:59 PAGE 103
 DDMMF.P11 12-MAY-77 14:19 SUBROUTINES

PAGE: 0123

035372		09900			
035372	000000	10000	.ERRTAB:		
035374	000000	10100		0	
035376	000000	10200		0	
035400	033664	10300		0	
035402	034670	10400	EM1		
035404	035204	10500	DH2	;HLT	1
035406	033722	10600	DT2		
035410	034670	10700	EM2		
035412	035204	10800	DH2	;HLT	2
035414	033765	10900	DT2		
035416	034670	11000	EM3		
035420	035204	11100	DH2	;HLT	3
035422	034031	11200	DT2		
035424	000000	11300	EM4		
035426	000000	11400	0	;HLT	4
035430	034073	11500	0		
035432	034670	11600	EM5		
035434	035204	11700	DH2	;HLT	5
035436	034073	11800	DT2		
035440	034726	11900	EM5		
035442	035222	12000	DH3	;HLT	6
035444	034123	12100	DT3		
035446	034647	12200	EM6		
035450	035172	12300	DH1	;HLT	7
035452	034142	12400	DT1		
035454	034647	12500	EM7		
035456	035172	12600	DH1	;HLT	10
035460	034167	12700	DT1		
035462	034647	12800	EM10		
035464	035172	12900	DH1	;HLT	11
035466	034213	13000	DT1		
035470	035024	13100	EM11		
035472	035246	13200	DH5	;HLT	12
035474	034242	13300	DT5		
035476	035024	13400	EM12		
035500	035246	13500	DH5	;HLT	13
035502	034213	13600	DT5		
035504	034764	13700	EM11		
035506	035234	13800	DH4	;HLT	14
035510	034266	13900	DT4		
035512	000000	14000	EM13		
035514	000000	14100	0	;HLT	15
035516	034213	14200	0		
035520	035024	14300	EM11		
035522	035264	14400	DH5	;HLT	16
035524	034242	14500	DT6		
035526	035024	14600	EM12		
035530	035264	14700	DH5	;HLT	17
035532	034213	14800	DT6		
035534	035056	14900	EM11		
035536	035302	15000	DH6	;HLT	20
035540	034213	15100	DT7		
035542	035056	15200	EM11		
035544	035324	15300	DH6	;HLT	21
		15400	DT10		

CCCMF MACY11 30.1046, 11-JUL-77 11:59 PAGE 104
 CCOME.P11 12-MAY-77 14:19 SUBROUTINES

PAGE: 0124

035546	034242	15500	EM12		
035550	035056	15600	DH6	:HLT	22
035552	035302	15700	DT7		
035554	034242	15800	EM12		
035556	035056	15900	DH6	:HLT	23
035560	035324	16000	DT10		
035562	034326	16100	EM14		
035564	000000	16200	0	:HLT	24
035566	000000	16300	0		
035570	034376	16400	EM15		
035572	034647	16500	DH1	:HLT	25
035574	035172	16500	DT1		
035576	034417	16700	EM16		
035600	034726	16800	DH3	:HLT	26
035602	035346	16900	DT11		
035604	034242	17000	EM12		
035606	034647	17100	DH1	:HLT	27
035610	035360	17200	DT12		
035612	034433	17300	EM17		
035614	000000	17400	0	:HLT	30
035616	000000	17500	0		
035620	034477	17600	EM20		
035622	034647	17700	DH1	:HLT	31
035624	035172	17800	DT1		
035626	034520	17900	EM21		
035630	035124	18000	DH7	:HLT	32
035632	000000	18100	0		
035634	034520	18200	EM21		
035636	034726	18300	DH3	:HLT	33
035640	035222	18400	DT3		
035642	034535	18500	EM22		
035644	035147	18600	DH10	:HLT	34
035646	000000	18700	0		
035650	034560	18800	EM23		
035652	034670	18900	DH2	:HLT	35
035654	035204	19000	DT2		
035656	034602	19100	EM24		
035660	000000	19200	0	:HLT	36
035662	000000	19300	0		
035664	034625	19400	EM25		
035666	000000	19500	0	:HLT	37
035670	000000	19500	0		
035672	034123	19700	EM6		
035674	034670	19800	DH2	:HLT	40
035676	035204	19900	DT2		
035700	034073	20000	EM5		
035702	035024	20100	DH5	:HLT	41
035704	035246	20200	DT5		
035706	034266	20300	EM13		
035710	034647	20400	DH1	:HLT	42
035712	035172	20500	DT1		
		20600			
		20700			
035714		20800	CCRMAT:		
	000001	21300	.END		

ACRCNT=	004373	879*	915*	924*										
ADJONE	003024	569	608	613	659*									
ALSTRT	002446	568*	663											
ALTS.S	010512	526	1368*											
BOCLC	033034	3424	3454	3506	3542	3594	3624	3676	3706	5224*				
EXPWARD	004714	965*	968*	969	1006*									
EXTCON	033612	2205*	2279*	2353*	2454*	2460*	2465*	2469	2471*	2504*	2537*	2554*	2621*	3195*
		3232*	3416*	3498*	3509*	3586*	3668*	3748*	3831*	3919*	4121*	4435*	5156*	5360*
		5362*	5363	5365*	5383*	5385*	5386	5388*	5413*					
BIT0	= 000001	95*	1155	1156										
BIT1	= 000002	94*	531	1149	1155	1156	1449	1532	1545					
BIT2	= 002000	85*	1513	1524										
BIT11	= 004000	84*	2057	2094	2142	2206	2280	2354	2424	2505	2585	2672	2746	2805
		2837	2869	2909	2943	3000	3046	3092	3138	3189	3280	3371	3417	3499
		3587	3669	3749	3832	3923	4055	4125	4439	4764	4832	4934		
BIT12	= 010000	93*	1464	1543										
BIT13	= 020000	82*	1467	1516	1547	1756	3320	3346	3367					
BIT14	= 040000	81*	781	1478	1480	1547	1558	1758	3322	3369	4826	4928		
BIT15	= 100000	80*	485	572	575	596	599	1451	1519					
BIT2	= 000004	93*	531	712	1156									
BIT3	= 000010	92*	1549	1556	3348									
BIT4	= 000020	91*	1139	1162	1164	2178	2957	3014	3060	3106	3152	5022	5064	5180
BIT5	= 000040	90*	1661	1761	2117	2254	2328	2402	2559	3237	4791			
BIT6	= 000100	89*	1144	1145	1551	2846	2886	2918						
BIT7	= 000200	88*	1145	1265	1661	2150	2962	5343						
BIT8	= 000400	87*	1523	1538	1540	1553	1555	1561	1564	1622				
BIT9	= 001000	86*	1521	1523	1535	1561	1564	1620	1622					
BM	007054	1187*	1492											
BSLVL	012252	1617	1627	1635	1647*									
BRW	003730	719	804*											
BRX	003732	719	805*											
CALBCC	033032	3443	3473	3531	3561	3613	3643	3695	3725	3798	3884	3951	3976*	3977
		4158	4183*	4184	4239	4261*	4265	4364	4407	4473	4498*	4499	4572	4597*
		4598	4697	4740	4844	4847*	4904	4946	4949*	4994	5196*	5197	5199*	5205
		5206*	5207*	5211	5218*	5431								
CHAR	032230	3281	5072*											
CHARSD	032342	2949	3006	3052	3098	3144	5099*							
CHARCNT	004712	963*	966	970	986*	1004*	1005							
CKSWR	007606	512	779	811	1026	1212*								
CKSWR1	007666	1225*	1237											
CKSWR2	007700	1228*												
CKSWR3	007704	1230*												
CKSWR4	007710	1231*	1239	1246										
CKSWR5	010014	1213	1220	1255*										
CLFX	001242	171*												
CLRIC	033374	2056	2093	2141	2204	2278	2352	2423	2503	2583	2671	2745	2804	2836
		2868	2908	2941	2998	3044	3090	3136	3192	3229	3279	3415	3419	3448
		3497	3500	3536	3585	3588	3619	3667	3670	3700	3747	3830	3919	4054
		4120	4434	4763	4831	4933	5339*							
CHERR	007277	626	1187*											
CNT_MA	001702	196	364*	484	486	488	1293							
CNVAT =	104411	233*	623	738	740	742	744	1060	1062	1122	1228			
CONERR	007223	621	1187*											
CONN	007114	1187*	1469											
CONTAB	002776	630	643*											
CON.F =	104410	231*	543	629	1076	1391								

CORMAX	035714	5431*												
CRAM	006606	1187*	1430											
CRC.CC=	102010	3420	3450	3502	3538	3590	3620	3672	3702	3755	3838	3945	4152	4233
		4467	4566	4834	4936	5221*								
CRC16 =	120001	5220*												
CREAM	001320	195*	483*	1289*	1290	1292*	1296							
CSR	006510	1187*	1395											
CSRMAP	010514	1370*												
CTSDLY	033564	5399*												
CYCLE	010060	720	761	762	1280*									
DATABP	005216	1049*	1052	1074	1077*									
DATACL=	104415	241*	2101	2149	2153	2221	2223	2237	2251	2295	2297	2311	2325	2369
		2371	2385	2399	2437	2439	2457	2472	2521	2523	2538	2556	2605	2607
		2625	2689	2692	2710	2719	2762	2763	2772	2809	2841	2881	2913	2951
		3008	3054	3100	3146	3191	3192	3241	3242	3286	3426	3427	3456	3457
		3508	3511	3544	3545	3596	3597	3626	3627	3678	3679	3708	3709	3769
		3773	3852	3857	3937	3952	3973	4002	4019	4026	4028	4066	4144	4159
		4186	4209	4240	4267	4290	4307	4314	4316	4327	4459	4474	4501	4524
		4542	4573	4600	4623	4640	4647	4649	4660	4773	4778	4784	5366	5399
		1048*	1070	1073*										
DATAHD	005204	237*												
DELAY =	104413	877*	912	922*										
DEVADR	004370	583	648*											
DEVTAB	003010	5431*												
DH1	034647	5431*												
DH10	035147	5431*												
DH2	034670	5431*												
DH3	034726	5431*												
DH4	034764	5431*												
DH5	035024	5431*												
DH6	035056	5431*												
DH7	035124	5431*												
DISPLA	001200	142*	498*	504*	735*									
DISPRE	000174	128*	504											
DMACTV	001306	189*	521	676*	677	1280	1294	1376*	1577*	1533*	1584*	1588	1613	
DMCM	007320	634	1187*											
DMCRO0	001500	279*												
DMCRO1	001510	284*												
DMCRO2	001520	289*												
DMCRO3	001530	294*												
DMCRO4	001540	299*												
DMCRO5	001550	304*												
DMCRO6	001560	309*												
DMCRO7	001570	314*												
DMCR10	001600	319*												
DMCR11	001610	324*												
DMCR12	001620	329*												
DMCR13	001630	334*												
DMCR14	001640	339*												
DMCR15	001650	344*												
DMCR16	001660	349*												
DMCR17	001670	354*												
DMCSP	001404	262*	568*	602	607*	612*	647	765	802	1094	1117	1163	1298*	1307
		1356	1678*											
DMCSFH	001406	263*	1144*	1145*	1149*	1155*	1156*	1162*	1164*	1307*	1309*	1309		
DMCT	001410	264*	1309*	1310*	1311									
DMCLM	001310	190*	479	751	1374*	1389*	1568*	1569	1578	1580				

PAGE: 0127

[illegible]

02DMF MACY11 301046 11-JUL-77 11:59 PAGE 109
02DME.P11 12-MAY-77 14:18 CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

[illegible]

N10

02DMF MACY11 30(1046) 11-JUL-77 11:59 PAGE 111

PAGE: 0130

DZDME.P11 12-MAY-77 14:18

CROSS REFERENCE TABLE -- USER SYMBOLS

MLOCK	006043	714	1187#											
MNEW	006144	669	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#												
MS*CLR=	104412	235#	1126	1725	1749	1781	1823	1865	1923	1974	1996	2018	2052	2089
		2137	2200	2274	2348	2419	2499	2579	2667	2741	2800	2832	2864	2904
		2937	2994	3040	3086	3132	3178	3225	3275	3319	3366	3411	3493	3581
		3663	3743	3826	3914	4050	4116	4430	4759	4825	4927			
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*	2087*	2135*	2198*	2272*	2346*	2417*	2497*	2577*	2665*	2739*	2798*
		2830*	2862*	2902*	2935*	2992*	3038*	3084*	3130*	3176*	3223*	3273*	3317*	3364*
		3408*	3490*	3578*	3660*	3741*	3824*	3912*	4048*	4114*	4428*	4757*	4823*	4925*
NITCH	032042	5009*	5010*	5011*	5012*	5014#								
NOACT	007154	523	1187#	1282										
NODEV	002674	577	616#											
NUM	006450	1187#	1382											
OCOR	032044	2100	2148	2218	2292	2366	2518	2604	2687	2760	2880	3768	3851	3935
		4065	4142	4457	4772	4777	4863	4955	5017#	5023	5053	5095	5111	5152
		5248	5334											
OK	002646	603	610#	639										
ONE	001302	187#	5431											
OUTRDY	032176	2207	2214	2281	2288	2355	2362	2425	2506	2514	2589	2595	2600	2647
		2675	2683	2747	2871	3283	3929	4059	4131	4445	4768	5037	5046	5057#
		5082	5091	5104	5108	5128	5137	5235	5244	5281	5297	5311	5328	
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#		</										

SPACNT=	004713	964*	988	991*	1005*									
SPEED	007340	1187*	1440											
STACK =	001200	64*	477	690	1093	1116								
STAT	001240	170*												
STAT1	001366	251*	1301*	1756	1758	3320	3322	3346	3367	3369	4826	4928		
STAT2	001370	252*	1302*	1978	2000									
STAT3	001372	253*	1303*											
STFFCK	033474	2549	2636	3512	3774	3858	3963	3990	4170	4197	4251	4278	4485	4512
		4584	4611	5373*										
STFFCL	033414	5144	5351*											
STR*SW	001236	169*	513*	516*	517	519	529	531	666	712	721	1323	1347*	1377
		1601												
STUFDT	033640	2432	2434	5422*										
STUFLG	032650	3186*	3233*	5141	5155*	5165*								
SVOS	004402	933*												
SWFLG	010016	480*	825	1224*	1251*	1257*								
SWMES	007205	1187*	1227											
SWMES1	007215	1187*	1230											
SWR	001202	143*	497*	499	503*	513	671	676	781	789	812	827	1027	1032
		1081	1088	1090	1152	1212	1250*							
SWREG	000176	129*	503	1212	1270									
SWOC =	000001	45*	517	1377	1601									
SWO1 =	000002	44*	721	1323	1347									
SWO2 =	000004	43*												
SWO3 =	000010	42*	666											
SWO4 =	000020	41*												
SWO5 =	000040	40*												
SWO6 =	000100	39*	1152											
SWO7 =	000200	38*												
SWO8 =	000400	37*	1088											
SWO9 =	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	032062	2807	2839	2911	5027*									
SYNLD	033176	3756	3839	3928	4058	4130	4444	4767	4833	4935	5273*			
TEMP	001416	271*	971	1118*	1119*	1160*	1165*	1171*	1183*	2019*	2025*	2029*	2034*	
TEMP1	001246	173*	537*	1189	1613*	1614*	5032*	5044*	5078*	5089*	5124*	5135*	5176*	5192*
		5194*	5208*	5210*	5213*	5214*	5231*	5242*	5277*	5288*	5327*	5332*		
TEMP2	001250	174*	538*	1191	4361*	4362*	4363	4404*	4405*	4406	4694*	4695*	4696	4737*
		4738*	4739	4901*	4902*	4903	4991*	4992*	4993	5076*	5092	5102*	5105	5119*
		5123*	5150*	5157*	5195*	5200*	5229*	5245						
TEMP3	001252	175*	540*	566*	618	627*	1193	1386	1389	1501*	2222*	2224*	2296*	2298*
		2370*	2372*	2438*	2440*	2522*	2524*	2606*	2608*	2691*	2693*	2771*	2773*	4001*
		4003*	4083*	4084*	4208*	4210*	4289*	4291*	4347*	4348*	4363*	4364	4390*	4391*
		4406*	4407	4523*	4525*	4541*	4543*	4622*	4624*	4680*	4681*	4696*	4697	4723*
		4724*	4739*	4740	4887*	4888*	4903*	4904	4977*	4978*	4993*	4994	5357*	5367*
		5380*	5393*	5431										
TEMP4	001254	176*	541*	1195	1399	1402	1408	1411	1487	1490	1496	1499		
TEMP5	001256	177*	542*	1197	1380*	1393*	1599	5060*	5066*					
TIME =	104416	243*	2063	2952	3009	3055	3101	3147	3328	3340	3375			
TCR	001204	148*	848	1214	1259	1655	5405							
TCR	001206	149*	350	856	1217	1219	1261	1657						

[illegible]

020MF MACY11 30.1046) 11-JUL-77 11:59 PAGE 115
020ME.P11 12-MAY-77 14:18 CROSS REFERENCE TABLE -- USER SYMBOLS

020MF MACY11 30(1046) 11-JUL-77 11:59 PAGE 115

4823	4924#	4998										
4925												
1820	1861#											
715*	716*	718*	719*	782#								
96#												
1053	1071	1074#										
219#	511	523	535	620	625	633	636	668	673	714	723	736
737	739	741	743	831	944	861	954	994	1054	1055	1058	1059
1061	1063	1067	1072	1121	1227	1230	1282	1322	1346	1352	1353	1412
1426	1429	1436	1439	1446	1452	1461	1468	1475	1590			
1051	1054#											
1187#	1404											
1589	1601#											
1392	1597#											
962*	995*	1003#										
1066	1069#											
1028	1030	1032#										
738	763#											
744	772#											
535	1187#											
593*	611*	614	645	658#								
742	769#											
3420*	3450*	3502*	3538*	3590*	3620*	3672*	3702*	3755*	3838*	3945*	4152*	4233*
4467*	4566*	4834*	4936*	5204	5206	5217#						
544	1187#											
1060	1100#											
740	766#											
186#	5431											
1												
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#	2075#	2079
2082#	2123#	2126	2130#	2184#	2187	2193#	2258#	2261	2267#	2332#	2335	2341#
2406#	2409	2412#	2483#	2486	2492#	2563#	2566	2572#	2652#	2655	2660#	2727#
2730	2734#	2787#	2790	2793#	2819#	2822	2825#	2851#	2854	2857#	2891#	2894
2897#	2923#	2926	2930#	2981#	2984	2987#	3027#	3030	3033#	3073#	3076	3079#
3119#	3122	3125#	3165#	3168	3171#	3210#	3213	3218#	3260#	3263	3268#	3305#
3308	3312#	3353#	3356	3359#	3396#	3399	3403#	3478#	3481	3485#	3566#	3569
3573#	3648#	3651	3655#	3730#	3733	3736#	3813#	3816	3819#	3899#	3902	3907#
4036#	4039	4043#	4098#	4101	4109#	4411#	4414	4423#	4744#	4747	4752#	4808#
4811	4818#	4911#	4914	4920#								
123	509	755#	1079									
1#	1665	1671	1673	1678#	1689	1695	1697	1702#	1712	1718	1720	1725
1726#	1736	1742	1744	1749	1750#	1767	1773	1775	1781	1782#	1809	1815
1817	1823	1824#	1851	1857	1859	1						

[illegible]

[illegible]

	2421	2427	2429	2501	2508	2510	2516	2557	2581	2591	2593	2597	2602	2646	2669
	2677	2679	2685	2743	2749	2751	2754	2756	2758	2802	2810	2834	2842	2866	2873
	2875	2878	2882	2906	2914	2939	2945	2947	2953	2963	2965	2996	3002	3004	3010
	3019	3042	3048	3050	3056	3065	3088	3094	3096	3102	3111	3134	3140	3142	3148
	3157	3180	3194	3227	3238	3244	3277	3284	3268	3296	3326	3329	3338	3341	3373
	3377	3384	3413	3495	3583	3665	3745	3758	3762	3766	3807	3828	3841	3845	3849
	3893	3916	4052	4068	4081	4086	4095	4118	4331	4345	4350	4359	4374	4382	4393
	4402	4432	4664	4678	4683	4692	4707	4721	4726	4735	4761	4779	4786	4796	4829
	4869	4885	4890	4899	4931	4960	4975	4980	4989	5007	5020	5035	5039	5042	5048
	5051	5062	5080	5084	5087	5093	5106	5109	5126	5130	5133	5139	5159	5161	5178
	5233	5237	5240	5246	5256	5266	5279	5283	5286	5299	5302	5313	5315	5330	5344
	5346														
\$ACRC	18	4036													
\$REC	18	2923	2981	3027	3073	3119									
\$SCOPE	18	775													
\$SIMBC	18	5198													
\$SINAC	18	2951													
\$SOFTC	18	1207													
\$STUFF	18	2406													
\$SWPAC	18	1961	1983												
\$YCHAR	18	3923	4055	4125	4439										
\$YCRC	18	3899	4098	4411											
\$TRANW	18	3944	4151	4232	4466	4565									
\$TRANI	18	2038	2075	2123											
\$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	2084	2132
	2195	2269	2343	2414	2494	2574	2662	2736	2755	2827	2859	2899	2932	2989	3035
	3081	3127	3173	3220	3270	3314	3361	3405	3487	3575	3657	3738	3821	3909	4045
	4111	4425	4754	4820	4922										
\$VARI	18	134													
\$WINDC	18	2184	2258	2332	2483										
\$XZ	18	1665	1671	1689	1695	1712	1718	1736	1742	1767	1773	1809	1815	1851	1857
	1909	1915	1961	1967	1983	1989	2005	2011	2038	2045	2075	2082	2123	2130	2184
	2193	2258	2267	2332	2341	2406	2412	2483	2492	2563	2572	2652	2660	2713	2714
	2787	2793	2819	2825	2851	2857	2891	2897	2923	2930	2981	2987	3027	3033	3073
	3079	3119	3125	3165	3171	3210	3218	3260	3268	3305	3312	3353	3359	3396	3403
	3478	3485	3566	3573	3648	3655	3730	3736	3813	3819	3899	3907	4036	4043	4058
	4109	4411	4423	4744	4752	4808	4818	4911	4920						
\$ZEROS	18	2727													

. RES. 035714 000

ERRORS DETECTED: 0

DZDMF DZDMF/SGL/CRF+IPLUTL DZDME/EG:LU/PE

RUN-TIME: 17 23 1 SECONDS

RUN-TIME RATIO: 257/42=6.0

CORE USED: 32K (63 PAGES)