PDP11

BIT STUFF MODE LINE UNITS

MD-11-DZKCF-A

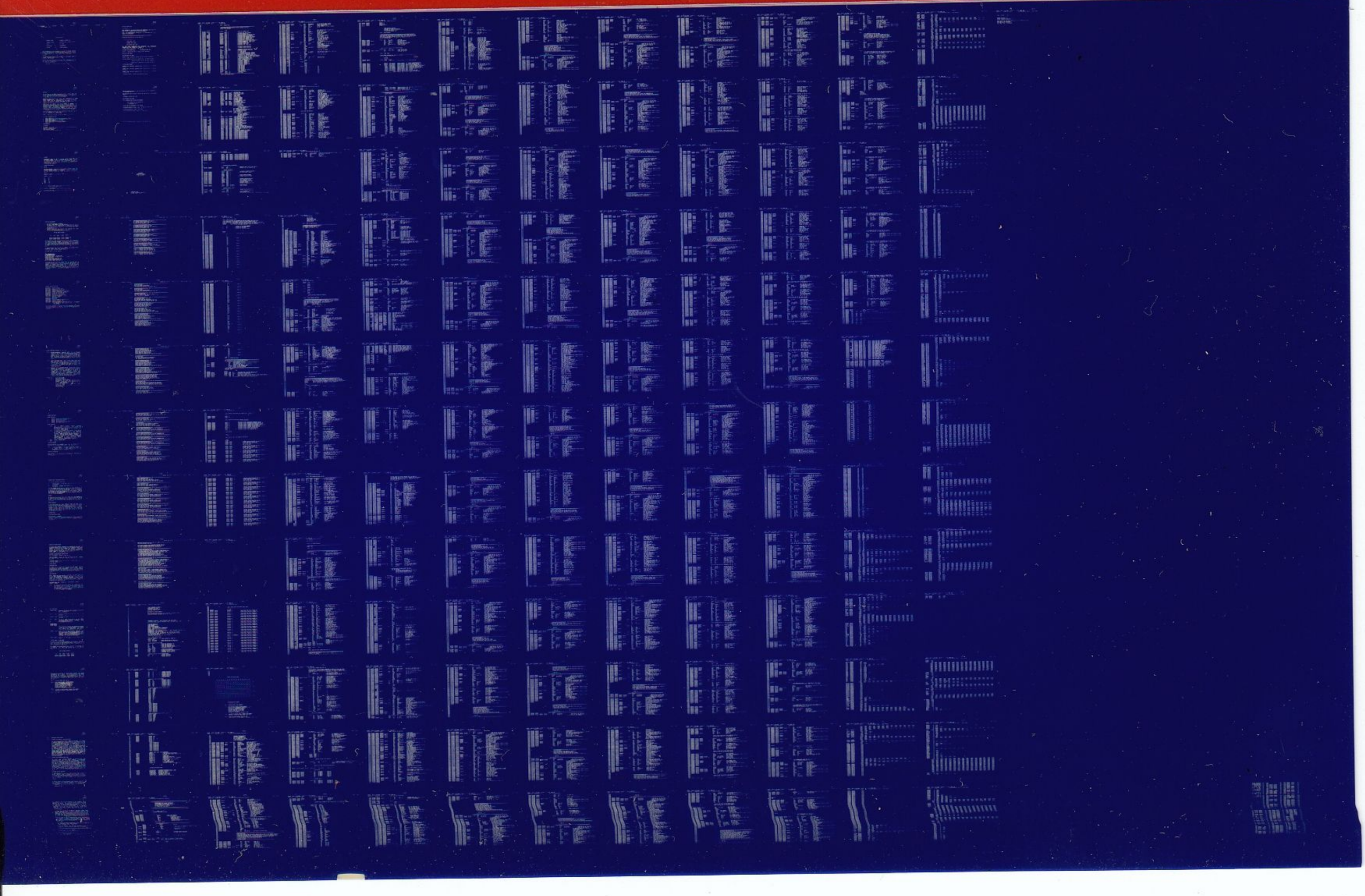
EP-DZKCF-A-DL-A

FICHE 1 OF 1

AUG 1977

digital

MADE IN USA



IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZKCF-A-D
PRODUCT NAME: BITSTUFF MODE LINE UNIT TESTS
DATE: MAY 1976
MAINTAINER: DIAGNOSTICS
AUTHOR: DINESH GORADIA

1. ABSTRACT

The function of the KMC11 diagnostics is to verify that the option operates according to specifications. The diagnostics verfiy that there are no malfunctions and tha all operations of the KMC11 are correct in its environment.

Parameters must be set up to alert the diagnostics to the KMC11 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.

DZKCF tests the KMC-11 Line Unit (M8201 or M8202). It performs write/read tests on the KMC Line Unit registers. it checks for proper transmitter, receiver, and BCC operation in BITSTUFF mode. The modem signals are also checked. DZKCF requires a KMC Micro-Processor (M8204) to run. For best diagnosis a turn-around connector should be installed, however the diagnostic will run without it (some tests are skipped).

Currently there are four 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 four diagnostics are:

1. DZKCC [REV] Basic W/R and Micro-processor tests
2. DZKCD [REV] Jump and main memorytests (Heat test tape)
3. DZKCE [REV] DDCMP Line unit tests
4. DZKCF [REV] BITSTUFF Line unit tests
5. DZKCA [REV] KMC11 CPU MICRO-DIGNOSTICS.

2. REQUIREMENTS

2.1 EQUIPMENT

Any PDP11 family CPU (except an LSI-11) with minimum 8K memory
ASR 33 (or equilivalent)
KMC11-AN IOP (M8204)
KMC11-DA OR KMC11-MD OR KMC11-MA

2.2 STORAGE

Program will use all 8K of memory except where ABL and BOOTSTRAP LOADER reside. Locations 2100 thru 2300; 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 PROCEEDURE

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 KMC11 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 KMC11 STATUS

PC	CSR	STAT1	STAT2	STAT3
002100	160010	045310	177777	000000
002110	160020	045320	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. 2100 in the program. In this example the table contains the information and status of two KMC11'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 KMC11'S TO BE TESTED?1

01
CSR ADDRESS?160010
VECTOR ADDRESS?310
BR PRIORITY LEVEL?  (4,5,6,7)?5
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 KMC11's desired active
SW 02 Set: Lock on selected test
SW 01 Set: Restart program at selected test
SW 00 Set: Build new status table from questions. (If SW07=0 and SW00=0 a new status table is built by auto-sizing)

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

4.1.2 SWITCH REGISTER OPTIONS (at start up)

SW 01 RESTART PROGRAM AT SELECTED TEST. It is strongly suggested that at least one pass has been made before trying to select a test, the reason being is that the program has to clear areas and set up parameters. When this switch is used the diagnostic will ask TEST NO.? Answer by typing the number of the test desired and carrige return to Begin execution at the selected test.

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

SW 03 RESELECT KMC11'S DESIRED ACTIVE. Please note that a message is typed out for setting the switch register equal to KMC11's active. this means if the system has four KMC11s; bits 00,01,02,03 will be set in loc 'KMACTV' from the switch register. Using this switch(SW00) alters that location;therefore if four KMC11s are in the system ***DO NOT*** set switchs greater than SW 03 in the up position. this would be a fatal error. do not select more active KMC11s 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 KMC desired active.
EXAMPLE: If you have 4 KMC's but only want to run the first and the last set SWR bits 0 and 3 = 1. PRESS CONTINUE
E: Number (IF VALID) will be in data lights (excluding 11/05)
F: Set with any other switch settings desired.
PRESS CONTINUE.

4.1.3 DYNAMIC SWITCHES

ERROR SWITCHES

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

SCOPE SWITCHES

1. SW06 Halt in ROMCLK routine before clocking micro-processor instruction. This allows the operator to scope a micro-processor instruction in the static state before it is clocked. Hit continue to resume running.
2. SW09 (if enabled by 'SCOPI') on an error; If an '*' is printed in front of the test no. (ex. *TEST NO. 10) SW09 is incorporated in that test and therefore SW09 is usually the best switch for the scope loop (SW14=0, SW10=0, SW09=1, SW08=0). If SW09 is not enabeled; and there is a HARD error (constant); SW08 is best. (SW14=1,0, SW10=0, SW09=0, SW08=1). for intermittemt errors; SW14=1 will loop on test reguardless of error or not error. (SW14=1, SW10=0, SW09=0, SW08=1,0)
3. SW11 Inhibit interations.
4. SW14 Loop on current test.

4.2 STARTING ADDRESS

Starting address is at 000200 there are no other starting addresses for the KMC11 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 KMC11'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 possibily an error message (this depends on the test) to give the operator an idea as to the source of the problem. If it is necessary to know more information concerning the error report; LOOK IN THE LISTING for that TEST NUMBER which was typed out and then NOTE THE PC of thE ERROR REPORT this way the EXACT FUNCTION of the test CAN BE DETERMINED.

6. ERRORS

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

6.2 ERROR RECOVERY

If for some reason the KMC11 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 'STSTNM' (address 1202)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 KMC11 was doing at the time of the error.

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

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

7.2 OPERATING RESTRICTIONS

The first time a KMC11 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 KMC 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

KMC11 IOP(M8204)- JUMPER W1 MUST BE IN,

LINE UNIT(M8201)- Jumpers W1, W2, and W4 must be IN. Jumpers W3, and W5 must be OUT. SW8 of E26 must be in the ON position.

LINE UNIT (M8202)- Jumper W1 must be in. SW8 of E26 must be in the OFF position.

8. MISCELLANEOUS

8.1 EXECUTION TIME

All KMC11 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 KMC11'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 DZKCF CSR: 175000 VEC: 0300 PASSES: 000001
ERRORS: 000000

NOTE: The pass count and error counts are cummulitive for each KMC11 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 KMC11 since the diagnostic was started are reflected in PASSES: and ERRORS:.

8.4 KEY LOCATIONS

SLPADR (1206) Contains the address where program will return when iteration count is reached or if loop on test is asserted.

NEXT (1442) Contains the address of the next test to be peformed.

STSTNM (1202) Contains the number of the test now being peformed.

RUN (1500) The bit in 'RUN' always points to the KMC11 currently being tested. EXAMPLE: (RUN) 1500/0000000001000000 Means that KMC11 no.06 is the KMC11 now running.

KMCR00-KMCR17
KMST00-KMST17
(2100)-(2300)

These locations contain the information needed to test up to 16 (decimal) KMC11s sequentialy. they contain the CSR,VECTOR and STATUS concerning the configuration of each KMC11.

KMACTV (1470) Each bit set in this location indicates that the associated KMC11 will be tested in turn. EXAMPLE: (KMACTV) 1470/0000000000011111 means that KMC11 no. 00,01,02,03,04 will be tested. EXAMPLE: (KMACTV) 1470/0000000000010001 Means that KMC11 no. 00,04 will be tested.

KMCSR (2066) Contains the CSR of the current KMC11 under test.

8.4A 'STATUS TABLE' (2100-2300)

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 KMC11'S. the table can contain up to 16 KMC11'S. Following the map is a description of the bits for each map entry

MAP OF KMC11 STATUS

PC	CSR	STAT1	STAT2	STAT3
002100	160010	045310	177777	000000
002110	160020	016320	000000	000000

Each map entry contains 4 words which contain the status information for 1 KMC11. The PC shows where in core memory the first of the 4 words is. In the example above the first KMC'S status is in locations, 2100, 2102, 2104, and 2106. The second KMC status is located at 2110, 2112, 2114, and 2116. The information contained in each 4 word entry is defined as follows:

CSR: Contains KMC11 CSR address

STAT1: BITS 00-08 IS KMC11 VECTOR ADDRESS
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 KMC11 BR PRIORITY LEVEL

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

STAT3: NOT USED

8.5 METHOD OF AUTO SIZING

8.5.1 FINDING THE CONTROL STATUS REGISTER.

The auto-sizing routine finds a KMC11 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 CRAM address is written to a 125252 then it is read back. If it contains a -1 or 125252 a KMC11 has been found, if not, the address is updated by 10 and the search continues. A -1 indicates a KMC11 with no CRAM, a 125252 indicates a KMC11 with CRAM 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 KMC11's in the system will be found by the auto-sizer. If it does not find a KMC11 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 KMC is programmed to interrupt. The PS is lowered by 1 until the KMC interrupts, a delay is made and if no interupt occures at PS level 3 (because of a bad KMC11) 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 interupt occured; the address to which the KMC11 interupted to is picked up and reported as the vector. NOTE: if the vector reported is not the vector set up by you; there is a problem and AUTO SIZING should not be done.

8.6 SOFTWARE SWITCH REGISTER

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

Control:

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

```
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.

****

Note:for ipg's line unit m8202-ye users.

Cable data test:[test 60, test 61]

these tests won't run reliably on line units without terminating resistence.

APT/ACT/XXDP/SLIDE

++++++++++

THIS DIAGNOSTIC IS APT/ACT/XXDP/SLIDE COMPATIBLE USER WOULD BE ABLE TO RUN IT UNDER APT/ACT/XXDP ENVIRONMENT.

NOTE: FOR MANUFACTURING PURPOSE ONLY ITS DESCRIBED HOW TO RUN UNDER APT ENVIRONMENT.

****

ETABLE SETTING FOR APT TO RUN UNDER APT

++++++++++

FIRST PASS TIME:

LONGEST TEST TIME:

ADDITIONAL TEST TIME:

ALL THE ABOVE PARAMETERS ARE DEPENDENT ON PARTICULAR DIAGNOSTICS AND SHOULD BE LOADED AT THE TIME OF SETTING ETABLE.THERE IS NO DEFAULT TIME SET UP.

SOFTWARE ENVIRONMENT:001 ENVIRONMENT MODE:200

SWITCH 1:-SHOULD BE USED AS NORMAL SWITCH REGISTER.

SWITCH 2:-NOT USED.

CPU OPTIONS:-NOT USED.

MEMORY TYPE 1:-BITS<2:4>:=BITS <12:14> OF STAT1 OF DEV:0.

MAXIMUM ADDRESS:-BITS<17:19>:=BITS<12:14> OF STAT1 OF DEV:1
BITS<2:4>:=BITS <12:14> OF STAT1 OF DEV:2
BITS<10:12>:=BITS<12:14> OF STAT1 OF DEV:3

IN THE SAME MANNER

MEMORY TYPE 2 MAXIMUM ADDRESS:-GETS STAT1<12:14> OF DEVICE 4,5,6,7.

MEMORY TYPE 3 MAXIMUM ADDRESS:-GETS STAT1<12:14> OF DEVICE 8,9,10,11.

MEMORY TYPE 4 MAXIMUM ADDRESS:-GETS STAT1<12:14> OF DEVICE 12,13,14,15.

INTERRUPT VECTOR 1:FIRST DEVICE RECEIVE VECTOR.

REST OF THE DEVICE(KMC'S) VECTOR SHOULD BE SET UP SEQUENTIALLY
IN INCREMENTS OF 10.

BUS PRIORITY:KMC'S PRIORITY(SHOULD BE SAME FOR ALL KMC'S UNDER
TEST).

INTERRUPT VECTOR 2:NOT USED.

BUS PRIORITY:NOT USED.

BASE ADDRESS:FIRST DEVICE CSR ADDRESS.

REST SHOULD FOLLOW SEQUENTIALLY

IN INCREMENTS OF 10.

DEVICE MAP:AS DESCRIBED IN APT MANUAL.

CONTROLLER SPECIFIC CODE 1:-NO. OF DEVICES UNDER TEST.

CONTROLLER SPECIFIC CODE 2:-NOT USED.

DEVICE DESCRIPTOR WORD 0:STAT2 OF FIRST DEVICE.

. .

TO

. .

. .

DEVICE DESCRIPTOR WORD 15:STAT2 OF 16TH DEVICE.(KMC)

DOCUMENT
****
MAINDEC-11-DZKCF-A
****

```
2265    ********** TEST 1 **********
        OUT CONTROL REGISTER READ/ONLY TEST
        DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
        BITS ARE IN THE CORRECT STATE

2291    ********** TEST 2 **********
        IN CONTROL REGISTER READ/ONLY TEST
        DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
        BITS ARE IN THE CORRECT STATE

2316    ********** TEST 3 **********
        MODEM CONTROL REGISTER READ/ONLY TEST
        DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
        BITS ARE IN THE CORRECT STATE

2342    ********** TEST 4 **********
        MAINTENANCE REGISTER READ/ONLY TEST
        DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
        BITS ARE IN THE CORRECT STATE

2372    ********** 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

2415    ********** 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

2458    ********** TEST 7 **********
        LINE UNIT REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH LINE UNIT REGISTER 13
        FLOAT A 0 THROUGH LINE UNIT REGISTER 13

2517    ********** TEST 10 **********
        LINE UNIT REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH LINE UNIT REGISTER 14
        FLOAT A 0 THROUGH LINE UNIT REGISTER 14
```

```
2570    ************************** TEST 11 ****************************
        SWITCH PAC TEST
        THIS TEST READS SWITCH PAC#1
        THIS SWITCH PAC CONTAINS THE DDCMP LINE #

2594    ************************** TEST 12 ****************************
        SWITCH PAC TEST
        THIS TEST READS SWITCH PAC#2
        THIS SWITCH PAC CONTAINS THE BM873 BOOT ADD

2618    ************************** TEST 13 ****************************
        LINE UNIT CLOCK TEST
        THIS TEST VERIFYS THAT THE LU INTERNAL CLOCK
        (BIT 1 IN LU-17) IS WORKING

2653    ************************** 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

2691    ************************** 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

2740    ************************** 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

2802    ************************** 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

2878    ************************** 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
```

```
2954    ************************** TEST 21 ****************************
        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

3030    ************************** TEST 22 ****************************
        BIT STUFF TEST
        THIS TEST CHECKS ZERO BIT STUFFING OF
         THE TRANSMITTER IN THE BIT WINDOW

3109    ************************** 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

3191    ************************** 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

3282    ************************** TEST 25 ****************************
        MULTIPLE FLAG AND TRANSMITTER ABORT TEST
        LOAD SILO WITH 5 FLAGS AND A CHAR (000)
        VERIFIY IN THE BIT WINDOW THAT THE FLAGS
        AND DATA ARE CORRECT AND FOLLOWED BY AN ABORT
        SEQUENCE (8 CONTIGUOUS 1'S)

3358    ************************** 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

3419    ************************** TEST 27 ****************************
        BITSTUFF STRIP FLAG TEST
        SET LU LOOP, SINGLE STEP 5 FLAGS,
        VERIFY THAT IN ACTIVE DOES NOT SET

3453    ************************** TEST 30 ****************************
        BITSTUFF IN ACTIVE TEST
        SET LU LOOP, SINGLE STEP 5 FLAGS AND A NON-FLAG (301)
        VERIFY THAT IN ACTIVE IS SET
```

```
3487    ********************** TEST 31 **********************
        BITSTUFF IN ACTIVE TEST
        SET LINE UNIT LOOP,SINGLE STEP ONE FLAG AND A CHAR (301)
        VERIFY THAT IN ACTIVE IS SET

3529    ********************** TEST 32 **********************
        BITSTUFF IN ACTIVE TEST
        SET LU LOOP, SINGLE STEP 2 FLAGS AND A NON-FLAG (301)
        VERIFY THAT IN ACTIVE IS SET

3563    ********************** 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

3623    ********************** 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

3671    ********************** 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

3719    ********************** 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

3767    ********************** 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

3815    ********************** TEST 40 **********************
        BITSTUFF DATA TEST
        THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
        CHECKING EACH CHARACTER AS IT IS RECEIVED

3862    ********************** 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

3914    ********************** 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 RECOGINIZED
        THE ABORT SEQUENCE (8 CONTIGUIOUS 1'S)
```

```
3961    ************************** TEST 43 ****************************
        CABLE TURNAROUND TEST
        CLEAR LINE UNIT LOOP, SET DTR
        VERIFY THAT RING AND MODEM READY ARE SET
        CLEAR DTR, VERIFY THAT RING AND MRDY ARE CLEARED

4014    ************************** TEST 44 ****************************
        CABLE TURNAROUND TEST
        CLEAR LINE UNIT LOOP, LOAD OUT DATA SILO
        VERIFY THAT ALL MODEM SIGNALS ARE SET

4062    ************************** TEST 45 ****************************
        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

4146    ************************** 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

4236    ************************** 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

4320    ************************** 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

4404    ************************** TEST 51 ****************************
        TRANSMITTER CRC TEST
        USING THE CRC.CCITT POLYNOMINAL, SINGLE CLOCK A BINARY
        COUNT PATTERN, VERIFY THE LSB OF THE TRANSMITTER BCC ON EACH SHIFT

4489    ************************** TEST 52 ****************************
        RECEIVER CRC TEST
        USING THE CRC.CCITT POLYNOMINAL, SINGLE CLOCK A BINARY
        COUNT PATTERN, VERIFY THE LSB OF THE RECEIVER BCC ON EACH SHIFT

4577    ************************** TEST 53 ****************************
        TRANSMITTER BITSTUFF CRC TEST
        THIS TEST TRANSMITS A FOUR CHARACTER MESSAGE WITH CRC
        BOTH DATA AND THE BCC ARE VERIFIED IN THE BIT
        WINDOW. THE FOUR CHARACTERS ARE 0,125,252,377
        THE TRANSMITTER IS CHECKED FOR GOING TO A MARK STATE AFTER THE BCC
```

```
4716    ********************** TEST 54 **********************
        RECEIVER BITSTUFF CRC TEST

4718    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

4780    ********************** 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 CHARCTERS LOADED WITH EOM SET ARE LOST
        ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
        THE FOUR CHARACTER MESSAGE IS 0,125,252,377
        RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED

5095    ********************** 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 CHARCTERS 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

5430    ********************** 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

5495    ********************** 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

5502    LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST

5600    ********************** 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
```

```
                        .TITLE  MAINDEC-11-DZKCF-A
                        ;*COPYRIGHT (C) 1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY DINESH GORADIA
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
                        ;*MAINDEC-11-DZKCF-A   KMC11 BITSTUFF LINE UNIT TESTS
                        ;*COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
                        ;*-----------------------------------------------------------------

                        ;STARTING PROCEDURE
                        ;LOAD PROGRAM
                        ;LOAD ADDRESS 000200
                        ;SWR=0  AUTOSIZE KMC11
                        ;SW07=1  USE CURRENT KMC11 PARAMETERS
                        ;SW00=1  INPUT NEW KMC11 PARAMETERS
                        ;PRESS START
                        ;PROGRAM WILL TYPE "MAINDEC-11-DZKCF-A   KMC11 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

                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1200 ***
001200                  STACK=  1200
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0=     %0              ;;GENERAL REGISTER
000001                  R1=     %1              ;;GENERAL REGISTER
000002                  R2=     %2              ;;GENERAL REGISTER
```

```
                        
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5=     %5              ;;GENERAL REGISTER
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER
                        
                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7
                        
                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0
                        
                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
```

```
                        BIT06=  100
                        BIT05=  40
                        BIT04=  20
                        BIT03=  10
                        BIT02=  4
                        BIT01=  2
000001                  BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR



                        ;INSTRUCTION DEFINITIONS
                        ;-----------------------

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

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

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

            000020              .=20
000020  004134                          $SCOPE          ; SCOPE LOOP HANDLER.
000022  000340                          PR7             ; SERVICE AT LEVEL 7.
000024  007126                                  $PWRDN          ;POWER FAIL HANDLER
000026  000340                          PR7                     ;SERVICE AT LEVEL 7
000030  006512                          $ERROR                  ;ERROR HANDLER
000032  000340                          PR7                     ;SERVICE AT LEVEL 7
000034  006414                          $TRAP                   ;GENERAL HANDLER DISPATCH SERVICE
000036  000340                          PR7                     ;SERVICE AT LEVEL 7
                                .SBTTL  ACT11 HOOKS

                                ;;**************************************************************************
                                ;HOOKS REQUIRED BY ACT11
            000040                      $SVPC=.                 ;SAVE PC
            000046                      .=46
000046  004070                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
            000052                      .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
            000040                      .=$SVPC                 ;; RESTORE PC

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

            000200              .=200
000200  000137  002402                  JMP     .START          ;GO TO START OF PROGRAM


            001000              .=1000
001000  005200  040515  047111  MTITLE: .ASCII  <200><12>/MAINDEC-11-DZKCF-A/<200>
(2) 001025     113  041515  030461          .ASCIZ  /KMC11 BITSTUFF LINE UNIT TESTS/<200>
(2)
            177570              DSWR    =       177570
            177570              DDISP   =       177570
```

```
                        .SBTTL  COMMON TAGS

                        ;;*****************************************************************
                        ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                        ;*USED IN THE PROGRAM.

001200                          .=1200
001200                  $CMTAG:                 ;;START OF COMMON TAGS
001200  000000                  .WORD   0
001202  000            $TSTNM:  .BYTE   0       ;;CONTAINS THE TEST NUMBER
001203  000            $ERFLG:  .BYTE   0       ;;CONTAINS ERROR FLAG
001204  000000         $ICNT:   .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
001206  000000         $LPADR:  .WORD   0       ;;CONTAINS SCOPE LOOP ADDRESS
001210  000000         $LPERR:  .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
001212  000000         $ERTTL:  .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
001214  000            $ITEMB:  .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
001215  001            $ERMAX:  .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
001216  000000         $ERRPC:  .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001220  000000         $GDADR:  .WORD   0       ;;CONTAINS ADDRESS OF 'GOOD' DATA
001222  000000         $BDADR:  .WORD   0       ;;CONTAINS ADDRESS OF 'BAD' DATA
001224  000000         $GDDAT:  .WORD   0       ;;CONTAINS 'GOOD' DATA
001226  000000         $BDDAT:  .WORD   0       ;;CONTAINS 'BAD' DATA
001230  000000                  .WORD   0       ;;RESERVED--NOT TO BE USED
001232  000000                  .WORD   0
001234  000            $AUTOB:  .BYTE   0       ;;AUTOMATIC MODE INDICATOR
001235  000            $INTAG:  .BYTE   0       ;;INTERRUPT MODE INDICATOR
001236  000000                  .WORD   0
001240  177570         $SWR:    .WORD   DSWR    ;;ADDRESS OF SWITCH REGISTER
001242  177570         DISPLAY: .WORD   DDISP   ;;ADDRESS OF DISPLAY REGISTER
001244  177560         $TKS:    177560          ;;TTY KBD STATUS
001246  177562         $TKB:    177562          ;;TTY KBD BUFFER
001250  177564         $TPS:    177564          ;;TTY PRINTER STATUS REG. ADDRESS
001252  177566         $TPB:    177566          ;;TTY PRINTER BUFFER REG. ADDRESS
001254  000            $NULL:   .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
001255  002            $FILLS:  .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001256  012            $FILLC:  .BYTE   12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001257  000            $TPFLG:  .BYTE   0       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001260  000000         $REGAD:  .WORD   0       ;;CONTAINS THE ADDRESS FROM
                                                ;;WHICH  ($REG0) WAS OBTAINED
001262  000000         $REG0:   .WORD   0       ;;CONTAINS (($REGAD)+0)
001264  000000         $REG1:   .WORD   0       ;;CONTAINS (($REGAD)+2)
001266  000000         $REG2:   .WORD   0       ;;CONTAINS (($REGAD)+4)
001270  000000         $REG3:   .WORD   0       ;;CONTAINS (($REGAD)+6)
001272  000000         $REG4:   .WORD   0       ;;CONTAINS (($REGAD)+10)
001274  000000         $REG5:   .WORD   0       ;;CONTAINS (($REGAD)+12)
001276  000000         $TMP0:   .WORD   0       ;;USER DEFINED
001300  000000         $TMP1:   .WORD   0       ;;USER DEFINED
001302  000000         $TMP2:   .WORD   0       ;;USER DEFINED
001304  000000         $TMP3:   .WORD   0       ;;USER DEFINED
001306  000000         $TMP4:   .WORD   0       ;;USER DEFINED
001310  000000         $TIMES:  0               ;;MAX. NUMBER OF ITERATIONS
001312  077            $QUES:   .ASCII  /?/     ;;QUESTION MARK
001313  015            $CRLF:   .ASCII  <15>    ;;CARRIAGE RETURN
001314  000012         $LF:     .ASCIZ  <12>    ;;LINE FEED
                        ;;*****************************************************************
                        .SBTTL  APT MAILBOX-ETABLE
```

```
                                ;;*****************************************************************
                                .EVEN
                                $MAIL:                          ;;APT MAILBOX
001316                          $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001316  000000                  $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001320  000000                  $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001322  000000                  $PASS:  .WORD   APASS           ;;PASS COUNT
001324  000000                  $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
001326  000000                  $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001330  000000                  $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
001332  000000                  $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
001334  000000                  $ETABLE:                        ;;APT ENVIRONMENT TABLE
001336                          $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
001336     002                  $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
001337     000                  $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
001340  000000                  $USWR:  .WORD   AUSWR           ;;USER SWITCHES
001342  000000                  $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
001344  000000                                                  ;*
                                                                ;*  BITS 15-11=CPU TYPE
                                                                ;*          11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                                                ;*          11/70=06,PDQ=07,Q=10
                                                                ;*  BIT 10=REAL TIME CLOCK
                                                                ;*  BIT  9=FLOATING POINT PROCESSOR
                                                                ;*  BIT  8=MEMORY MANAGEMENT
001346     000                  $MAMS1: .BYTE   AMAMS1          ;;HIGH ADDRESS,M.S. BYTE
001347     000                  $MTYP1: .BYTE   AMTYP1          ;;MEM. TYPE,BLK#1
                                                                ;*  MEM.TYPE BYTE   --  (HIGH BYTE)
                                                                ;*          900 NSEC CORE=001
                                                                ;*          300 NSEC BIPOLAR=002
                                                                ;*          500 NSEC MOS=003
001350  000000                  $MADR1: .WORD   AMADR1          ;;HIGH ADDRESS,BLK#1
                                                                ;*  MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001352     000                  $MAMS2: .BYTE   AMAMS2          ;;HIGH ADDRESS,M.S. BYTE
001353     000                  $MTYP2: .BYTE   AMTYP2          ;;MEM.TYPE,BLK#2
001354  000000                  $MADR2: .WORD   AMADR2          ;;MEM.LAST ADDRESS,BLK#2
001356     000                  $MAMS3: .BYTE   AMAMS3          ;;HIGH ADDRESS,M.S.BYTE
001357     000                  $MTYP3: .BYTE   AMTYP3          ;;MEM.TYPE,BLK#3
001360  000000                  $MADR3: .WORD   AMADR3          ;;MEM.LAST ADDRESS,BLK#3
001362     000                  $MAMS4: .BYTE   AMAMS4          ;;HIGH ADDRESS,M.S.BYTE
001363     000                  $MTYP4: .BYTE   AMTYP4          ;;MEM.TYPE,BLK#4
001364  000000                  $MADR4: .WORD   AMADR4          ;;MEM.LAST ADDRESS,BLK#4
001366  000000                  $VECT1: .WORD   AVECT1          ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001370  000000                  $VECT2: .WORD   AVECT2          ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001372  000000                  $BASE:  .WORD   ABASE           ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001374  000000                  $DEVM:  .WORD   ADEVM           ;;DEVICE MAP
001376  000000                  $CDW1:  .WORD   ACDW1           ;;CONTROLLER DESCRIPTION WORD#1
001400  000000                  $CDW2:  .WORD   ACDW2           ;;CONTROLLER DESCRIPTION WORD#2
001402  000000                  $DDW0:  .WORD   ADDW0           ;;DEVICE DESCRIPTOR WORD#0
001404  000000                  $DDW1:  .WORD   ADDW1           ;;DEVICE DESCRIPTOR WORD#1
001406  000000                  $DDW2:  .WORD   ADDW2           ;;DEVICE DESCRIPTOR WORD#2
001410  000000                  $DDW3:  .WORD   ADDW3           ;;DEVICE DESCRIPTOR WORD#3
001412  000000                  $DDW4:  .WORD   ADDW4           ;;DEVICE DESCRIPTOR WORD#4
001414  000000                  $DDW5:  .WORD   ADDW5           ;;DEVICE DESCRIPTOR WORD#5
001416  000000                  $DDW6:  .WORD   ADDW6           ;;DEVICE DESCRIPTOR WORD#6
001420  000000                  $DDW7:  .WORD   ADDW7           ;;DEVICE DESCRIPTOR WORD#7
001422  000000                  $DDW8:  .WORD   ADDW8           ;;DEVICE DESCRIPTOR WORD#8
```

```
                                ;;DEVICE DESCRIPTOR WORD#9
                                ;;DEVICE DESCRIPTOR WORD#10
                                ;;DEVICE DESCRIPTOR WORD#11
                                ;;DEVICE DESCRIPTOR WORD#12
                                ;;DEVICE DESCRIPTOR WORD#13
                                ;;DEVICE DESCRIPTOR WORD#14
                                ;;DEVICE DESCRIPTOR WORD#15

                                ;
                                ;  PROGRAM CONTROL PARAMETERS
                                ;-------------------------------
                                ;ADDRSS OF NEXT TEST TO BE EXECUTED
                                ;ADDRESS FOR LOCK CURRENT DATA
                                ;
                                ;   PROGRAM VARIABLES
                                ;-----------------------
                                ;SWITCHES AT START OF PROGRAM
                                ;KM STATUS WORD STORAGE
                                ;
                                ;
                                ;STACK POINTER STORAGE
                                ;PROGRAM COUNTER STORAGE
                                ;
                                ;
                                ;HIGHEST LOCATION FOR NPR'S
                                ;KMC11 SELECTED ACTIVE
                                ;OCTAL NUMBER OF KMC11'S
                                ;ORIGINAL ACTIVE DEVICES.
                                ;WORKABLE NUMBER.
                                ;POINTER TO RUNNING DEVICES

                                ;TABLE POINTER
                                ;TABLE POINTER
                                ;
                                ;  PROGRAM CONTROL FLAGS
                                ; ----------------------
                                ;PROGRAM INITIALIZING FLAG

                                ;LOCK ON CURRENT TEST FLAG
                                ;QUICK VERIFY FLAG
                                ;ON FIRST PASS OF EACH KMC11 ITERATIONS WILL BE SUPPRES
```

```
                        .SBTTL  ERROR POINTER TABLE

                        ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                        ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                        ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                        ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                        ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                        ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                        ;*      DH              ;;POINTS TO THE DATA HEADER
                        ;*      DT              ;;POINTS TO THE DATA
                        ;*      DF              ;;POINTS TO THE DATA FORMAT


001512                  $ERRTB:
                        .EVEN
                        ;*      DF              ;; DOES NOT APPLY IN THIS DIAGNOSTIC.
001512  000000                  0
001514  000000                  0
001516  000000                  0
001520  035322                  EM1
001522  036326                  DH2             ; ERROR 1
001524  036642                  DT2
001526  035360                  EM2
001530  036326                  DH2             ; ERROR 2
001532  036642                  DT2
001534  035423                  EM3
001536  036326                  DH2             ; ERROR 3
001540  036642                  DT2
001542  035467                  EM4
001544  000000                  0               ; ERROR 4
001546  000000                  0
001550  035531                  EM5
001552  036326                  DH2             ; ERROR 5
001554  036642                  DT2
001556  035531                  EM5
001560  036364                  DH3             ; ERROR 6
001562  036660                  DT3
001564  035561                  EM6
001566  036305                  DH1             ; ERROR 7
001570  036630                  DT1
001572  035600                  EM7
001574  036305                  DH1             ; ERROR 10
001576  036630                  DT1
001600  035625                  EM10
001602  036305                  DH1             ; ERROR 11
001604  036630                  DT1
001606  035651                  EM11
001610  036462                  DH5             ; ERROR 12
001612  036704                  DT5
001614  035700                  EM12
001616  036462                  DH5             ; ERROR 13
001620  036704                  DT5
001622  035651                  EM11
001624  036422                  DH4             ; ERROR 14
001626  036672                  DT4
```

```
421  001630  035724          EM13
422  001632  000000          0                   ; ERROR 15
423  001634  000000          0
424  001636  035651          EM11
425  001640  036462          DH5                 ; ERROR 16
426  001642  036722          DT6
427  001644  035700          EM12
428  001646  036462          DH5                 ; ERROR 17
429  001650  036722          DT6
430  001652  035651          EM11
431  001654  036514          DH6                 ; ERROR 20
432  001656  036740          DT7
433  001660  035651          EM11
434  001662  036514          DH6                 ; ERROR 21
435  001664  036762          DT10
436  001666  035700          EM12
437  001670  036514          DH6                 ; ERROR 22
438  001672  036740          DT7
439  001674  035700          EM12
440  001676  036514          DH6                 ; ERROR 23
441  001700  036762          DT10
442  001702  035764          EM14
443  001704  000000          0                   ; ERROR 24
444  001706  000000          0
445  001710  036034          EM15
446  001712  036305          DH1                 ; ERROR 25
447  001714  036630          DT1
448  001716  036055          EM16
449  001720  036364          DH3                 ; ERROR 16
450  001722  037004          DT11
451  001724  035700          EM12
452  001726  036305          DH1                 ; ERROR 27
453  001730  037016          DT12
454  001732  036071          EM17
455  001734  000000          0                   ; ERROR 30
456  001736  000000          0
457  001740  036135          EM20
458  001742  036305          DH1                 ; ERROR 31
459  001744  036630          DT1
460  001746  036156          EM21
461  001750  036562          DH7                 ; ERROR 32
462  001752  000000          0
463  001754  036156          EM21
464  001756  036364          DH3                 ; ERROR 33
465  001760  036660          DT3
466  001762  036173          EM22
467  001764  036605          DH10                ; ERROR 34
468  001766  000000          0
469  001770  036216          EM23
470  001772  036326          DH2                 ; ERROR 35
471  001774  036642          DT2
472  001776  036240          EM24
473  002000  000000          0                   ; ERROR 36
474  002002  000000          0
475  002004  036263          EM25
476  002006  000000          0                   ; ERROR 37
```

```
000000                  0
035561                  EM6
036326                  DH2             ; ERROR 40
036642                  DT2
035531                  EM5
036462                  DH5             ; ERROR 41
036704                  DT5
035724                  EM13
036305                  DH1             ; ERROR 42
036630                  DT1
002034          .=2034
                .SBTTL  APT PARAMETER BLOCK

                ;;*****************************************************************
                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                ;;*****************************************************************
002034                  .$X=.   ;;SAVE CURRENT LOCATION
000024                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000200                  200     ;;FOR APT START UP
000044                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
002034                  $APTHDR ;;POINT TO APT HEADER BLOCK
002034                  .=.$X   ;;RESET LOCATION COUNTER
                ;;*****************************************************************
                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                ;INTERFACE SPEC.

002034          $APTHD:
002034  000000  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
002036  001316  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
002040  000132  $TSTM:  .WORD   90.     ;;RUN TIM OF LONGEST TEST
002042  000137  $PASTM: .WORD   95.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
002044  000137  $UNITM: .WORD   95.     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
002046  000052          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                ;KMC11 CONTROL INDICATORS FOR CURRENT KMC11 UNDER TEST
                                ;-----------------------------------------------------

002050  000000                  STAT1:  0
002052  000000                  STAT2:  0
002054  000000                  STAT3:  0

                                ;KMC11 VECTOR AND REGISTER INDIRECT POINTERS
                                ;-------------------------------------------

002056  000000                  KMRVEC: 0               ;POINTER TO KMC11 RECEIVER INTERRUPT VECTOR
002060  000000                  KMRLVL: 0               ;POINTER TO KMC11 RECEIVER INTERRUPT SERVICE PS
002062  000000                  KMTVEC: 0               ;POINTER TO KMC11 TRANSMITTER INTERRUPT VECTOR
002064  000000                  KMTLVL: 0               ;POINTER TO KMC11 TRANSMITTER INTERRUPT SERVICE PS
002066  000000                  KMCSR:  0               ;POINTER TO KMC11 CONTROL STATUS REGISTER
002070  000000                  KMCSRH: 0               ;POINTER TO KMC11 CONTROL STATUS REGISTER HIGH BYTE.
002072  000000                  KMCTL:  0               ;POINTER TO KMC11 CONTOL OUT REGISTER
002074  000000                  KMP04:  0               ;POINTER TO KMC11 PORT REGISTER(SEL 4)
002076  000000                  KMP06:  0               ;POINTER TO KMC11 PORT REGISTER(SEL 6)

                                ;TEMP STORAGE
                                ;------------

                                ;TEMP:  0
                                ;.=.+40

                                ;KMC11 STATUS TABLE AND ADDRESS ASSIGNMENTS
                                ;------------------------------------------

        002100                  .=2100
002100                          KM.MAP:
002100  000001                  KMCR00: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 00
002102  000001                  KMS100: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 00
002104  000001                  KMS200: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 00
002106  000001                  KMS300: .BLKW   1               ;3RD STATUS WORD

002110  000001                  KMCR01: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 01
002112  000001                  KMS101: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 01
002114  000001                  KMS201: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 01
002116  000001                  KMS301: .BLKW   1               ;3RD STATUS WORD

002120  000001                  KMCR02: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 02
002122  000001                  KMS102: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 02
002124  000001                  KMS202: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 02
002126  000001                  KMS302: .BLKW   1               ;3RD STATUS WORD

002130  000001                  KMCR03: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 03
002132  000001                  KMS103: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 03
002134  000001                  KMS203: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 03
002136  000001                  KMS303: .BLKW   1               ;3RD STATUS WORD

002140  000001                  KMCR04: .BLKW   1               ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 04
002142  000001                  KMS104: .BLKW   1               ;VECTOR FOR KMC11 NUMBER 04
002144  000001                  KMS204: .BLKW   1               ;DDCMP LINE# FOR KMC11 NUMBER 04
002146  000001                  KMS304: .BLKW   1               ;3RD STATUS WORD
```

```
                        
002150  000001          KMCR05: .BLKW   1       ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 05
002152  000001          KMS105: .BLKW   1       ;VECTOR FOR KMC11 NUMBER 05
002154  000001          KMS205: .BLKW   1       ;DDCMP LINE# FOR KMC11 NUMBER 05
002156  000001          KMS305: .BLKW   1       ;3RD STATUS WORD
                        
002160  000001          KMCR06: .BLKW   1       ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 06
002162  000001          KMS106: .BLKW   1       ;VECTOR FOR KMC11 NUMBER 06
002164  000001          KMS206: .BLKW   1       ;DDCMP LINE# FOR KMC11 NUMBER 06
002166  000001          KMS306: .BLKW   1       ;3RD STATUS WORD
                        
002170  000001          KMCR07: .BLKW   1       ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 07
002172  000001          KMS107: .BLKW   1       ;VECTOR FOR KMC11 NUMBER 07
002174  000001          KMS207: .BLKW   1       ;DDCMP LINE# FOR KMC11 NUMBER 07
002176  000001          KMS307: .BLKW   1       ;3RD STATUS WORD
                        
002200  000001          KMCR10: .BLKW   1       ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 10
002202  000001          KMS110: .BLKW   1       ;VECTOR FOR KMC11 NUMBER 10
002204  000001          KMS210: .BLKW   1       ;DDCMP LINE# FOR KMC11 NUMBER 10
002206  000001          KMS310: .BLKW   1       ;3RD STATUS WORD
                        
002210  000001          KMCR11: .BLKW   1       ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 11
002212  000001          KMS111: .BLKW   1       ;VECTOR FOR KMC11 NUMBER 11
002214  000001          KMS211: .BLKW   1       ;DDCMP LINE# FOR KMC11 NUMBER 11
002216  000001          KMS311: .BLKW   1       ;3RD STATUS WORD
                        
002220  000001          KMCR12: .BLKW   1       ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 12
002222  000001          KMS112: .BLKW   1       ;VECTOR FOR KMC11 NUMBER 12
002224  000001          KMS212: .BLKW   1       ;DDCMP LINE# FOR KMC11 NUMBER 12
002226  000001          KMS312: .BLKW   1       ;3RD STATUS WORD
                        
002230  000001          KMCR13: .BLKW   1       ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 13
002232  000001          KMS113: .BLKW   1       ;VECTOR FOR KMC11 NUMBER 13
002234  000001          KMS213: .BLKW   1       ;DDCMP LINE# FOR KMC11 NUMBER 13
002236  000001          KMS313: .BLKW   1       ;3RD STATUS WORD
                        
002240  000001          KMCR14: .BLKW   1       ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 14
002242  000001          KMS114: .BLKW   1       ;VECTOR FOR KMC11 NUMBER 14
002244  000001          KMS214: .BLKW   1       ;DDCMP LINE# FOR KMC11 NUMBER 14
002246  000001          KMS314: .BLKW   1       ;3RD STATUS WORD
                        
002250  000001          KMCR15: .BLKW   1       ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 15
002252  000001          KMS115: .BLKW   1       ;VECTOR FOR KMC11 NUMBER 15
002254  000001          KMS215: .BLKW   1       ;DDCMP LINE# FOR KMC11 NUMBER 15
002256  000001          KMS315: .BLKW   1       ;3RD STATUS WORD
                        
002260  000001          KMCR16: .BLKW   1       ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 16
002262  000001          KMS116: .BLKW   1       ;VECTOR FOR KMC11 NUMBER 16
002264  000001          KMS216: .BLKW   1       ;DDCMP LINE# FOR KMC11 NUMBER 16
002266  000001          KMS316: .BLKW   1       ;3RD STATUS WORD
                        
002270  000001          KMCR17: .BLKW   1       ;CONTROL STATUS REGISTER FOR KMC11 NUMBER 17
002272  000001          KMS117: .BLKW   1       ;VECTOR FOR KMC11 NUMBER 17
002274  000001          KMS217: .BLKW   1       ;DDCMP LINE# FOR KMC11 NUMBER 17
002276  000001          KMS317: .BLKW   1       ;3RD STATUS WORD
                        
```

623 002300 000000 KM.END: 000000

```
                        ;KMC11 PASS COUNT AND ERROR COUNT TABLE
                        ;--------------------------------------

002302                  CNT.MAP:
002302  000000          PACT00: 0               ;PASS COUNT FOR KMC11 NUMBER 00
002304  000000          ERCT00: 0               ;ERROR COUNT FOR KMC11 NUMBER 00

002306  000000          PACT01: 0               ;PASS COUNT FOR KMC11 NUMBER 01
002310  000000          ERCT01: 0               ;ERROR COUNT FOR KMC11 NUMBER 01

002312  000000          PACT02: 0               ;PASS COUNT FOR KMC11 NUMBER 02
002314  000000          ERCT02: 0               ;ERROR COUNT FOR KMC11 NUMBER 02

002316  000000          PACT03: 0               ;PASS COUNT FOR KMC11 NUMBER 03
002320  000000          ERCT03: 0               ;ERROR COUNT FOR KMC11 NUMBER 03

002322  000000          PACT04: 0               ;PASS COUNT FOR KMC11 NUMBER 04
002324  000000          ERCT04: 0               ;ERROR COUNT FOR KMC11 NUMBER 04

002326  000000          PACT05: 0               ;PASS COUNT FOR KMC11 NUMBER 05
002330  000000          ERCT05: 0               ;ERROR COUNT FOR KMC11 NUMBER 05

002332  000000          PACT06: 0               ;PASS COUNT FOR KMC11 NUMBER 06
002334  000000          ERCT06: 0               ;ERROR COUNT FOR KMC11 NUMBER 06

002336  000000          PACT07: 0               ;PASS COUNT FOR KMC11 NUMBER 07
002340  000000          ERCT07: 0               ;ERROR COUNT FOR KMC11 NUMBER 07

002342  000000          PACT10: 0               ;PASS COUNT FOR KMC11 NUMBER 10
002344  000000          ERCT10: 0               ;ERROR COUNT FOR KMC11 NUMBER 10

002346  000000          PACT11: 0               ;PASS COUNT FOR KMC11 NUMBER 11
002350  000000          ERCT11: 0               ;ERROR COUNT FOR KMC11 NUMBER 11

002352  000000          PACT12: 0               ;PASS COUNT FOR KMC11 NUMBER 12
002354  000000          ERCT12: 0               ;ERROR COUNT FOR KMC11 NUMBER 12

002356  000000          PACT13: 0               ;PASS COUNT FOR KMC11 NUMBER 13
002360  000000          ERCT13: 0               ;ERROR COUNT FOR KMC11 NUMBER 13

002362  000000          PACT14: 0               ;PASS COUNT FOR KMC11 NUMBER 14
002364  000000          ERCT14: 0               ;ERROR COUNT FOR KMC11 NUMBER 14

002366  000000          PACT15: 0               ;PASS COUNT FOR KMC11 NUMBER 15
002370  000000          ERCT15: 0               ;ERROR COUNT FOR KMC11 NUMBER 15

002372  000000          PACT16: 0               ;PASS COUNT FOR KMC11 NUMBER 16
002374  000000          ERCT16: 0               ;ERROR COUNT FOR KMC11 NUMBER 16

002376  000000          PACT17: 0               ;PASS COUNT FOR KMC11 NUMBER 17
002400  000000          ERCT17: 0               ;ERROR COUNT FOR KMC11 NUMBER 17
```

677
678
679
680
681
682

FORMAT OF STATUS TABLE

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

DEFINITION OF FORMAT

CSR: CONTAINS KMC11 CSR ADDRESS

STAT1: BITS 00-08 IS KMC11 VECTOR ADDRESS
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 KMC11 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 [PROGRAMS G AND H ONLY])

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

002402  012737  000340  177776  .START: MOV     #340,PS         ;LOCK OUT INTERRUPTS
002410  012706  001200                  MOV     #STACK,SP       ;SET UP STACK
002414  012737  007122  000024          MOV     #$PWRDN,@#24    ;SET UP POWER FAIL VECTOR
002422  013737  001472  001476          MOV     KMNUM,SAVNUM    ;SAVE NUMBER OF DEVICES IN SYSTEM.
002430  005037  001416                  CLR     SWFLG           ;CLEAR SOFT TYPEOUT FLAG
002434  105037  001203                  CLRB    $ERFLG          ;CLEAR ERROR FLAG
002440  105037  001511                  CLRB    QV.FLG          ;ZERO QUICK VERIFY FLAG
002444  012737  002070  001502          MOV     #KM.MAP-10,CREAM;GET MAP POINTER.
002452  012737  002276  001504          MOV     #CNT.MAP-4,MILK ;GET PASS COUNT MAP POINTER
002460  012737  100000  001500          MOV     #BIT15,RUN      ;POINT POINTER TO FIRST DEVICE.
002466  012700  002302                  MOV     #CNT.MAP,R0     ;PASS COUNT POINTER TO R0
002472  005020                  23$:     CLR     (R0)+           ;CLEAR TABLE
002474  022700  002402                  CMP     #CNT.MAP+100,R0 ;DONE YET?
002500  001374                          BNE     23$             ;KEEP GOING
002502  005037  001216                  CLR     $ERRPC          ;CLEAR LAST ERROR POINTER
002506  012737  000001  001202          MOV     #1,$TSTNM       ;SET UP FOR TEST 1
002514  012737  002402  001206          MOV     #.START,$LPADR  ;SET UP FOR POWER FAIL BEFORE
                                        ;TESTING STARTS
002522  132737  000001  001336          BITB    #1,$ENV         ; IS IT RUNNING UNDER APT?
002530  001404                          BEQ     3$              ; IF NOT CHECK FOR TYPE OF SWITCH REGISTER.
002532  013737  001340  000176          MOV     $SWREG,SWREG    ; LOAD SOFTWARE SWITCH REG.
002540  000423                          BR      6$+2            ; GO SET UP SOFTWARE SWITCH REG.
002542  013746  000006          3$:     MOV     @#6,-(SP)       ;SAVE CURRENT VECTORS
002546  013746  000004                  MOV     @#4,-(SP)       ;
002552  012737  002606  000004          MOV     #6$,@#4         ;SET UP FOR TIMEOUT
002560  012737  177570  001240          MOV     #177570,SWR     ;SET SWR TO HARD SWR ADDRESS
002566  012737  177570  001242          MOV     #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
002574  022777  177777  176436          CMP     #-1,@SWR        ;REFERENCE HARDWARE SWITCH REGISTER
002602  001402                          BEQ     6$+2            ;IF = -1 USE SOFT SWR ANYWAY
002604  000407                          BR      7$              ;IF IT EXISTS AND NOT = -1 USE HARD SWR
002606  022626                  6$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
002610  012737  000176  001240          MOV     #SWREG,SWR      ;POINTER TO SOFT SWR
002616  012737  000174  001242          MOV     #DISPREG,DISPLAY;POINTER TO SOFT DISPLAY REG
002624  012637  000004          7$:     MOV     (SP)+,@#4       ;RESTORE VECTORS
002630  012637  000006                  MOV     (SP)+,@#6       ;
002634  105737  001506                  TSTB    INIFLG          ;HAS INITIALIZATION BEEN PERFORMED
002640  001006                          BNE     20$             ;BR IF YES
002642  022737  004070  000042          CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
002650  001402                          BEQ     20$
002652  104401  001000                  TYPE    ,MTITLE         ;TYPE TITLE MESSAGE
002656  004737  011212          20$:    JSR     PC,CKSWR                ;CHECK FOR SOFT SWR
002662  017737  176352  001446          MOV     @SWR,STRTSW     ;STORE STARTING SWITCHES
002670  005737  000042                  TST     @#42            ;IS IT RUNNING IN AUTO MODE?
002674  001402                          BEQ     .+6             ;BR IF NO
002676  005037  001446                  CLR     STRTSW          ;IF YES, CLEAR SWITCHES
002702  032737  000001  001446          BIT     #SW00,STRTSW    ;IF SW00=1, QUESTIONS ARE ASKED.
002710  001012                          BNE     17$             ;BR IF SW00=1
002712  105737  001446                  TSTB    STRTSW          ;BIT7=1??
```

```
 787  002716  100007                        BPL     17$             ;BR IF SW07=0
 788  002720  005737  001470                TST     RMACTV          ;ARE ANY DEVICES SELECTED?
 789  002724  001027                        BNE     16$             ;BR IF YES
 790  002726  104401  010731                TYPE,   NOACT           ;NO DEVICES SELECTED.
 791  002732  000000                        HALT                    ;STOP THE SHOW
 792  002734  000776                        BR      .-2             ;DISQUALIFY CONTINUE SWITCH
 793  002736  105737  001336        17$:    TSTB    $ENV            ; IS IT UNDER APT DUMP MODE?
 794  002742  001405                        BEQ     27$             ; YES, CHECK IF APT SIZED IT?
 795  002744  132737  000001  001336        BITB    #1,$ENV         ; IS IT UNDER Q,V OR RUN MODE?
 796  002752  001012                        BNE     30$             ; YES, NEEDS ONLY APT SIZING.
 797  002754  000406                        BR      33$             ; NO, NEEDS REGULAR AUTO.SIZE.
 798  002756  105737  001337        27$:    TSTB    $ENVM           ; IS IT SIZED BY APT?
 799  002762  100406                        BMI     30$             ; YES, NEEDS ONLY APT SIZING.
 800  002764  042737  000001  001446        BIC     #SW00,STRTSW    ; SIZE ONLY IN AUTO MODE.
 801  002772  004737  012110        33$:    JSR     PC,AUTO.SIZE    ; GO DO THE AUTO.SIZE.
 802  002776  000402                        BR      16$             ; GO PRINT THE MAP.
 803  003000  004737  013510        30$:    JSR     PC,APT.SIZE     ; GO DO THE APT SIZING.
 804  003004  105737  001506        16$:    TSTB    INIFLG          ;FIRST TIME?
 805  003010  001410                        BEQ     21$             ;BR IF YES
 806  003012  105737  001446                TSTB    STRTSW          ;IF USING SAME PARAMETERS DONT TYPE MAP
 807  003016  100431                        BMI     1$
 808  003020  032737  000006  001446        BIT     #BIT1!BIT2,STRTSW;IS TEST NO. OR LOCK SELECTED
 809  003026  001403                        BEQ     24$             ;IF NO THEN TYPE STATUS
 810  003030  000424                        BR      1$              ;IF YES DO NOT TYPE STATUS
 811  003032  105137  001506        21$:    COMB    INIFLG          ;SET FLAG
 812  003036  104401  010077        24$:    TYPE    ,XHEAD          ;TYPE HEADER
 813  003042  012704  002100                MOV     #KM.MAP,R4      ;SET POINTER
 814  003046  010437  001276        5$:     MOV     R4,$TMP0        ;SET ADDRESS
 815  003052  012437  001300                MOV     (R4)+,$TMP1     ;SET CSR
 816  003056  001411                        BEQ     1$              ;ALL DONE IF ZERO
 817  003060  012437  001302                MOV     (R4)+,$TMP2     ;SET STAT1
 818  003064  012437  001304                MOV     (R4)+,$TMP3     ;SET STAT2
 819  003070  012437  001306                MOV     (R4)+,$TMP4     ;SET STAT3
 820  003074  104416                        CONVRT                  ;TYPE OUT STATUS MAP
 821  003076  011060                        XSTATQ                  ;
 822  003100  000762                        BR      5$
 823  003102  012700  002100        1$:     MOV     #KM.MAP,R0      ;R0 POINTS TO STATUS TABLE
 824
 825                                        ;;**************************************************************************
 826                                        ;;*AUTO SIZE TEST
 827                                        ;;*THIS TEST VERIFYS THAT THE KMC11S AND/OR KMC11S ARE AT THE CORRECT FLOATING
 828                                        ;;*ADDRESSES FOR YOUR SYSTEM. IF THIS TEST FAILS, IT IS NOT A HARDWARE ERROR.
 829                                        ;;*CHECK THE ADDRESSES OF ALL FLOATING DEVICES (DJ,DH,DQ,DU,DUP,LK,DMC,DZ,KMC).
 830                                        ;;*IF THERE ARE NO OTHER FLOATING DEVICES BEFORE THE KMC11, THE FIRST
 831                                        ;;* KMC11 IS 760110. NO DEVICE SHOULD EVER BE AT
 832                                        ;;*ADDRESS 760000.
 833                                        ;;**************************************************************************
 834
 835  003106  013746  000004                MOV     @#4,-(SP)       ;SAVE LOC 4
 836  003112  013746  000006                MOV     @#6,-(SP)       ;SAVE LOC 6
 837  003116  005037  000006                CLR     @#6             ;CLEAR VEC+2
 838  003122  005037  001302                CLR     $TMP2           ;CLEAR FLAG
 839  003126  011037  002066        AUSTRT: MOV     (R0),KMCSR      ;GET NEXT KMC CSR
 840  003132  001510                        BEQ     AUDONE          ;BR IF DONE
 841  003134  012737  003240  000004  2$:     MOV     #NODEV,@#4      ;SET UP FOR TIMEOUT
 842  003142  012703  000010        3$:     MOV     #10,R3          ;R3 IS COUNT OF DEVICES BEFORE KMC
```

```
 843  003146  012702  003342          4$:     MOV     #DEVTAB,R2      ;R2 IS DEVICE TABLE PONTER
 844  003152  012701  160010                  MOV     #160010,R1      ;START WITH ADDRESS 160010
 845  003156  005711                 FLOAT:  TST     (R1)            ;CHECK ADDRESS IN R1
 846  003160  111204                         MOVB    (R2),R4         ;IF NO TIMEOUT, GET NEXT ADDRESS
 847  003162  060401                         ADD     R4,R1           ;IN R1
 848  003164  005201                         INC     R1              ;
 849  003166  040401                         BIC     R4,R1           ;
 850  003170  005703                         TST     R3              ;ANY MORE DEVICES TO CHECK FOR?
 851  003172  001371                         BNE     FLOAT           ;BR IF YES
 852  003174  012737  003244  000004         MOV     #ERR,@#4        ;OK ONLY KMC'S ARE LEFT, SET UP FOR TIMEOUT
 853  003202  005711                 FY:     TST     (R1)            ;CHECK KMC ADDRESS
 854  003204  020137  002066                 CMP     R1,KMCSR        ;DOES IT MATCH
 855  003210  001403                         BEQ     OK              ;BR IF YES
 856  003212  062701  000010                 ADD     #10,R1          ;GET NEXT KMC ADDRESS
 857  003216  000771                         BR      FY              ;DO IT AGAIN
 858  003220  062700  000010          OK:    ADD     #10,R0          ;SKIP TO NEXT KMC CSR
 859  003224  062701  000010                 ADD     #10,R1          ; GET NEXT KMC ADDRESS
 860  003230  011037  002066                 MOV     (R0),KMCSR      ; GET NEXT KMC CSR
 861  003234  001447                         BEQ     ALDONE          ; BRANCH IF ALL DONE.
 862  003236  000761                         BR      FY              ; DO IT AGAIN.
 863  003240  122243                 NODEV:  CMPB    (R2)+,-(R3)     ;ON TIMEOUT, INC R2, DEC R3
 864  003242  000002                         RTI                     ;$LPADR
 865  003244  005737  001302         ERR:    TST     $TMP2           ;CHECK FLAG IF = 0 TYPE HEADER
 866  003250  001014                         BNE     1$              ;SKIP HEADER
 867  003252  104401                         TYPE                    ;TYPEOUT HEADER MESSAGE
 868  003254  010762                         CONERR                  ;CONFIGURATION ERROR!!!!
 869  003256  012737  003244  001460         MOV     #ERR,SAVPC      ;SAVE PC FOR TYPEOUT
 870  003264  104417                         CNVRT                   ;TYPE OUT ERROR PC
 871  003266  003322                         ERRPC                   ;
 872  003270  104401                         TYPE                    ;TYPE REST OF HEADER
 873  003272  011027                         CNERR                   ;
 874  003274  012737  177777  001302         MOV     #-1,$TMP2       ;SET FLAG SO IT ONLY GETS TYPED ONCE
 875  003302  010137  001264          1$:    MOV     R1,$REG1        ;SAVE R1 FOR TYPEOUT
 876  003306  104416                         CONVRT
 877  003310  003330                         CONTAB                  ;TYPE CSR VALUES
 878  003312  104401                  3$:    TYPE
 879  003314  011050                         KMCM
 880  003316  022626                  4$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
 881  003320  000737                         BR      OK              ;BR TO GET OUT
 882  003322  000001                 ERRPC:  1
 883  003324     006      002                .BYTE   6,2
 884  003326  001460                         SAVPC
 885  003330  000002                 CONTAB: 2
 886  003332     006      004                .BYTE   6,4
 887  003334  001264                         $REG1
 888  003336     006      002                .BYTE   6,2
 889  003340  002066                         KMCSR
 890  003342     007                 DEVTAB: .BYTE   7               ;DJ
 891  003343     017                         .BYTE   17              ;DH
 892  003344     007                         .BYTE   7               ;DQ
 893  003345     007                         .BYTE   7               ;DU
 894  003346     007                         .BYTE   7               ;DUP
 895  003347     007                         .BYTE   7               ;LK
 896  003350     007                         .BYTE   7               ;DMC
 897  003351     007                         .BYTE   7               ;DZ
 898  003352     007                         .BYTE   7               ;KMC
```

```
899                 003354                          .EVEN
900  003354                                         AUDONE:
901  003354  012637  000006                 1$:     MOV     (SP)+,@#6       ;RESTORE LOC 6
902  003360  012637  000004                         MOV     (SP)+,@#4       ;RESTORE LOC 4
903  003364  032737  000010  001446                 BIT     #SW03,STRTSW    ;SELECT SPECIFIC DEVICES??
904  003372  001422                                 BEQ     3$              ;BR IF NO.
905  003374  104401  010017                         TYPE    ,MNEW           ;TYPE THE MESSAGE.
906  003400  005000                                 CLR     R0              ;ZERO DATA LIGHTS
907  003402  000000                                 HALT                    ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
908  003404  027737  175630  001474                 CMP     @SWR,SAVACT     ;IS THE NUMBER VALID?
909  003412  101404                                 BLOS    2$              ;BR IF NUMBER IS OK.
910  003414  104401  007672                         TYPE    ,MERR3          ;TELL USER OF INVALID NUMBER.
911  003420  000000                                 HALT                    ;STOP EVERY THING.
912  003422  000776                                 BR      .-2             ;RESTART THE PROGRAM AGAIN.
913  003424  017737  175610  001470         2$:     MOV     @SWR,KMACTV     ;GET NEW DEVICE PATTERN
914  003432  013700  001470                         MOV     KMACTV,R0       ;SHOW THE USER WHAT HE SELECTED.
915  003436  000000                                 HALT                    ;CONTINUE DYNAMIC SWITCHES.
916  003440  012700  000300                 3$:     MOV     #300,R0         ;PREPARE TO CLEAR THE FLOATING
917  003444  012701  000302                         MOV     #302,R1         ;VECTOR AREA. 300-776
918  003450  010120                         4$:     MOV     R1,(R0)+        ;START PUTTING "PC+2 - HALT"
919  003452  005021                                 CLR     (R1)+           ;IN VECTOR AREA.
920  003454  022021                                 CMP     (R0)+,(R1)+     ;POP POINTERS
921  003456  022700  001000                         CMP     #1000,R0        ;ALL DONE??
922  003462  001372                                 BNE     4$              ;BR IF NO.
923
924                                         ;TEST START AND RESTART
925                                         ;----------------------
926
927  003464  012706  001200                 .BEGIN: MOV     #STACK,SP       ;SET UP STACK
928  003470  013746  000006                         MOV     @#6,-(SP)       ;SAVE LOC 6
929  003474  013746  000004                         MOV     @#4,-(SP)       ;SAVE LOC 4
930  003500  005000                                 CLR     R0              ;START AT 0
931  003502  012737  003546  000004                 MOV     #2$,@#4         ;SET UP FOR TIME OUT
932  003510  005037  000006                         CLR     @#6             ;TO AUTOSIZE MEMORY
933  003514  005720                         6$:     TST     (R0)+           ;CHECK ADDRESS IN R0
934  003516  022700  157776                         CMP     #157776,R0      ;IS IT AT LEAST 28K
935  003522  001374                                 BNE     6$              ;BR IF NO
936  003524  162700  007776                         SUB     #7776,R0        ;SAVE 2K FOR MONITORS
937  003530  010037  001466                 7$:     MOV     R0,MEMLIM       ;STORE MEMORY LIMIT
938  003534  012637  000004                         MOV     (SP)+,@#4       ;RESTORE LOC 4
939  003540  012637  000006                         MOV     (SP)+,@#6       ;RESTORE LOC 6
940  003544  000413                                 BR      10$             ;CONTINUE
941  003546  022626                         2$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
942  003550  162700  000004                         SUB     #4,R0           ;GET LAST GOOD ADDRESS
943  003554  162700  007776                         SUB     #7776,R0        ;SAVE 2K FOR MONITORS
944  003560  022700  030000                         CMP     #30000,R0       ;IS IT 8K?
945  003564  001361                                 BNE     7$              ;BR IF NO
946  003566  012700  037400                         MOV     #37400,R0       ;IF 8K DON'T SAVE 2K
947  003572  000756                                 BR      7$              ;
948  003574  012737  000340  177776  10$:     MOV     #340,PS         ;LOCK OUT INTERRUPTS
949  003602  032737  000004  001446                 BIT     #BIT2,STRTSW    ;CHECK FOR LOCK ON TEST
950  003610  001406                                 BEQ     1$              ;BR IF NO LOCK DESIRED.
951  003612  104401  007716                         TYPE    ,MLOCK          ;TYPE LOCK SELECTED.
952  003616  012737  000240  004146                 MOV     #NOP,TTST       ;SET UP TO LOCK
953  003624  000403                                 BR      3$              ;CONTINUE ALONG.
954  003626  013737  004360  004146  1$:     MOV     BRW,TTST        ;PREPARE NORMAL SCOPE ROUTINE
```

```
  955  003634  012737  011460  001206  3$:     MOV     #CYCLE,$LPADR   ;START AT "CYCLE" FIND WHICH DEVICE TO TEST
  956  003642  032737  000002  001446  4$:     BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED?
  957  003650  001002                          BNE     5$              ;BR IF YES
  958  003652  104401  007642                  TYPE    ,MR             ;TYPE R
  959  003656  000177  175324          5$:     JMP     @$LPADR         ;START TESTING
```

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

                                .SBTTL  END OF PASS ROUTINE

                                ;;**********************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO CYCLE

003662                          $EOP:
                                        RESET
003662  000005
003664  005237  001324                  INC     $PASS           ; INCREMENT THE PASS COUNT
003670  105037  001203                  CLRB    $ERFLG          ;CLEAR ERROR FLAG
003674  104401  007620                  TYPE    ,MEPASS         ;TYPE END PASS.
003700  104401  007745                  TYPE    ,MCSRX          ;TYPE "CSR"
003704  104417  004104                  CNVRT   ,XCSR           ;SHOW IT.
003710  104401  007753                  TYPE    ,MVECX          ;TYPE VECTOR.
003714  104417  004112                  CNVRT   ,XVEC           ;SHOW IT.
003720  104401  007761                  TYPE    ,MPASSX         ;TYPE " PASSES "
003724  104417  004120                  CNVRT   ,XPASS          ;SHOW IT.
003730  104401  007772                  TYPE    ,MERRX          ;TYPE " ERRORS "
003734  104417  004126                  CNVRT   ,XERR           ;SHOW IT.
003740  013700  001504                  MOV     #ILK,R0         ;SET POINTER TO PASSCNT.
003744  013720  001324                  MOV     $PASS,(R0)+     ;SAVE THE PASS COUNT.
003750  013720  001212                  MOV     $ERTTL,(R0)+    ;SAVE ERROR COUNT
003754  013777  002060  176074          MOV     KMRLVL,@KMRVEC  ;RESTORE THE RECEIVER INTERRUPT VECTOR.
003762  005077  176072                  CLR     @KMRLVL         ;RESTORE RECEIVER LEVEL
003766  013777  002064  176066          MOV     KMTLVL,@KMTVEC  ;RESTORE THE TRANSMIT INTERRUPT VECTOR.
003774  005077  176064                  CLR     @KMTLVL         ;RESTORE TRANSMITTER LEVEL
004000  005337  001476                  DEC     SAVNUM          ;ALL DEVICE TESTED?
004004  001035                          BNE     $DOAGN          ;BRANCH IF NO.
004006  112737  000377  001511          MOVB    #377,QV.FLG     ;SET QUICK VERIFY FLAG.
004014  013737  001472  001476          MOV     KMNUM,SAVNUM    ;RESTORE DEVICE COUNT.
004022  005037  001216                  CLR     $ERRPC          ;CLEAR LAST ERROR PC
004026  005037  001310                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
004032  005237  001324                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
004036  042737  100000  001324          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
004044  005327                          DEC     (PC)+           ;;LOOP?
004046  000001                  $EOPCT: .WORD   1
004050  003013                          BGT     $DOAGN          ;;YES
004052  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
004054  000001                  $ENDCT: .WORD   1
004056  004046                          $EOPCT
004060  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
004064  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
004066  000005                          RESET                   ;;CLEAR THE WORLD
004070  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
004072  000240                          NOP                     ;;SAVE ROOM
004074  000240                          NOP                     ;;FOR
004076  000240                          NOP                     ;;ACT11
004100                          $DOAGN:
004100  000137                          JMP     @(PC)+          ;;RETURN
```

```
                        END OF PASS ROUTINE

1016  004102  011460                $RTNAD: .WORD   CYCLE
1017  004104  000001                XCSR:   1
1018  004106     006      002                .BYTE   6,2
1019  004110  002066                        KMCSR
1020  004112  000001                XVEC:   1
1021  004114     004      002                .BYTE   4,2
1022  004116  002056                        KMRVEC
1023  004120  000001                XPASS:  1
1024  004122     006      002                .BYTE   6,2
1025  004124  001324                        $PASS
1026  004126  000001                XERR:   1
1027  004130     006      002                .BYTE   6,2
1028  004132  001212                        $ERTTL
1029
1030                                ;SCOPE LOOP AND INTERATION HANDLER
1031                                ;--------------------------------
1032
1033                                .SBTTL  SCOPE HANDLER ROUTINE
1034
1035                    ;;**********************************************************************
1036                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
1037                    ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
1038                    ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
1039                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1040                    ;*SW14=1         LOOP ON TEST
1041                    ;*SW11=1         INHIBIT ITERATIONS
1042                    ;*CALL
1043                    ;*      SCOPE           ;;SCOPE=IOT
1044
1045  004134            $SCOPE:
1046  004134  005037  001216            CLR     $ERRPC                  ; CLEAR LAST ERROR PC
1047  004140  023716  013734            CMP     TST1+2,(SP)             ; IS THIS TEST #1 ?
1048  004144  001413                    BEQ     $XTSTR                  ; IF SO DON'T LOOP.
1049  004146  000406            TTST:   BR      1$                      ;
1050  004150  105777  175070            TSTB    @$TKS                   ; KEYBOARD DONE ?
1051  004154  100067                    BPL     $OVER                   ; IF NO DONT WAIT.
1052  004156  017766  175064  177776    MOV     @$TKB,-2(SP)            ;
1053  004164  032777  040000  175046  1$:BIT     #BIT14,@$WR     ;;LOOP ON PRESENT TEST?
1054  004172  001060                    BNE     $OVER           ;;YES IF SW14=1
1055                                ;#####START OF CODE FOR THE XOR TESTER#####
1056  004174  000416            $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
1057                                                            ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
1058  004176  013746  000004            MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
1059  004202  012737  004222  000004    MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
1060  004210  005737  177060            TST     @#177060        ;;TIME OUT ON XOR?
1061  004214  012637  000004            MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
1062  004220  000436                    BR      $SVLAD          ;;GO TO THE NEXT TEST
1063  004222  022626            5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
1064  004224  012637  000004            MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
1065  004230  000441                    BR      $OVER           ;;LOOP ON THE PRESENT TEST
1066  004232                    6$:;#####END OF CODE FOR THE XOR TESTER#####
1067  004232  105737  001203    2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
1068  004236  001404                    BEQ     3$              ;;BR IF NO
1069  004240  105037  001203    4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
1070  004244  005037  001310            CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
1071  004250  032777  004000  174762  3$:BIT     #BIT11,@$WR     ;;INHIBIT ITERATIONS?
```

```
1072  004256  001011                        BNE     1$              ;;BR IF YES
1073  004260  005737  001324                TST     $PASS           ;;IF FIRST PASS OF PROGRAM
1074  004264  001406                        BEQ     1$              ;;       INHIBIT ITERATIONS
1075  004266  005237  001204                INC     $ICNT           ;;INCREMENT ITERATION COUNT
1076  004272  023737  001310  001204        CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
1077  004300  002015                        BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
1078  004302  012737  000001  001204  1$:    MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
1079  004310  013737  004362  001310        MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
1080  004316  105237  001202        $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
1081  004322  113737  001202  001322        MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
1082  004330  011637  001206                MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
1083  004334  013777  001202  174700  $OVER: MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
1084  004342  013716  001206                MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
1085  004346  005037  001444                CLR     LOCK                    ; RESET LOCK ON DATA.
1086  004352  013701  002066                MOV     KMCSR,R1                ; R1 CONTAINS BASE KMC ADDRESS.
1087  004356  000002                        RTI
1088  004360  000406                BRW:    .WORD   406
1089  004362  000020                $MXCNT: 20                      ;;MAX. NUMBER OF ITERATIONS
1090
1091                                        ;CHECK FOR FREEZE ON CURRENT DATA
1092                                        ;-------------------------------
1093
1094  004364  004737  011212        .SCOP1: JSR     PC,CKSWR                ;CHECK FOR SOFT SWR
1095  004370  032777  001000  174642        BIT     #SW09,@SWR      ;IS SW09=1(SET)?
1096  004376  001405                        BEQ     1$              ;BR IF NOT SET.
1097  004400  005737  001444                TST     LOCK
1098  004404  001402                        BEQ     1$
1099  004406  013716  001444                MOV     LOCK,(SP)       ;GOTO THE ADDRESS IN LOCK.
1100  004412  000002                1$:     RTI                     ;GO BACK.
1101
1102                                        ;TELETYPE OUTPUT ROUTINE
1103                                        ;----------------------
1104
1105                                .SBTTL  TYPE ROUTINE
1106
1107                                ;;*****************************************************************
1108                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1109                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1110                                ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1111                                ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1112                                ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1113                                ;*
1114                                ;*CALL:
1115                                ;*1) USING A TRAP INSTRUCTION
1116                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
1117                                ;*OR
1118                                ;*      TYPE
1119                                ;*      MESADR
1120                                ;*
1121
1122  004414  105737  001257        $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
1123  004420  100002                        BPL     1$              ;;BR IF YES
1124  004422  000000                        HALT                    ;;HALT HERE IF NO TERMINAL
1125  004424  000430                        BR      3$              ;;LEAVE
1126  004426  010046                1$:     MOV     R0,-(SP)        ;;SAVE R0
1127  004430  017600  000002                MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
```

```
1128  004434  122737  000001  001336          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
1129  004442  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
1130  004444  132737  000100  001337          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
1131  004452  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
1132  004454  010037  004464                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
1133  004460  004737  004704                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
1134  004464  000000                61$:     .WORD   0               ;;MESSAGE ADDRESS
1135  004466  132737  000040  001337  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
1136  004474  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
1137  004476  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1138  004500  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
1139  004502  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
1140  004504  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
1141  004506  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
1142  004512  000002                          RTI                     ;;RETURN
1143  004514  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
1144  004520  001430                          BEQ     8$
1145  004522  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
1146  004526  001006                          BNE     5$
1147  004530  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
1148  004532  104401                          TYPE                    ;;TYPE A CR AND LF
1149  004534  001313                          $CRLF
1150  004536  105037  004672                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
1151  004542  000755                          BR      2$              ;;GET NEXT CHARACTER
1152  004544  004737  004626          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
1153  004550  123726  001256          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
1154  004554  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
1155  004556  013746  001254                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
1156                                                                  ;;AND THE NULL CHAR.
1157  004562  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
1158  004566  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
1159  004570  004737  004626                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
1160  004574  105337  004672                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
1161  004600  000770                          BR      7$              ;;LOOP
1162
1163                                  ;HORIZONTAL TAB PROCESSOR
1164
1165  004602  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
1166  004606  004737  004626          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
1167  004612  132737  000007  004672          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
1168  004620  001372                          BNE     9$              ;;TAB STOP
1169  004622  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
1170  004624  000724                          BR      2$              ;;GET NEXT CHARACTER
1171  004626  105777  174416          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
1172  004632  100375                          BPL     $TYPEC
1173  004634  116677  000002  174410          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
1174  004642  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
1175  004650  001003                          BNE     1$              ;;BRANCH IF NO
1176  004652  105037  004672                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
1177  004656  000406                          BR      $TYPEX          ;;EXIT
1178  004660  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
1179  004666  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
1180  004670  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
1181  004672  000000          $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
1182  004674  000207          $TYPEX: RTS     PC
1183
```

```
                                .SBTTL  APT COMMUNICATIONS ROUTINE
                                ;;*****************************************************************
004676  112737  000001  005142  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
004704  112737  000001  005140  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
004712  000403                  BR      $ATYC
004714  112737  000001  005142  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
004722                          $ATYC:
004722  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
004724  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
004726  105737  005140          TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
004732  001450                  BEQ     5$              ;;IF NOT:  BR
004734  122737  000001  001336  CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
004742  001031                  BNE     3$              ;;IF NOT:  BR
004744  132737  000100  001337  BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
004752  001425                  BEQ     3$              ;;IF NOT:  BR
004754  017600  000004          MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
004760  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
004766  005737  001316          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
004772  001375                  BNE     1$              ;;IF NOT:  WAIT
004774  010037  001332          MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
005000  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
005002  001376                  BNE     2$
005004  163700  001332          SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
005010  006200                  ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
005012  010037  001334          MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
005016  012737  000004  001316  MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
005024  000413                  BR      5$
005026  017637  000004  005052  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
005034  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
005042  013746  177776          MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
005046  004737  004414          JSR     PC,$TYPE        ;;CALL TYPE MACRO
005052  000000                  4$:     .WORD   0
005054                          5$:
005054  105737  005142          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
005060  001416                  BEQ     12$             ;;IF NOT:  BR
005062  005737  001336          TST     $ENV            ;;RUNNING UNDER APT?
005066  001413                  BEQ     12$             ;;IF NOT:  BR
005070  005737  001316          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
005074  001375                  BNE     11$             ;;IF NOT:  WAIT
005076  017637  000004  001320  MOV     @4(SP),$FATAL   ;;GET ERROR #
005104  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
005112  005237  001316          INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
005116  105037  005142          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
005122  105037  005141          CLRB    $LFLG           ;;CLEAR LOG FLAG
005126  105037  005140          CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
005132  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
005134  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
005136  000207                  RTS     PC              ;;RETURN
005140     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
005141     000          $LFLG:  .BYTE   0               ;;LOG FLAG
005142     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
005144                          .EVEN
                        APTSIZE=200
                        APTENV=001
                        APTSPOOL=100
```

```
1240          000040                  APTCSUP=040
1241                                          ;-------------------------
1242
1243                                  .SBTTL  TTY INPUT ROUTINE
1244
1245                                  ;;**********************************************************************
1246                                  .ENABL  LSB
1247
1248                                  .DSABL  LSB
1249
1250
1251                                  ;;**********************************************************************
1252                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
1253                                  ;*CALL:
1254                                  ;*      RDCHR                       ;;INPUT A SINGLE CHARACTER FROM THE TTY
1255                                  ;*      RETURN HERE                 ;;CHARACTER IS ON THE STACK
1256                                  ;*                                  ;;WITH PARITY BIT STRIPPED OFF
1257                                  ;
1258
1259  005144  011646                  $RDCHR: MOV     (SP),-(SP)          ;;PUSH DOWN THE PC
1260  005146  016666  000004  000002          MOV     4(SP),2(SP)         ;;SAVE THE PS
1261  005154  105777  174064          1$:     TSTB    @$TKS               ;;WAIT FOR
1262  005160  100375                          BPL     1$                  ;;A CHARACTER
1263  005162  117766  174060  000004          MOVB    @$TKB,4(SP)         ;;READ THE TTY
1264  005170  042766  177600  000004          BIC     #^C<177>,4(SP)      ;;GET RID OF JUNK IF ANY
1265  005176  026627  000004  000023          CMP     4(SP),#23           ;;IS IT A CONTROL-S?
1266  005204  001013                          BNE     3$                  ;;BRANCH IF NO
1267  005206  105777  174032          2$:     TSTB    @$TKS               ;;WAIT FOR A CHARACTER
1268  005212  100375                          BPL     2$                  ;;LOOP UNTIL ITS THERE
1269  005214  117746  174026                  MOVB    @$TKB,-(SP)         ;;GET CHARACTER
1270  005220  042716  177600                  BIC     #^C177,(SP)         ;;MAKE IT 7-BIT ASCII
1271  005224  022627  000021                  CMP     (SP)+,#21           ;;IS IT A CONTROL-Q?
1272  005230  001366                          BNE     2$                  ;;IF NOT DISCARD IT
1273  005232  000750                          BR      1$                  ;;YES, RESUME
1274  005234  026627  000004  000140  3$:     CMP     4(SP),#140          ;;IS IT UPPER CASE?
1275  005242  002407                          BLT     4$                  ;;BRANCH IF YES
1276  005244  026627  000004  000175          CMP     4(SP),#175          ;;IS IT A SPECIAL CHAR?
1277  005252  003003                          BGT     4$                  ;;BRANCH IF YES
1278  005254  042766  000040  000004          BIC     #40,4(SP)           ;;MAKE IT UPPER CASE
1279  005262  000002                  4$:     RTI                         ;;GO BACK TO USER
1280                                  ;;**********************************************************************
1281                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
1282                                  ;*CALL:
1283                                  ;*      RDLIN                       ;;INPUT A STRING FROM THE TTY
1284                                  ;*      RETURN HERE                 ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
1285                                  ;*                                  ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
1286
1287  005264  010346                  $RDLIN: MOV     R3,-(SP)            ;;SAVE R3
1288  005266  005046                          CLR     -(SP)               ;;CLEAR THE RUBOUT KEY
1289  005270  012703  005520          1$:     MOV     #$TTYIN,R3          ;;GET ADDRESS
1290  005274  022703  005527          2$:     CMP     #$TTYIN+7,R3        ;;BUFFER FULL?
1291  005300  101456                          BLOS    4$                  ;;BR IF YES
1292  005302  104402                          RDCHR                       ;;GO READ ONE CHARACTER FROM THE TTY
1293  005304  112613                          MOVB    (SP)+,(R3)          ;;GET CHARACTER
1294  005306  122713  000177          10$:    CMPB    #177,(R3)           ;;IS IT A RUBOUT
1295  005312  001022                          BNE     5$                  ;;BR IF NO
```

```
1296  005314  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
1297  005316  001007                          BNE     6$              ;;BR IF NO
1298  005320  112737  000134  005516          MOVB    #'\,9$          ;;TYPE A BACK SLASH
1299  005326  104401  005516                  TYPE    9$
1300  005332  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
1301  005336  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
1302  005340  020327  005520                  CMP     R3,#STTYIN      ;;STACK EMPTY?
1303  005344  103434                          BLO     4$              ;;BR IF YES
1304  005346  111337  005516                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
1305  005352  104401  005516                  TYPE    9$              ;;GO TYPE
1306  005356  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
1307  005360  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
1308  005362  001406                          BEQ     7$              ;;BR IF NO
1309  005364  112737  000134  005516          MOVB    #'\,9$          ;;TYPE A BACK SLASH
1310  005372  104401  005516                  TYPE    9$
1311  005376  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
1312  005400  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
1313  005404  001003                          BNE     8$              ;;BR IF NO
1314  005406  104401  005527                  TYPE    $CNTLU          ;;TYPE A CONTROL "U"
1315  005412  000726                          BR      1$              ;;GO START OVER
1316  005414  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
1317  005420  001011                          BNE     3$              ;;BRANCH IF NO
1318  005422  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
1319  005424  104401  001313                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
1320  005430  104401  005520                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
1321  005434  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
1322  005436  104401  001312          4$:     TYPE    $QUES           ;;TYPE A '?'
1323  005442  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
1324  005444  111337  005516          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
1325  005450  104401  005516                  TYPE    9$
1326  005454  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
1327  005460  001305                          BNE     2$              ;;LOOP IF NOT RETURN
1328  005462  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
1329  005466  104401  001314                  TYPE    $LF             ;;TYPE A LINE FEED
1330  005472  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
1331  005474  012603                          MOV     (SP)+,R3        ;;RESTORE R3
1332  005476  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
1333  005500  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
1334  005506  012766  005520  000004          MOV     #STTYIN,4(SP)
1335  005514  000002                          RTI                     ;;RETURN
1336  005516     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
1337  005517     000                          .BYTE   0               ;;TERMINATOR
1338  005520  000007          $TTYIN: .BLKB   7               ;;RESERVE 7 BYTES FOR TTY INPUT
1339  005527     136  006525  000012  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
1340  005534  043536  005015     000  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
1341  005541     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
1342  005546  036440  000040
1343  005552  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
1344  005560  020075     000
1345          005564                  .EVEN
1346                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
1347
1348                                  ;;*****************************************************************
1349                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
1350                                  ;*CHANGE IT TO BINARY.
1351                                  ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
```

```
1352                                    ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
1353                                    ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
1354                                    ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
1355                                    ;*CALL:
1356                                    ;*      RDOCT                   ;;READ AN OCTAL NUMBER
1357                                    ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
1358                                    ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
1359
1360  005564  011646                    $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
1361  005566  016666  000004  000002            MOV     4(SP),2(SP)     ;;INPUT NUMBER
1362  005574  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1363  005576  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1364  005600  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
1365  005602  104403                    1$:     RDLIN                   ;;READ AN ASCIZ LINE
1366  005604  012600                            MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
1367  005606  010037  005712                    MOV     R0,5$           ;;AND SAVE IT
1368  005612  005001                            CLR     R1              ;;CLEAR DATA WORD
1369  005614  005002                            CLR     R2
1370  005616  112046                    2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
1371  005620  001420                            BEQ     3$              ;;IF ZERO GET OUT
1372  005622  122716  000060                    CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
1373  005626  003026                            BGT     4$              ;;IS AN OCTAL DIGIT
1374  005630  122716  000067                    CMPB    #'7,(SP)
1375  005634  002423                            BLT     4$
1376  005636  006301                            ASL     R1              ;;*2
1377  005640  006102                            ROL     R2
1378  005642  006301                            ASL     R1              ;;*4
1379  005644  006102                            ROL     R2
1380  005646  006301                            ASL     R1              ;;*8
1381  005650  006102                            ROL     R2
1382  005652  042716  177770                    BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
1383  005656  062601                            ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
1384  005660  000756                            BR      2$              ;;LOOP
1385  005662  005726                    3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
1386  005664  010166  000012                    MOV     R1,12(SP)       ;;SAVE THE RESULT
1387  005670  010237  005722                    MOV     R2,$HIOCT
1388  005674  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
1389  005676  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
1390  005700  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
1391  005702  000002                            RTI                     ;;RETURN
1392  005704  005726                    4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
1393  005706  105010                            CLRB    (R0)            ;;SET A TERMINATOR
1394  005710  104401                            TYPE                    ;;TYPE UP THRU THE BAD CHAR.
1395  005712  000000X                   5$:     .WORD   0
1396  005714  104401  001312                    TYPE    ,$QUES          ;;"?" "CR" & "LF"
1397  005720  000730                            BR      1$              ;;TRY AGAIN
1398  005722  000000                    $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
1399                                  ;
1400                                  ;       INPUT OCTAL NUMBER ROUTINE
1401                                  ;-----------------------------------
1402                                  ;
1403  005724  010546                    $INPUT: MOV     R5,-(SP)                ; SAVE REGISTER R5.
1404  005726  016605  000002                    MOV     2(SP),R5                ; GET FIRST PARAMETER ADDRESS.
1405  005732  012537  005770                    MOV     (R5)+,WHAT              ; GET MESSAGE ADDRESS.
1406  005736  012537  006050                    MOV     (R5)+,LOLIM             ; GET LOW LIMIT FOR THE  #
1407  005742  012537  006052                    MOV     (R5)+,HILIM             ; GET HIGH LIMIT FOR THE #.
```

```
1408  005746  012537  006054          MOV     (R5)+,WHERE     ;GET ADDRESS OF INBUFFER
1409  005752  112537  006056          MOVB    (R5)+,LOBITS    ;GET LOWMASK BITS,
1410  005756  112537  006057          MOVB    (R5)+,ADRCNT    ;GET # OF #'S TO BE GENERATED.
1411  005762  010566  000002          MOV     R5,2(SP)        ;SAVE THE RETURN ADDRESS.
1412  005766  104401            INLP1: TYPE                    ;TYPE THE MESSAGE.
1413  005770  000000            WHAT:  .WORD   0
1414  005772  104404                   RDOCT                   ;READ OCTAL # FROM KEYBOARD.
1415  005774  021637  006052          CMP     (SP),HILIM      ;IS IT IN HIGH LIMIT?
1416  006000  003003                   BGT     2$              ;BRANCH IF NO.
1417  006002  021637  006050          CMP     (SP),LOLIM      ;IS IT MORE THAN LOW LIMIT.
1418  006006  002005                   BGE     3$              ;BRANCH IF YES.
1419  006010  104401  001312    2$:    TYPE    ,$QUES          ;TYPE "?"
1420  006014  104401  001313          TYPE    ,$CRLF          ;TYPE <CR>,<LF>
1421  006020  000762                   BR      INLP1
1422  006022  013705  006054    3$:    MOV     WHERE,R5        ;GET BUFFER ADDRESS.
1423  006026  011625    4$:    MOV     (SP),(R5)+      ;SAVE THE # IN RIGHT PLACE.
1424  006030  062716  000002          ADD     #2,(SP)         ;NEXT SEQUENTIAL NUMBER.
1425  006034  105337  006057          DECB    ADRCNT          ;COUNT BY 1.
1426  006040  001372                   BNE     4$              ;BRANCH IF NOT DONE.
1427  006042  005726                   TST     (SP)+           ;POP THE STACK POINTER.
1428  006044  012605                   MOV     (SP)+,R5        ;POP THE REG.5
1429  006046  000002                   RTI
1430  006050  000000            LOLIM: .WORD   0
1431  006052  000000            HILIM: .WORD   0
1432  006054  000000            WHERE: .WORD   0
1433  006056     000             LOBITS: .BYTE   0
1434  006057     000             ADRCNT: .BYTE   0
1435
1436                                    ;  ADVANCE TO NEXT TEST HANDLER
1437                                    ;---------------------------------
1438                                    ;
1439  006060  013716  001442    .ADVANCE:       MOV     NEXT,(SP)               ; CRUNCH STACK WITH ADDRESSOF SCOPE CALL
1440  006064  005037  001444            CLR     LOCK                    ; RESET TIGHT LOOP ADDRESS
1441  006070  000002                    RTI                             ; CHECK TO SEE IF OLD TEST GETS REPEATED
1442
1443                                    ;SAVE PC OF TEST THAT FAILED AND R0-R5
1444                                    ;-------------------------------------
1445
1446  006072  016637  000004  001460  .SAV05: MOV     4(SP),SAVPC     ;SAVE R7 (PC)
1447
1448                                    ;SAVE R0-R5
1449
1450  006100  010537  001274    SV05:   MOV     R5,$REG5        ;SAVE R5
1451  006104  010437  001272            MOV     R4,$REG4        ;SAVE R4
1452  006110  010337  001270            MOV     R3,$REG3        ;SAVE R3
1453  006114  010237  001266            MOV     R2,$REG2        ;SAVE R2
1454  006120  010137  001264            MOV     R1,$REG1        ;SAVE R1
1455  006124  010037  001262            MOV     R0,$REG0        ;SAVE R0
1456  006130  000002                    RTI                     ;LEAVE.
1457
1458                                    ;RESTORE R0-R5
1459
1460  006132  013700  001262    .RES05: MOV     $REG0,R0        ;RESTORE R0
1461  006136  013701  001264            MOV     $REG1,R1        ;RESTORE R1
1462  006142  013702  001266            MOV     $REG2,R2        ;RESTORE R2
1463  006146  013703  001270            MOV     $REG3,R3        ;RESTORE R3
```

```
1464  006152  013704  001272                  MOV     $REG4,R4        ;RESTORE R4
1465  006156  013705  001274                  MOV     $REG5,R5        ;RESTORE R5
1466  006162  000002                           RTI                     ;LEAVE
1467
1468                                  ;       ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
1469                                  ;       ;-----------------------------------------------------
1470                                  ;
1471  006164  104401  001313          .CONVR: TYPE    ,$CRLF
1472  006170  010046                  .CNVRT: MOV     R0,-(SP)
1473  006172  010146                          MOV     R1,-(SP)
1474  006174  010346                          MOV     R3,-(SP)
1475  006176  010446                          MOV     R4,-(SP)
1476  006200  010546                          MOV     R5,-(SP)
1477  006202  017601  000012                  MOV     @12(SP),R1
1478  006206  062766  000002  000012          ADD     #2,12(SP)
1479  006214  012137  006406                  MOV     (R1)+,WRDCNT
1480  006220  112137  006410          1$:     MOVB    (R1)+,CHRCNT
1481  006224  112137  006411                  MOVB    (R1)+,SPACNT
1482  006230  013137  006412                  MOV     @(R1)+,BINWRD
1483  006234  122737  000003  006410          CMPB    #3,CHRCNT
1484  006242  001003                          BNE     2$
1485  006244  042737  177400  006412          BIC     #177400,BINWRD
1486  006252  013704  006412          2$:     MOV     BINWRD,R4
1487  006256  113705  006410                  MOVB    CHRCNT,R5
1488  006262  012700  011106                  MOV     #TEMP,R0
1489  006266  010403                  3$:     MOV     R4,R3
1490  006270  042703  177770                  BIC     #177770,R3
1491  006274  062703  000060                  ADD     #060,R3
1492  006300  110320                          MOVB    R3,(R0)+
1493  006302  000241                          CLC
1494  006304  006004                          ROR     R4
1495  006306  000241                          CLC
1496  006310  006004                          ROR     R4
1497  006312  000241                          CLC
1498  006314  006004                          ROR     R4
1499  006316  005305                          DEC     R5
1500  006320  001362                          BNE     3$
1501  006322  012703  011150                  MOV     #MDATA,R3
1502  006326  114023          4$:     MOVB    -(R0),(R3)+
1503  006330  105337  006410                  DECB    CHRCNT
1504  006334  001374                          BNE     4$
1505  006336  105737  006411                  TSTB    SPACNT
1506  006342  001405                          BEQ     6$
1507  006344  112723  000040          5$:     MOVB    #040,(R3)+
1508  006350  105337  006411                  DECB    SPACNT
1509  006354  001373                          BNE     5$
1510  006356  105013                  6$:     CLRB    (R3)
1511  006360  104401  011150                  TYPE    ,MDATA
1512  006364  005337  006406                  DEC     WRDCNT
1513  006370  001313                          BNE     1$
1514  006372  012605                          MOV     (SP)+,R5
1515  006374  012604                          MOV     (SP)+,R4
1516  006376  012603                          MOV     (SP)+,R3
1517  006400  012601                          MOV     (SP)+,R1
1518  006402  012600                          MOV     (SP)+,R0
1519  006404  000002                          RTI
```

```
1520  006406  000000                  WRDCNT: 0
1521  006410  000000                  CHRCNT: 0
1522          006411                  SPACNT=CHRCNT+1
1523  006412  000000                  BINWRD: 0
1524
1525
1526                                          ;TRAP DISPATCH SERVICE
1527                                          ;ARGUMENT OF TRAP IS EXTRACTED
1528                                          ;AND USED AS OFFSET TO OBTAIN POINTER
1529                                          ;TO SELECTED SUBROUTINE
1530
1531                                  .SBTTL  TRAP DECODER
1532
1533                                  ;;**************************************************************
1534                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
1535                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
1536                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
1537                                  ;*GO TO THAT ROUTINE.
1538
1539  006414  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
1540  006416  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
1541  006422  005740                          TST     -(R0)           ;;BACKUP BY 2
1542  006424  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
1543  006426  006300                          ASL     R0              ;;POSITION FOR INDEXING
1544  006430  016000  006450                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
1545  006434  000200                          RTS     R0              ;;GO TO ROUTINE
1546
1547
1548                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
1549
1550  006436  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
1551  006440  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
1552  006446  000002                          RTI                     ;;RESTORE THE PSW
1553
1554                                  .SBTTL  TRAP TABLE
1555
1556                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
1557                                  ;*BY THE "TRAP" INSTRUCTION.
1558
1559                                  ;       ROUTINE
1560                                  ;       -------
1561  006450  006436                  $TRPAD: .WORD   $TRAP2
1562  006452  004414                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
1563
1564
1565  006454  005144                          $RDCHR  ;;CALL=RDCHR    TRAP+2(104402)  TTY TYPEIN CHARACTER ROUTINE
1566  006456  005264                          $RDLIN  ;;CALL=RDLIN    TRAP+3(104403)  TTY TYPEIN STRING ROUTINE
1567  006460  005564                          $RDOCT  ;;CALL=RDOCT    TRAP+4(104404)  READ AN OCTAL NUMBER FROM TTY
1568  006462  004364                          .SCOP1  ;;CALL=SCOP1    TRAP+5(104405)  CALL TO LOOP ON CURRENT DATA HANDLER
1569  006464  006072                          .SAV05  ;;CALL=SAV05    TRAP+6(104406)  CALL TO REGISTER SAVE ROUTINE
1570  006466  006132                          .RES05  ;;CALL=RES05    TRAP+7(104407)  CALL TO REGISTER RESTORE ROUTINE
1571  006470  007362                          .MSTCLR ;;CALL=MSTCLR   TRAP+10(104410) CALL TO ISSUE A MASTER CLEAR
1572  006472  007332                          .DELAY  ;;CALL=DELAY    TRAP+11(104411) CALL TO DELAY
1573  006474  007400                          .ROMCLK ;;CALL=ROMCLK   TRAP+12(104412) CALL TO CLOCK ROM ONCE
1574  006476  007446                          .DATACLK        ;;CALL=DATACLK  TRAP+13(104413) CALL TO CLOCK DATA
1575  006500  007512                          .TIMER  ;;CALL=TIMER    TRAP+14(104414) CALL TO DELAY A CLOCK TICK
```

```
                                        $INPUT  ;;CALL=INPUT    TRAP+15(104415) CALL TO OCTAL # INPUT ROUTINE
                                        .CONVRT ;;CALL=CONVRT   TRAP+16(104416) CALL TO    .....
                                        .CNVRT  ;;CALL=CNVRT    TRAP+17(104417) CALL TO    .....
                                        .ADVANCE        ;;CALL=ADVANCE  TRAP+20(104420) CALL TO ADVANCE TO NEXT TEST
                                ;
                                ;-------------------------------------------------------------------
                                ;:*****************************************************************
                                        ;ERROR HANDLER
                                        ;------------
006512  004737  011212          $ERROR: JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
006516  032777  010000  172514          BIT     #SW12,@SWR      ;BELL ON ERROR?
006524  001406                        BEQ     XBX             ;BR IF NO BELL
006526  105777  172516                TSTB    @STPS           ;TTY READY.
006532  100003                        BPL     XBX             ;DON'T WAIT IF TTY NOT READY.
006534  112777  000207  172510        MOVB    #207,@STPB      ;PUSH A BELL AT THE TTY.
006542  032777  020000  172470  XBX:  BIT     #SW13,@SWR      ;DELETE ERROR PRINT OUT?
006550  001107                        BNE     HALTS           ;BR IF NO PRINT OUT WANTED.
006552  021637  001216                CMP     (SP),$ERRPC     ;WAS THIS ERROR FOUND LAST TIME?
006556  001404                        BEQ     1$              ;BR IF YES
006560  011637  001216                MOV     (SP),$ERRPC     ;RECORD BEING HERE
006564  105037  001203                CLRB    $ERFLG          ;PREPARE HEADER
006570  104406                1$:     SAV05                   ;SAVE ALL PROC REGISTERS
006572  011605                        MOV     (SP),R5         ;GET THE PC OF ERROR
006574  162705  000002                SUB     #2,R5           ;GET ADDRESS OF TRAP CALL
006600  011504                        MOV     (R5),R4         ;GET ERROR INSTRUCTION
006602  110437  001214                MOVB    R4,$ITEMB       ; COPY ERROR # FOR APT HANDLING
006606  006304                        ASL     R4              ;MULT BY TWO
006610  061504                        ADD     (R5),R4         ;DOUBLE IT
006612  006304                        ASL     R4              ;MULT AGAIN
006614  042704  177001                BIC     #177001,R4      ;CLEAR JUNK
006620  062704  001512                ADD     #$ERRTB,R4      ;GET POINTER
006624  012437  006740                MOV     (R4)+,ERRMSG    ;GET ERROR MESSAGE
006630  012437  006752                MOV     (R4)+,DATAHD    ;GET DATA HEADRER
006634  011437  006764                MOV     (R4),DATABP     ;GET DATA TABLE
006640  105737  001203                TSTB    $ERFLG          ;TYPE HEADREER
006644  001403                        BEQ     TYPMSG          ;BR IF YES
006646  005737  006764                TST     DATABP          ;DOES DATA TABLE EXIST?
006652  001040                        BNE     TYPDAT          ;BR IF YES.
006654  104401  001313        TYPMSG: TYPE    ,$CRLF
006660  104401  001313                TYPE    ,$CRLF
006664  005737  001444                TST     LOCK
006670  001402                        BEQ     1$
006672  104401  010015                TYPE    ,MASTEK
006676  104401  010003        1$:     TYPE    ,MTSTN
006702  104417  007120                CNVRT   ,XTSTN          ;SHOW IT
006706  104401  010072                TYPE    ,MERRPC         ;TYPE PC.
006712  104417  007112                CNVRT   ,ERTABO         ;SHOW IT
006716  104401  001313                TYPE    ,$CRLF          ;GIVE A CR/LF
006722  112737  177777  001203        MOVB    #-1,$ERFLG      ;NO MORE HEADER UNLESS NO DATA TABLE.
006730  005737  006740                TST     ERRMSG          ;IS THERE AN ERROR MESSAGE?
006734  001402                        BEQ     WRKO.FM         ;BR IF NO.
006736  104401                        TYPE                    ;TYPE
006740  000000          ERRMSG: 0                             ;    ERROR MESSAGE
006742                  WRKO.FM:                              ;
006742  005737  006752                TST     DATAHD          ;DATA HEADER?
```

```
1632  006746  001402                        BEQ     TYPDAT          ;BR IF NO
1633  006750  104401                        TYPE                    ;TYPE
1634  006752  000000                DATAHD: 0                       ;    DATA HEADER
1635  006754  005737  006764        TYPDAT: TST     DATABP          ;DATA TABLE?
1636  006760  001402                        BEQ     RESREG          ;BR IF NO.
1637  006762  104416                        CONVRT                  ;SHOW
1638  006764  000000                DATABP: 0                       ;    DATA TABLE
1639  006766  104407                RESREG: RESET                   ;RESTORE PROC REGISTERS
1640  006770  122737  000001  001336  HALTS:  CMPB    #APTENV,$ENV    ; IS APT RUNNING ?
1641  006776  001007                        BNE     3$              ; SKIP APT CALL IF NOT.
1642  007000  113737  001214  007012        MOVB    $ITEMB,6$       ; COPY ERROR #.
1643  007006  004737  004714                JSR     PC,$ATY4        ; CALL APT SERVICES.
1644  007012  000000                6$:     .WORD   0               ; ERROR # GOES HERE.
1645  007014  000777                9$:     BR      9$              ; LOCK HERE.
1646  007016  022737  004070  000042  3$:     CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, HALT!!
1647  007024  001403                        BEQ     1$
1648  007026  005777  172206                TST     @SWR            ;HALT ON ERROR?
1649  007032  100005                        BPL     EXITER          ;BR IF NO HALT ON ERROR
1650  007034  010046                1$:     PUSHRO                  ;SAVE R0
1651  007036  016600  000002                MOV     2(SP),R0        ;SHOW ERROR PC IN DATA LIGHTS
1652  007042  000000                        HALT                    ;HALT
1653  007044  012600                        POPRO                   ;GET R0
1654  007046  005237  001212        EXITER: INC     $ERTTL          ;UPDATE ERROR COUNT
1655  007052  032777  000400  172160        BIT     #SW08,@SWR      ;GOTO TOP OF TEST?
1656  007060  001007                        BNE     1$              ;BR IF YES
1657  007062  032777  002000  172150        BIT     #SW10,@SWR      ;GOTO NEXT TEST?
1658  007070  001407                        BEQ     2$              ;BR IF NO
1659  007072  013737  001442  001206        MOV     NEXT,$LPADR     ;SET FOR NEXT TEST
1660  007100  012706  001200        1$:     MOV     #STACK,SP       ;RESET SP
1661  007104  000177  172076                JMP     @$LPADR         ;GOTO SPECIFIED TEST
1662  007110  000002                2$:     RTI                     ;$LPADR
1663  007112  000001                ERTABO: 1
1664  007114     006     002                .BYTE   6,2
1665  007116  001460                        $AVPC
1666  007120  000001                XTSTN:  1
1667  007122     003     002                .BYTE   3,2
1668  007124  001202                        $TSTNM
1669                                        ;ENTER HERE ON POWER FAILURE
1670                                        ;--------------------------
1671
1672                                .SBTTL  POWER DOWN AND UP ROUTINES
1673
1674                                ;;*****************************************************************
1675                                ;POWER DOWN ROUTINE
1676  007126  012737  007316  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
1677  007134  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
1678  007142  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1679  007144  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1680  007146  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
1681  007150  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
1682  007152  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
1683  007154  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
1684  007156  017746  172056                MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
1685  007162  010637  007322                MOV     SP,$SAVR6       ;;SAVE SP
1686  007166  012737  007200  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
1687  007174  000000                        HALT
```

```
1688  007176  000776                    BR      .-2             ;;HANG UP
1689
1690                                    ;;*****************************************************************
1691                                    ;POWER UP ROUTINE
1692  007200  012737  007316  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
1693  007206  013706  007322            MOV     $SAVR6,SP       ;;GET SP
1694  007212  005037  007322            CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
1695  007216  005237  007322      1$:   INC     $SAVR6          ;;WAIT FOR THE INC
1696  007222  001375                    BNE     1$              ;;OF  WORD
1697  007224  104401  007562            TYPE    ,MPFAIL         ;
1698  007230  104417  007324            CNVRT   ,PFTAB          ;
1699  007234  105037  001203            CLRB    $ERFLG          ;CLEAR ERROR FLAG.
1700  007240  005037  001216            CLR     $ERRPC          ; CLEAR LAST ERROR PC
1701  007244  013701  002066            MOV     KMCSR,R1        ; RESTORE DEVICE ADDRESS.
1702  007250  005011                    CLR     (R1)            ; CLEAR THE CSR.
1703  007252  104410                    MSTCLR                  ;
1704  007254  012677  171760            MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
1705  007260  012605                    MOV     (SP)+,R5        ;;POP STACK INTO R5
1706  007262  012604                    MOV     (SP)+,R4        ;;POP STACK INTO R4
1707  007264  012603                    MOV     (SP)+,R3        ;;POP STACK INTO R3
1708  007266  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
1709  007270  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
1710  007272  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
1711  007274  012737  007126  000024    MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
1712  007302  012737  000340  000026    MOV     #340,@#PWRVEC+2 ;;PRIO:7
1713  007310  104401                    TYPE                    ;;REPORT THE POWER FAILURE
1714  007312  007562            $PWRMG: .WORD   MPFAIL          ;;POWER FAIL MESSAGE POINTER
1715  007314  000002                    RTI
1716  007316  000000            $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
1717  007320  000776                    BR      .-2             ;;  BEFORE THE POWER DOWN WAS COMPLETE
1718  007322  000000            $SAVR6: 0                       ;;PUT THE SP HERE
1719
1720  007324  000001            PFTAB:  1
1721  007326     003     002    .BYTE   3,2
1722  007330  001202                    $TSTNM
1723
1724  007332                    .DELAY:
1725  007332  012777  000020  172534    MOV     #20,@KMP04
1726  007340  104412                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1727  007342  121111                    121111                  ;POKE CLOCK DELAY BIT
1728  007344                      1$:
1729  007344  104412                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1730  007346  121224                    121224                  ;PORT4←IBUS*11
1731  007350  032777  000020  172516    BIT     #BIT4,@KMP04    ;IS CLOCK BIT SET?
1732  007356  001772                    BEQ     1$              ;BR IF NO
1733  007360  000002                    RTI
1734
1735  007362                    .MSTCLR:
1736  007362  152777  000100  172500    BISB    #BIT6,@KMCSRH   ;SET MASTER CLEAR
1737  007370  142777  000300  172472    BICB    #BIT6!BIT7,@KMCSRH  ;CLEAR MASTER CLEAR AND RUN
1738  007376  000002                    RTI                     ;RETURN
1739
1740  007400                    .ROMCLK:
1741  007400  152777  000002  172462    BISB    #BIT1,@KMCSRH   ;SET ROMI
1742  007406  013677  172464            MOV     @(SP)+,@KMP06   ;LOAD INSTRUCTION IN SEL6
1743  007412  062746  000002            ADD     #2,-(SP)        ;ADJUST STACK
```

```
1744  007416  032777  000100  171614          BIT     #SW06,@SWR      ;HALT IF SW06 =1
1745  007424  001401                          BEQ     1$              ;BR IF SW06 =0
1746  007426  000000                          HALT                    ;HALT BEFORE CLOCKING INSTRUCTION
1747  007430  152777  000003  172432  1$:     BISB    #BIT1!BIT0,@KMCSRH ;CLOCK IMSTRUCTION
1748  007436  142777  000007  172424          BICB    #BIT2!BIT1!BIT0,@KMCSRH  ;CLEAR ROM0, ROMI, STEP
1749  007444  000002                          RTI
1750
1751  007446                          .DATACLK:
1752  007446  013637  011106                  MOV     @(SP)+,TEMP     ;PUT TICK COUNT IN TEMP
1753  007452  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
1754  007456  152777  000020  172404  1$:     BISB    #BIT4,@KMCSRH   ;SET STEP LU
1755  007464  027777  172376  172374          CMP     @KMCSR,@KMCSR   ;WASTE TIME
1756  007472  142777  000020  172370          BICB    #BIT4,@KMCSRH   ;CLEAR STEP LU
1757  007500  005337  011106                  DEC     TEMP            ;DEC TICK COUNT
1758  007504  001364                          BNE     1$              ;BR IF NOT DONE
1759  007506  000002                          RTI                     ;RETURN
1760  007510  000001                  3$:     .BLKW 1
1761
1762  007512                          .TIMER:
1763  007512  013637  011106                  MOV     @(SP)+,TEMP     ;MOVE COUNT TO TEMP
1764  007516  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
1765  007522                          1$:
1766  007522  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1767  007524  021364                          021364                  ;PORT4=IBUS* REG11
1768  007526  032777  000002  172340          BIT     #2,@KMP04       ;IS PGM CLOCK BIT CLEAR?
1769  007534  001772                          BEQ     1$              ;BR IF YES
1770  007536                          2$:
1771  007536  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1772  007540  021364                          021364                  ;PORT4=IBUS* REG11
1773  007542  032777  000002  172324          BIT     #2,@KMP04       ;IS PGM CLOCK BIT SET?
1774  007550  001372                          BNE     2$              ;BR IF YES
1775  007552  005337  011106                  DEC     TEMP            ;DEC COUNT
1776  007556  001361                          BNE     1$              ;BR IF NOT DONE
1777  007560  000002                          RTI                     ;RETURN
1778
1779  007562  050200  051127  043040  MPFAIL: .ASCIZ   <200>/PWR FAILED. RESTART AT TEST /
 (2)  007620  042600  042116  050040  MEPASS: .ASCIZ   <200>/END PASS DZKCF  /
 (2)  007642  051200     000  MR:     .ASCIZ   <200>/R/
 (2)  007645     200  047516  042040  MERR2:  .ASCIZ   <200>/NO DEVICES PRESENT./
 (2)  007672  044600  051516  043125  MERR3:  .ASCIZ   <200>/INSUFFICIENT DATA!/
 (2)  007716  046200  041517  020113  MLOCK:  .ASCIZ   <200>/LOCK ON SELECTED TEST/
 (2)  007745     103  051123  020072  MCSRX:  .ASCIZ    /CSR: /
 (2)  007753     126  041505  020072  MVECX:  .ASCIZ   /VEC: /
 (2)  007761     120  051501  042523  MPASSX: .ASCIZ   /PASSES: /
 (2)  007772  051105  047522  051522  MERRX:  .ASCIZ   /ERRORS: /
 (2)  010003     124  051505  020124  MTSTN:  .ASCIZ   /TEST NO: /
 (2)  010015     052     000  MASTEK: .ASCIZ   /*/
 (2)  010017     200  042523  020124  MNEW:   .ASCIZ   <200>/SET SWITCH REG TO KMC11'S DESIRED ACTIVE./
 (2)  010072  041520  020072     000  MERRPC: .ASCIZ   /PC: /
 (2)  010077     200  020040  020040  XHEAD:  .ASCII   <200>/             MAP OF KMC11 STATUS/
 (2)  010136  020200  020040  020040          .ASCII   <200>/             ------------------/
 (2)  010175     200  020040  041520          .ASCII   <200>/  PC      CSR     STAT1    STAT2    STAT3/
 (2)  010247     200  026455  026455          .ASCIZ   <200>/------   ------   ------   ------   ------/
 (2)  010323     200  047510  020127  NUM:    .ASCIZ   <200>/HOW MANY KMC11'S TO BE TESTED?/
 (2)  010363     200  051503  020122  CSR:    .ASCIZ   <200>/CSR ADDRESS?/
 (2)  010401     200  042526  052103  VEC:    .ASCIZ   <200>/VECTOR ADDRESS?/
```

```
(2)  010422  041200  020122  051120  PRIO:   .ASCIZ  <200>/BR PRIORITY LEVEL? (4,5,6,7)?/
(2)  010461     200  044127  041511  MODU:   .ASCIZ  <200>/WHICH LINE UNIT? IF NONE TYPE "N", IF M8201 TYPE "1", IF M8202 TYP
(2)  010573     200  053523  052111  LINE:   .ASCIZ  <200>/SWITCH PAC#1 (DDCMP LINE #)?/
(2)  010631     200  053523  052111  BM:     .ASCIZ  <200>/SWITCH PAC#2 (BM873 BOOT ADD)?/
(2)  010671     200  051511  052040  CONN:   .ASCIZ  <200>/IS THE LOOP BACK CONNECTOR ON?/
(2)  010731     200  047516  042040  NOACT:  .ASCIZ  <200>/NO DEVICES ARE SELECTED/
(2)  010762  100200  046513  030503  CONERR: .ASCIZ  <200><200>/KMC11 AT NONSTANDARD ADDRESS  PC: /
(2)  011027     200  054105  042520  CNERR:  .ASCIZ  <200>/EXPECTED  FOUND/
(2)  011050  024040  046513  024503  KMCM:   .ASCIZ  / (KMC) /
(2)                                   .EVEN
(2)  011060  000005                   XSTATQ: 5
1780 011062     006     003                   .BYTE   6,3
1781 011064  001276                           $TMP0
1782 011066     006     003                   .BYTE   6,3
1783 011070  001300                           $TMP1
1784 011072     006     003                   .BYTE   6,3
1785 011074  001302                           $TMP2
1786 011076     006     003                   .BYTE   6,3
1787 011100  001304                           $TMP3
1788 011102     006     002                   .BYTE   6,2
1789 011104  001306                           $TMP4
1790                                  .EVEN
1791
1792                                          ;BUFFERS FOR INPUT-OUTPUT
1793
1794 011106  000000                   TEMP:   0
1795         011150                   .=.+40
1796 011150  000000                   MDATA:  0
1797         011212                   .=.+40
1798
1799
1800                                          ;ROUTINE USED TO CHANGE SOFTWARE SWITCH
1801                                          ;REGISTER USING THE CONSOLE TERMINAL
1802                                          ;---------------------------------------
1803
1&J4 011212  022737  000176  001240  CKSWR:  CMP     #SWREG,SWR      ;IS THE SOFT SWR BEING USED?
1805 011220  001075                   BNE     CKSWR5          ;BR IF NO
1806 011222  132737  000001  001336  BITB    #1,$ENV         ; IS IT RUNNING UNDER APT?
1807 011230  001071                   BNE     CKSWR5          ; EXIT IF YES.
1808 011232  022777  000007  170006  CMP     #7,@$TKB        ;WAS CTRL G TYPED? (7 BIT ASCII)
1809 011240  001404                   BEQ     1$              ;BR IF YES
1810 011242  022777  000207  167776  CMP     #207,@$TKB      ;WAS CTRL G TYPED? (8 BIT ASCII)
1811 011250  001061                   BNE     CKSWR5          ;BR IF NO
1812 011252  010246          1$:      MOV     R2,-(SP)        ;STORE R2
1813 011254  010346                   MOV     R3,-(SP)        ;STORE R3
1814 011256  010446                   MOV     R4,-(SP)        ;STORE R4
1815 011260  012737  177777  011416   MOV     #-1,SWFLG       ;SET SOFT TYPE OUT FLAG
1816 011266  005002          CKSWR1:  CLR     R2              ;CLEAR NEW SWR CONTENTS
1817 011270  012704  177777           MOV     #-1,R4          ;SET FLAG TO ALL ONES
1818 011274  104401  005541           TYPE    ,$MSWR          ;TYPE "SWR= "
1819 011300  104417          CKSWR2:  CNVRT                   ;TYPE OUT PRESENT CONTENTS
1820 011302  011452                   SOFTSW                  ;OF SOFT SWITCH REGISTER
1821 011304  104401  005552  CKSWR3:  TYPE    ,$MNEW          ;TYPE "NEW? "
1822 011310  004737  011420  CKSWR4:  JSR     PC,INCHAR       ;GET RESPONSE
1823 011314  022703  000015           CMP     #15,R3          ;WAS IT A CR?
1824 011320  001424                   BEQ     5$              ;BR IF YES
```

```
1825  011322  022703  000012            CMP     #12,R3          ;WAS IT A LF?
1826  011326  001416                    BEQ     4$              ;BR IF YES
1827  011330  022703  000025            CMP     #25,R3          ;WAS IT CTRL U?
1828  011334  001754                    BEQ     CKSWR1          ;BR IF YES(START OVER)
1829  011336  022703  000007            CMP     #7,R3           ;IF CNTL G GET NEXT CHAR
1830  011342  001762                    BEQ     CKSWR4
1831  011344  005004                    CLR     R4              ;IT MUST BE A DIGIT SO CLR FLAG
1832  011346  042703  177770            BIC     #177770,R3      ;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
1833  011352  006302                    ASL     R2              ;SHIFT R2 3 TIMES
1834  011354  006302                    ASL     R2
1835  011356  006302                    ASL     R2
1836  011360  050302                    BIS     R3,R2           ;ADD LAST DIGIT
1837  011362  000752                    BR      CKSWR4          ;GET NEXT CHARACTER
1838  011364  012766  002402  000006  4$:     MOV     #.START,6(SP)   ;LF WAS TYPED SO GO TO START
1839  011372  005704            5$:     TST     R4              ;IS FLAG CLEAR?
1840  011374  001002                    BNE     6$              ;IF NOT DON'T CHANGE SOFT SWR
1841  011376  010277  167636            MOV     R2,@SWR         ;IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
1842  011402  005037  011416    6$:     CLR     SWFLG           ;CLEAR TYPEOUT FLAG
1843  011406  012604                    MOV     (SP)+,R4        ;RESTORE R4
1844  011410  012603                    MOV     (SP)+,R3        ;RESTORE R3
1845  011412  012602                    MOV     (SP)+,R2        ;RESTORE R2
1846  011414  000207            CKSWR5: RTS     PC              ;RETURN
1847
1848  011416  000000            SWFLG:  0
1849
1850  011420  105777  167620    INCHAR: TSTB    @$TKS
1851  011424  100375                    BPL     .-4
1852  011426  017703  167614            MOV     @$TKB,R3
1853  011432  105777  167612            TSTB    @$TPS
1854  011436  100375                    BPL     .-4
1855  011440  010377  167606            MOV     R3,@$TPB
1856  011444  042703  000200            BIC     #BIT7,R3
1857  011450  000207                    RTS     PC
1858
1859  011452  000001            SOFTSW: 1
1860  011454     006      002           .BYTE   6,2
1861  011456  000176                    SWREG
```

```
1862
1863                                    ;
1864                                    ;ROUTINE USED TO "CYCLE" THROUGH UP TO 16 KMC11'S
1865                                    ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
1866                                    ;AND RUNS THE SPECIFIED KMC11'S.  THIS ROUTINE *MUST*
1867                                    ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
1868                                    ;SETUP NECESSARY.
1869                                    ;
1870
1871  011460  005737  001470        CYCLE:  TST     KMACTV          ;ARE ANY KMC11'S TO BE TESTED?
1872  011464  001004                        BNE     1$              ;BR IF OK.
1873  011466  104401  010731                TYPE    ,NOACT          ;NO KMC11'S SELECTED!!
1874  011472  000000                        HALT                    ;STOP THE SHOW.
1875  011474  000776                        BR      .-2             ;DISQUALIFY CONT. SW.
1876  011476  000241                1$:     CLC                     ;CLEAR PROC. CARRY BIT.
1877  011500  006137  001500                ROL     RUN             ;UPDATE POINTER
1878  011504  005537  001500                ADC     RUN             ;CATCH CARRY FROM RUN
1879  011510  062737  000004  001504        ADD     #4,MILK         ;UPDATE POINTER
1880  011516  062737  000010  001502        ADD     #10,CREAM       ;UPDATE ADDRESS POINTER.
1881  011524  022737  002300  001502        CMP     #KM.MAP+200,CREAM
1882  011532  001006                        BNE     2$              ;KEEP GOING; NOT ALL TESTED FOR.
1883  011534  012737  002100  001502        MOV     #KM.MAP,CREAM   ;RESET ADDRESS POINTER.
1884  011542  012737  002302  001504        MOV     #CNT.MAP,MILK   ;RESET PASS COUNT POINTER
1885  011550  033737  001500  001470  2$:   BIT     RUN,KMACTV      ;IS THIS ONE ACTIVE?
1886  011556  001747                        BEQ     1$              ;BR IF NO
1887  011560  013700  001502                MOV     CREAM,R0        ;GET ADDRESS POINTER
1888  011564  013702  001504                MOV     MILK,R2         ;GET PASS COUNT POINTER
1889  011570  012037  002066                MOV     (R0)+,KMCSR     ;LOAD SYSTEM CTRL. REG
1890  011574  011037  002056                MOV     (R0),KMRVEC     ;LOAD VECTOR
1891  011600  042737  177000  002056        BIC     #177000,KMRVEC  ;CLEAR UNWANTED BITS
1892  011606  012037  002050                MOV     (R0)+,STAT1     ;LOAD STAT1
1893  011612  012037  002052                MOV     (R0)+,STAT2     ;LOAD STAT2
1894  011616  012037  002054                MOV     (R0)+,STAT3     ;LOAD STAT3
1895  011622  012237  001324                MOV     (R2)+,$PASS     ;LOAD PASS COUNT
1896  011626  012237  001212                MOV     (R2)+,$ERTTL    ;LOAD ERROR COUNT
1897  011632  012700  000002                MOV     #2,R0           ;SAVE CORE THIS WAY!
1898  011636  013737  002066  002070        MOV     KMCSR,KMCSRH
1899  011644  005237  002070                INC     KMCSRH
1900  011650  013737  002070  002072        MOV     KMCSRH,KMCTL
1901  011656  005237  002072                INC     KMCTL
1902  011662  013737  002072  002074        MOV     KMCTL,KMP04
1903  011670  060037  002074                ADD     R0,KMP04
1904  011674  013737  002074  002076        MOV     KMP04,KMP06
1905  011702  060037  002076                ADD     R0,KMP06
1906
1907  011706  013737  002056  002060        MOV     KMRVEC,KMRLVL   ;PTY LVL
1908  011714  060037  002060                ADD     R0,KMRLVL       ;
1909  011720  013737  002060  002062        MOV     KMRLVL,KMTVEC   ;TX VEC
1910  011726  060037  002062                ADD     R0,KMTVEC       ;
1911  011732  013737  002062  002064        MOV     KMTVEC,KMTLVL   ;TX LVL
1912  011740  060037  002064                ADD     R0,KMTLVL
1913
1914  011744  032737  000002  001446        BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED
1915  011752  001447                        BEQ     7$              ;BR IF NO
1916  011754                         4$:
1917  011754  005737  000042                TST     @#42            ;RUNNING IN AUTO MODE?
```

```
1918  011760  001044                        BNE     7$              ;BR IF YES
1919  011762  104401  001313                TYPE    ,$CRLF
1920  011766  104415                        INPUT
1921  011770  010003                        MTSTN
1922  011772  000001                        1
1923  011774  001000                        1000
1924  011776  001202                        $TSTNM
1925  012000     000                .BYTE   0
1926  012001     001                .BYTE   1
1927  012002  012700  013732                MOV     #TST1,R0
1928  012006  022710                5$:     CMP     (PC)+,(R0)      ;CMP FIRST WORD TO 12737
1929  012010  012737                        MOV     (PC)+,@(PC)+
1930  012012  001020                        BNE     6$              ;BR IF NOT SAME
1931  012014  023760  001202  000002        CMP     $TSTNM,2(R0)    ;DOES $TSTNM MATCH?
1932  012022  001014                        BNE     6$              ;BR IF NO
1933  012024  022760  001202  000004        CMP     #$TSTNM,4(R0)   ;IS LAST WORD OK?
1934  012032  001010                        BNE     6$              ;BR IF NO
1935  012034  010037  001206                MOV     R0,$LPADR       ;IT IS A LEGAL TEST SO DO IT
1936  012040  104401  007642                TYPE    ,MR
1937  012044  042737  000002  001446        BIC     #SW01,STRTSW
1938  012052  000412                        BR      8$
1939  012054  005720                6$:     TST     (R0)+           ;POP R0
1940  012056  020027  033114                CMP     R0,#TLAST+10    ;AT END YET?
1941  012062  001351                        BNE     5$              ;BR IF NO
1942  012064  104401  001312                TYPE    ,$QUES          ;YES ILLEGAL TEST NO.
1943  012070  000731                        BR      4$              ;TRY AGAIN
1944
1945  012072  012737  013732  001206  7$:   MOV     #TST1,$LPADR    ;PREPARE $LPADR ADDRESS
1946  012100  013701  002066          8$:   MOV     KMCSR,R1        ;R1 = BASE KMC11 ADDRESS
1947  012104  000177  167076                JMP     @$LPADR         ;GO START TESTING.
1948
1949
1950                                        ;ROUTINE USED TO "AUTO SIZE" THE KMC11
1951                                        ;CSR AND VECTOR.
1952                                        ;NOTE:  THE CSR MAY BE ANY WHERE IN THE FLOATING
1953                                        ;       ADDRESS RANGE (160000:164000)
1954                                        ;       AND THE VECTOR MAY BE ANY WHERE IN THE
1955                                        ;       FLOATING VECTOR RANGE (300:770)
1956                                        ;
1957
1958  012110                        AUTO.SIZE:
1959  012110  000005                        RESET                   ;INSURE A BUS INIT.
1960  012112  012702  002100        CSRMAP: MOV     #KM.MAP,R2      ;LOAD MAP POINTER.
1961  012116  005022                1$:     CLR     (R2)+           ;ZERO ENTIRE MAP
1962  012120  022702  002300                CMP     #KM.END,R2      ;ALL DONE?
1963  012124  001374                        BNE     1$              ;BR IF NO
1964  012126  005037  001472                CLR     KMNUM           ;SET OCTAL NUMBER OF KMC11'S TO 0
1965  012132  012702  002100                MOV     #KM.MAP,R2      ;R2 POINTS TO KMC MAP
1966  012136  005037  001470                CLR     KMACTV          ;CLEAR ACTIVE
1967  012142  032737  000001  001446        BIT     #SW00,STRTSW    ;QUESTIONS?
1968  012150  001002                        BNE     .+6             ;BR IF YES
1969  012152  000137  012532                JMP     7$              ;IF NO SKIP QUESTIONS
1970  012156  012737  000001  001306        MOV     #1,$TMP4        ;START WITH 1
1971  012164  104415                        INPUT
1972  012166  010323                        NUM
1973  012170  000001                        1
```

```
 1974  012172  000020                        16.
 1975  012174  001302                        $TMP2
 1976  012176     000                        .BYTE   0
 1977  012177     001                        .BYTE   1
 1978  012200  013737  001302  001472        MOV     $TMP2,KMNUM     ;KMNUM = HOW MANY
 1979  012206  104401  001313          12$:  TYPE    ,$CRLF
 1980  012212  104416                        CONVRT                  ;TYPE WHICH KMC IS BEING DONE
 1981  012214  013164                        WHICH                   ;$TMP4 IS WHICH KMC
 1982  012216  005237  001306                INC     $TMP4
 1983  012222  104415                        INPUT
 1984  012224  010363                        CSR
 1985  012226  160000                        160000
 1986  012230  164000                        164000
 1987  012232  001304                        $TMP3
 1988  012234     000                        .BYTE   0
 1989  012235     001                        .BYTE   1
 1990  012236  013722  001304                MOV     $TMP3,(R2)+     ;STORE CSR IN MAP
 1991  012242  104415                        INPUT
 1992  012244  010401                        VEC
 1993  012246  000000                        0
 1994  012250  000776                        776
 1995  012252  001304                        $TMP3
 1996  012254     000                        .BYTE   0
 1997  012255     001                        .BYTE   1
 1998  012256  013712  001304                MOV     $TMP3,(R2)      ;STORE VECTOR IN MAP
 1999  012262  104401          10$:  TYPE
 2000  012264  010422                        PRIO                    ;ASK WHAT BR LEVEL
 2001  012266  004737  013456                JSR     PC,INTTY        ;GET RESPONSE
 2002  012272  022703  000024                CMP     #24,R3          ;
 2003  012276  101014                        BHI     50$             ;BR IF LESS THAN 4
 2004  012300  022703  000027                CMP     #27,R3          ;
 2005  012304  103411                        BLO     50$             ;BR IF GREATER THAN 7
 2006  012306  012704  000011                MOV     #11,R4          ;R4 = NUMBER OF SHIFTS
 2007  012312  006303                        ASL     R3              ;SHIFT R3 LEFT
 2008  012314  005304                        DEC     R4              ;DEC SHIFT COUNT
 2009  012316  001375                        BNE     .-4             ;BR IF NOT DONE
 2010  012320  042703  170777                BIC     #170777,R3      ;BIC UNWANTED BITS
 2011  012324  050312                        BIS     R3,(R2)         ;PUT BR LEVEL IN STATUS MAP
 2012  012326  000403                        BR      8$              ;CONTINUE
 2013  012330  104401          50$:  TYPE
 2014  012332  001312                        $QUES                   ;RESPONSE IS OUT OF LIMITS
 2015  012334  000752                        BR      10$             ;TRY AGAIN
 2016  012336                        8$:
 2017  012336                        9$:
 2018  012336  104401          16$:  TYPE
 2019  012340  010461                        MODU                    ;ASK WHICH LINE UNIT
 2020  012342  004737  013456                JSR     PC,INTTY        ;GET REPLY
 2021  012346  022703  000021                CMP     #21,R3          ;"1"
 2022  012352  001417                        BEQ     30$
 2023  012354  022703  000022                CMP     #22,R3          ;"2"
 2024  012360  001412                        BEQ     31$
 2025  012362  022703  000116                CMP     #116,R3         ;"N"
 2026  012366  001403                        BEQ     32$
 2027  012370  104401                        TYPE
 2028  012372  001312                        $QUES                   ;IF NOT A 1,2 OR N TYPE "?"
 2029  012374  000760                        BR      16$             ;TRY AGIAN
```

```
2030  012376  052722  010000          32$:    BIS     #BIT12,(R2)+    ;SET BIT 12 IN STAT2 IF NO LU
2031  012402  022222                          CMP     (R2)+,(R2)+     ;POP OVER STAT2 AND STAT3
2032  012404  000445                          BR      33$
2033  012406  052712  020000          31$:    BIS     #BIT13,(R2)     ;SET BIT 13 IN STAT2 IF M8202
2034  012412  104401                  30$:    TYPE
2035  012414  010671                          CONN                    ;ASK IF LOOP-BACK IS ON
2036  012416  004737  013456                  JSR     PC,INTTY        ;GET REPLY
2037  012422  022703  000131                  CMP     #131,R3         ;Y
2038  012426  001406                          BEQ     17$
2039  012430  022703  000116                  CMP     #116,R3         ;N
2040  012434  001406                          BEQ     18$
2041  012436  104401                          TYPE
2042  012440  001312                          $QUES                   ;IF NOT Y OR N TYPE "?"
2043  012442  000763                          BR      30$             ;TRY AGAIN
2044  012444  052722  040000          17$:    BIS     #BIT14,(R2)+    ;TURNAROUND IS CONNECTED
2045  012450  000402                          BR      19$
2046  012452  042722  040000          18$:    BIC     #BIT14,(R2)+    ;NO TURNAROUND
2047  012456                          19$:
2048  012456  104415                          INPUT
2049  012460  010573                          LINE
2050  012462  000000                          0
2051  012464  000377                          377
2052  012466  001304                          $TMP3
2053  012470     000                          .BYTE   0
2054  012471     001                          .BYTE   1
2055  012472  113722  001304                  MOVB    $TMP3,(R2)+     ;STORE SWITCH PAC IN MAP
2056  012476  104415                          INPUT
2057  012500  010631                          BM
2058  012502  000000                          0
2059  012504  000377                          377
2060  012506  001304                          $TMP3
2061  012510     000                          .BYTE   0
2062  012511     001                          .BYTE   1
2063  012512  113722  001304                  MOVB    $TMP3,(R2)+     ;STORE SWITCH PAC IN MAP
2064  012516  005722                          TST     (R2)+           ;POP OVER STAT3
2065  012520  005337  001302          33$:    DEC     $TMP2           ;DEC KMC COUNT
2066  012524  001230                          BNE     12$             ;BR IF MORE TO DO
2067  012526  000137  013064                  JMP     13$             ;CONTINUE
2068  012532  012701  160000          7$:     MOV     #160000,R1      ;SET FOR FIRST ADDRESS TO BE TESTED
2069  012536  012737  013156  000004          MOV     #6$,@#4         ;SET FOR NON-EXISTANT DEVICE TIME OUT
2070  012544  005011                  2$:     CLR     (R1)            ;CLEAR SEL0
2071  012546  005711                          TST     (R1)            ;IF KMC11 KMCSR S/B 0
2072  012550  001135                          BNE     3$              ;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO KMC11
2073  012552  005061  000006                  CLR     6(R1)           ;CLEAR SEL6
2074  012556  005761  000006                  TST     6(R1)           ;IF KMC11 THEN KMRIC S/B =0!
2075  012562  001130                          BNE     3$              ;BR IF NOT KMC11
2076  012564  012711  002000                  MOV     #BIT10,(R1)     ;SET ROMO
2077  012570  005061  000004                  CLR     4(R1)           ;CLEAR SEL4
2078  012574  012761  125252  000006          MOV     #125252,6(R1)   ;WRITE THIS TO SEL6
2079  012602  052711  020000                  BIS     #BIT13,(R1)     ;WRITE IT!
2080  012606  022761  125252  000004          CMP     #125252,4(R1)   ;WAS IT WRITTEN?
2081  012614  001113                          BNE     3$              ;IF NO IT IS NOT CRAM
2082                          ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A KMC11 CSR ADDRESS.
2083  012616                  21$:
2084  012616  010122                  22$:    MOV     R1,(R2)+        ;STORE CSR IN CORE TABLE.
2085  012620  012711  001000          15$:    MOV     #BIT9,(R1)      ;CLEAR LINE UNIT LOOP
```

```
2086  012624  005061  000004          CLR     4(R1)                   ;CLEAR PORT4
2087  012630  012761  122113  000006  MOV     #122113,6(R1)           ;LOAD INSTRUCTION (CLR DTR)
2088  012636  052711  000400          BIS     #BIT8,(R1)              ;CLOCK INSTRUCTION
2089  012642  012761  021264  000006  MOV     #021264,6(R1)           ;LOAD INSTRUCTION
2090  012650  052711  000400          BIS     #BIT8,(R1)              ;CLOCK INSTRUCTION
2091  012654  122761  000377  000004  CMPB    #377,4(R1)              ;IS IT ALL ONES?
2092  012662  001003                  BNE     .+10                    ;BR IF NO
2093  012664  052712  010000          BIS     #BIT12,(R2)             ;IF YES, NO LINE UNIT, SET STATUS BIT
2094  012670  000436                  BR      20$
2095  012672  032761  000002  000004  BIT     #BIT1,4(R1)             ;IS SWITCH A ONE?
2096  012700  001403                  BEQ     .+10                    ;BR IF M8201
2097  012702  052712  060000          BIS     #BIT13!BIT14,(R2) ;M8202 ASSUME CONNECTOR
2098  012706  000427                  BR      20$                     ;CONNECTOR ON)
2099  012710  032761  000010  000004  BIT     #BIT3,4(R1)             ;IS MRDY SET
2100  012716  001023                  BNE     20$                     ;BR IF M8201 NO CONNECTOR (ON LINE)
2101  012720  012761  000100  000004  MOV     #BIT6,4(R1)             ;LOAD PORT4
2102  012726  012761  122113  000006  MOV     #122113,6(R1)           ;LOAD INSTRUCTION
2103  012734  052711  000400          BIS     #BIT8,(R1)              ;CLOCK INSTRUCTION(SET DTR)
2104  012740  012761  021264  000006  MOV     #021264,6(R1)           ;LOAD INSTRUCTION
2105  012746  052711  000400          BIS     #BIT8,(R1)              ;CLOCK INSTRUCTION(READ MODEM REG)
2106  012752  032761  000010  000004  BIT     #BIT3,4(R1)             ;IS MRDY SET NOW?
2107  012760  001402                  BEQ     20$                     ;BR IF NO CONNECTOR
2108  012762  052712  040000          BIS     #BIT14,(R2)             ;SET STATUS BIT FOR CONNECTOR
2109  012766  005722          20$:    TST     (R2)+                   ;POP POINTER
2110  012770  012761  021324  000006  MOV     #021324,6(R1)           ;PUT INSTRUCTION IN PORT6
2111  012776  012711  001400          MOV     #BIT9!BIT8,(R1)         ;PORT4←LU 15
2112  013002  156122  000004          BISB    4(R1),(R2)+             ;STORE DDCMP LINE # IN TABLE
2113  013006  012761  021344  000006  MOV     #021344,6(R1)           ;PORT6←INSTRUCTION
2114  013014  012711  001400          MOV     #BIT8!BIT9,(R1)         ;CLOCK INSTR.
2115  013020  156122  000004          BISB    4(R1),(R2)+             ;STORE BM873 ADD IN TABLE
2116  013024  005722                  TST     (R2)+                   ;POP OVER STAT3
2117  013026  005011                  CLR     (R1)                    ;CLEAR ROMI
2118  013030  005237  001472          INC     KMNUM                   ;UPDATE DEVICE COUNTER
2119  013034  022737  000020  001472  CMP     #20,KMNUM               ;ARE MAX. NO. OF DEV FOUND?
2120  013042  001410                  BEQ     13$                     ;YES DON'T LOOK FOR ANY MORE.
2121  013044  005011          3$:     CLR     (R1)                    ;CLEAR BIT 10
2122  013046  005061  000006          CLR     6(R1)                   ;CLEAR SEL 6
2123  013052  062701  000010  14$:    ADD     #10,R1                  ;UPDATE CSR POINTER ADDRESS
2124  013056  022701  164000          CMP     #164000,R1
2125  013062  001230                  BNE     2$                      ;BR IF MORE ADDRESS TO CHECK.
2126  013064  005037  001470  13$:    CLR     KMACTV
2127  013070  005737  001472          TST     KMNUM                   ;WERE ANY KMC11'S FOUND AT ALL?
2128  013074  001423                  BEQ     5$                      ;ERROR AUTO SIZER FOUND NO KMC11'S IN THIS SYS.
2129  013076  013701  001472          MOV     KMNUM,R1
2130  013102  010137  001476          MOV     R1,SAVNUM               ;SAVE NUMBER OF DEVICES
2131  013106  000241          4$:     CLC
2132  013110  006137  001470          ROL     KMACTV                  ;GENERATE ACTIVE REGISTER OF DEVICES.
2133  013114  005237  001470          INC     KMACTV                  ;SET THE BIT
2134  013120  005301                  DEC     R1
2135  013122  001371                  BNE     4$                      ;BR IF MORE TO GENERATE
2136  013124  012737  000006  000004  MOV     #6,@#4                  ;RESTORE TRAP VECTOR
2137  013132  013737  001470  001474  MOV     KMACTV,SAVACT           ;SAVE ACTIVE REGISTER
2138  013140  000137  013172          JMP     VECMAP                  ;GO FIND THE VECTOR NOW.
2139  013144  104401  007645  5$:     TYPE    ,MERR2                  ;NOTIFY OPR THAT NO KMC11'S FOUND.
2140  013150  005000                  CLR     R0                      ;MAKE DATA LIGHTS ZERO
2141  013152  000000                  HALT                            ;STOP THE SHOW
```

```
2142  013154  000776                    BR      .-2             ;DISABLE CONT. SW.
2143  013156  012716  013052    6$:     MOV     #1$,(SP)        ;ENTERED BY NON-EXISTANT TIME-OUT.
2144  013162  000002                    RTI                     ;RETURN TO MAINSTREAM
2145
2146  013164  000001            WHICH:  1
2147  013166      002  002             .BYTE   2,2
2148  013170  001306                    $TMP4
2149
2150  013172  032737  000001  001446  VECMAP: BIT     #SW00,STRTSW
2151  013200  001114                    BNE     5$
2152  013202  012737  000340  000022          MOV     #340,@#22       ;SET IOT TRAP PRIO TO 7
2153  013210  012737  013364  000020          MOV     #4$,@#20        ;SET IOT TRAP VECTOR
2154  013216  012702  002100                  MOV     #KM.MAP,R2      ;SET SOFTWARE POINTER
2155  013222  012700  000300                  MOV     #300,R0         ;FLOATING VECTORS START HERE.
2156  013226  012701  000302                  MOV     #302,R1         ;PC OF IOT INSTR.
2157  013232  010120            1$:     MOV     R1,(R0)+        ;START FILLING VECTOR AREA
2158  013234  012721  000004                  MOV     #4,(R1)+        ;WITH .+2; IOT
2159  013240  022021                    CMP     (R0)+,(R1)+     ;ADD 2 TO R0 +R1
2160  013242  020127  001000                  CMP     R1,#1000
2161  013246  101771                    BLOS    1$              ;BR IF MORE TO FILL
2162  013250  013737  001470  001276          MOV     KMACTV,$TMP0    ;STORE TEMPORALLY
2163  013256  006037  001276    2$:     ROR     $TMP0           ;BRING OUT A BIT
2164  013262  103063                    BCC     5$              ;BR IF ALL DONE
2165  013264  012704  000012                  MOV     #12,R4          ;R4 IS INDEX REGISTER
2166  013270  016437  013442  177776          MOV     BRLVL(R4),PS    ;SET PS TO 7
2167  013276  011201                    MOV     (R2),R1
2168  013300  012761  000200  000004          MOV     #200,4(R1)
2169  013306  012711  001000                  MOV     #BIT9,(R1)      ;SET ROMI
2170  013312  012761  121111  000006          MOV     #121111,6(R1)   ;PUT INSTRUCTION IN PORT6
2171  013320  012711  001400                  MOV     #BIT9!BIT8,(R1) ;FORCE AN INTERRUPT
2172  013324  105200            7$:     INCB    R0              ;STALL
2173  013326  001376                    BNE     .-2             ;FOR TIME TO INTERUPT
2174  013330  162704  000002                  SUB     #2,R4           ;GET NEXT LOWEST PS LEVEL
2175  013334  001404                    BEQ     6$              ;BR IF R4 = 0
2176  013336  016437  013442  177776          MOV     BRLVL(R4),PS    ;MOVE NEXT LOWER LEVEL IN PS
2177  013344  000767                    BR      7$              ;BR TO DELAY
2178  013346  052762  005300  000002  6$:     BIS     #5300,2(R2)     ;NO INTERUPT ASSUME 300 AT LEVEL 5 AND FIX KMC11 LATER
2179  013354  005011            3$:     CLR     (R1)            ;CLEAR ROMI
2180  013356  062702  000010                  ADD     #10,R2          ;POP SOFTWARE POINTER
2161  013362  000735                    BR      2$              ;KEEP GOING
2182  013364  051662  000002    4$:     BIS     (SP),2(R2)      ;GET VECTOR ADDRESS
2183  013370  042762  000007  000002          BIC     #7,2(R2)        ;CLEAR JUNK
2184  013376  016405  013444                  MOV     BRLVL+2(R4),R5  ;GET BR LEVEL OF KMC11
2185  013402  006305                    ASL     R5              ;SHIFT LEVEL 4 PLACES
2186  013404  006305                    ASL     R5              ;TO THE LEFT FOR THE
2187  013406  006305                    ASL     R5              ;STATUS TABLE
2188  013410  006305                    ASL     R5
2189  013412  042705  170777                  BIC     #170777,R5      ;CLEAR UNWANTED BITS
2190  013416  050562  000002                  BIS     R5,2(R2)        ;PUT BR LEVEL IN STATUS TABLE
2191  013422  022626                    CMP     (SP)+,(SP)+     ;POP IOT JUNK OFF STACK
2192  013424  012716  013354                  MOV     #3$,(SP)        ;SET FOR RETURN
2193  013430  000002                    RTI
2194  013432  012737  004134  000020  5$:     MOV     #$SCOPE,@#20    ; RESTORE SCOPE VECTOR
2195  013440  000207                    RTS     PC              ;ALL DONE WITH "AUTO SIZING"
2196
2197  013442  000000            BRLVL:  PRO     ;LEVEL 0
```

```
2198  013444  000000                        PR0     ;LEVEL 0
2199  013446  000200                        PR4     ;LEVEL 4
2200  013450  000240                        PR5     ;LEVEL 5
2201  013452  000300                        PR6     ;LEVEL 6
2202  013454  000340                        PR7     ;LEVEL 7
2203
2204
2205  013456  105777  165562        INTTY:  TSTB    @STKS           ;WAIT FOR DONE
2206  013462  100375                        BPL     .-4
2207  013464  017703  165556                MOV     @STKB,R3        ;PUT CHAR IN R3
2208  013470  105777  165554                TSTB    @STPS           ;WAIT UNTIL PRINTER IS READY
2209  013474  100375                        BPL     .-4
2210  013476  010377  165550                MOV     R3,@STPB        ;ECHO CHAR
2211  013502  042703  000240                BIC     #BIT7!BIT5,R3   ;MASK OFF LOWER CASE
2212  013506  000207                        RTS     PC              ;RETURN
2213
2214  013510                        APT.SIZE:
2215  013510  000005                        RESET
2216  013512  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2217  013514  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2218  013516  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2219  013520  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2220  013522  005037  013724                CLR     VECTR           ; CLEAR THE LOCAL VARIABLE
2221  013526  005037  013730                CLR     PRIRTY          ; CLEAN UP LOCAL VARIABLE
2222  013532  013700  001376                MOV     $CDWI,R0        ; GET THE DEVICE COUNT
2223  013536  010037  001476                MOV     R0,SAVNUM       ; SAVE THE NO. OF DEVICES
2224  013542  012701  001346                MOV     #SPARMS1,R1     ; GET EXTRA INFO, BITS POINTER
2225  013546  013737  001372  013726        MOV     $BASE,BASE      ; GET BASE CSR ADDRESS
2226  013554  113737  001366  013724        MOVB    $VECTI,VECTR    ; GET THE VECTOR
2227  013562  113737  001367  013730        MOVB    $VECTI+1,PRIRTY ; GET THE PRIORITY
2228  013570  013737  001374  001470        MOV     $DEVM,KMACTV    ; SAVE THE KMC'S SELECTED ACTIVE
2229  013576  013737  001470  001474        MOV     KMACTV,SAVACT   ; SAVE THE ACTIVE REGISTER
2230  013604  012702  001402                MOV     #$DDWD,R2       ; GET ADDRESS OF FIRST DEVICE DESCRIPTOR WORD
2231  013610  012703  002100                MOV     #KM.MAP,R3      ; GET POINTER TO DEVICE MAP
2232  013614  005023                3$:     CLR     (R3)+           ; CLEAR DEVICE MAP
2233  013616  022703  002300                CMP     #KM.END,R3      ; IS WHOLE DEV.MAP CLEARED?
2234  013622  003374                        BGT     3$              ; NO, THEN GO ON.
2235  013624  012703  002100                MOV     #KM.MAP,R3      ; RESTORE DEV.MAP POINTER.
2236  013630  013723  013726        1$:     MOV     BASE,(R3)+      ; LOAD CSR ADDRESS
2237  013634  112163  000001                MOVB    (R1)+,1(R3)     ; GET EXTRA INFO. BITS
2238  013640  006213                        ASR     (R3)            ; SET IT IN RIGHT POSITION.
2239  013642  006213                        ASR     (R3)            ; SET IT IN RIGHT POSITION.
2240  013644  053713  013730                BIS     PRIRTY,(R3)     ; GET PRIORITY IN STAT1
2241  013650  006313                        ASL     (R3)            ; SET THEM IN RIGHT POSITION
2242  013652  006313                        ASL     (R3)            ;  "   "   "   "   "
2243  013654  006313                        ASL     (R3)            ;  "   "   "   "    "
2244  013656  006313                        ASL     (R3)            ;  "   "   "   "    "
2245  013660  053723  013724                BIS     VECTR,(R3)+     ; GET THE VECTOR IN STAT1.
2246  013664  012223                        MOV     (R2)+,(R3)+     ; GET THE STAT2 FROM DDWXX
2247  013666  005723                        TST     (R3)+           ; SKIP OVER STAT3
2248  013670  005300                        DEC     R0              ; COUNT BY 1
2249  013672  001407                        BEQ     2$              ; ALL DONE?
2250  013674  062737  000010  013726        ADD     #10,BASE        ; INCREMENT BASE CSR ADDRESS BY 10
2251  013702  062737  000010  013724        ADD     #10,VECTR       ; INCREMENT VECTOR ADDRESS BY 10
2252  013710  000747                        BR      1$              ; SET THE NEXT MAP ENTRY
2253  013712                        2$:
```

```
2254  013712  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
2255  013714  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
2256  013716  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
2257  013720  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
2258  013722  000207                        RTS     PC              ; RETURN
2259  013724  000000              VECTR:    .WORD   0
2260  013726  000000              BASE:     .WORD   0
2261  013730  000000              PRIRTY:   .WORD   0
2262
2263
2264
2265                                ;*************************** TEST 1 ****************************
2266                                ;*OUT CONTROL REGISTER READ/ONLY TEST
2267                                ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
2268                                ;*BITS ARE IN THE CORRECT STATE
2269                                ;;***********************************************************
2270
2271                                ;  TEST 1
2272                                ;--------------
2273                              ;;************************************************************
2274  013732  000004              TST1:     SCOPE
2275  013734  012737  000001  001202        MOV     #1,$TSTNM               ; LOAD THE NO. OF THIS TEST
2276  013742  012737  014006  001442        MOV     #TST2,NEXT              ; POINT TO THE START OF NEXT TEST.
2277                                ;R1 CONTAINS BASE KMC11 ADDRESS
2278  013750  005077  166112               CLR     @KMCSR          ;CLEAR SELO
2279  013754  012702  000011               MOV     #11,R2          ;SAVE R2 FOR TYPEOUT
2280  013760  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2281  013762  021224                       021004!<20*11>          ;PORT4+LINE UNIT REG 11
2282  013764  016104  000004               MOV     4(R1),R4        ;PUT "FOUND" IN R4
2283  013770  042704  000054               BIC     #54,R4          ;CLEAR UNKNOWN BITS
2284  013774  012705  000020               MOV     #20,R5          ;PUT "EXPECTED" IN R5
2285  014000  120504                       CMPB    R5,R4           ;IS OUT READY SET?
2286  014002  001401                       BEQ     1$              ;BR IF YES
2287  014004  104002                       ERROR   2               ;ERROR IN LU 11
2288  014006                       1$:
2289
2290
2291                                ;*************************** TEST 2 ****************************
2292                                ;*IN CONTROL REGISTER READ/ONLY TEST
2293                                ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
2294                                ;*BITS ARE IN THE CORRECT STATE
2295                                ;;***********************************************************
2296
2297                                ;  TEST 2
2298                                ;--------------
2299                              ;;************************************************************
2300  014006  000004              TST2:     SCOPE
2301  014010  012737  000002  001202        MOV     #2,$TSTNM               ; LOAD THE NO. OF THIS TEST
2302  014016  012737  014054  001442        MOV     #TST3,NEXT              ; POINT TO THE START OF NEXT TEST.
2303                                ;R1 CONTAINS BASE KMC11 ADDRESS
2304  014024  012702  000012               MOV     #12,R2          ;SAVE R2 FOR TYPEOUT
2305  014030  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2306  014032  021244                       021004!<20*12>          ;PORT4+LINE UNIT REG 12
2307  014034  016104  000004               MOV     4(R1),R4        ;PUT "FOUND" IN R4
2308  014040  042704  000017               BIC     #17,R4          ;CLEAR UNKNOWN BITS
2309  014044  005005                       CLR     R5              ;PUT "EXPECTED" IN R5
```

```
2310  014046  120504                        CMPB    R5,R4           ;ARE ALL BITS CLEARED?
2311  014050  001401                        BEQ     1$              ;BR IF YES
2312  014052  104002                        ERROR   2               ;ERROR IN LU 12
2313  014054                          1$:
2314
2315
2316                                        ;*************************** TEST 3 ****************************
2317                                        ;*MODEM CONTROL REGISTER READ/ONLY TEST
2318                                        ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
2319                                        ;*BITS ARE IN THE CORRECT STATE
2320                                        ;;***********************************************************************
2321
2322                                        ;  TEST 3
2323                                        ;-------------
2324                                    ;;**********************************************************************
2325  014054  000004                    TST3:   SCOPE
2326  014056  012737  000003  001202            MOV     #3,$TSTNM               ; LOAD THE NO. OF THIS TEST
2327  014064  012737  014126  001442            MOV     #TST4,NEXT              ; POINT TO THE START OF NEXT TEST.
2328                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
2329  014072  104410                            MSTCLR                  ;MASTER CLEAR KMC11
2330  014074  012702  000013                    MOV     #13,R2          ;SAVE R2 FOR TYPEOUT
2331  014100  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2332  014102  021264                            021004!<20*13>          ;PORT4+LINE UNIT REG 13
2333  014104  016104  000004                    MOV     4(R1),R4        ;PUT "FOUND" IN R4
2334  014110  042704  000213                    BIC     #213,R4         ;CLEAR UNKNOWN BITS
2335  014114  012705  000100                    MOV     #100,R5         ;PUT "EXPECTED" IN R5
2336  014120  120504                            CMPB    R5,R4           ;ARE RING, DTR,  AND MODEM READY SET?
2337  014122  001401                            BEQ     1$              ;BR IF YES
2338  014124  104002                            ERROR   2               ;ERROR IN LU 13
2339  014126                          1$:
2340
2341
2342                                        ;*************************** TEST 4 ****************************
2343                                        ;*MAINTENANCE REGISTER READ/ONLY TEST
2344                                        ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
2345                                        ;*BITS ARE IN THE CORRECT STATE
2346                                        ;;***********************************************************************
2347
2348                                        ;  TEST 4
2349                                        ;-------------
2350                                    ;;**********************************************************************
2351  014126  000004                    TST4:   SCOPE
2352  014130  012737  000004  0012u2            MOV     #4,$TSTNM               ; LOAD THE NO. OF THIS TEST
2353  014136  012737  014220  001442            MOV     #TST5,NEXT              ; POINT TO THE START OF NEXT TEST.
2354                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
2355  014144  104410                            MSTCLR                  ;MASTER CLEAR KMC11
2356  014146  012702  000017                    MOV     #17,R2          ;SAVE R2 FOR TYPEOUT
2357  014152  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2358  014154  021364                            021004!<20*17>          ;PORT4+LINE UNIT REG 17
2359  014156  016104  000004                    MOV     4(R1),R4        ;PUT "FOUND" IN R4
2360  014162  042704  000206                    BIC     #206,R4         ;CLEAR UNKNOWN BITS
2361  014166  012705  000051                    MOV     #51,R5          ;PUT "EXPECTED" IN R5
2362  014172  032737  020000  002050            BIT     #BIT13,STAT1    ;IS LU AN M8202 OR M8201?
2363  014200  001404                            BEQ     .+12            ;BR IF M8201
2364  014202  042704  000040                    BIC     #40,R4          ;MASK OFF SI BIT IF M8202
2365  014206  042705  000040                    BIC     #BIT5,R5        ;SI BIT IS UNKNOWN ON AN M8202
```

```
2366  014212  120504                        CMPB    R5,R4           ;ARE SI AND ICIR SET?
2367  014214  001401                        BEQ     1$              ;BR IF YES
2368  014216  104002                        ERROR   2               ;ERROR IN LU 17
2369  014220                          1$:
2370
2371
2372                                        ;*************************** TEST 5 ****************************
2373                                        ;*LINE UNIT REGISTER WRITE/READ TEST
2374                                        ;*SET BITS IN LU REGISTER 12, VERIFY IT IS SET
2375                                        ;*CLEAR BITS IN LU REGISTER 12, VERIFY IT IS CLEAR
2376                                        ;;***********************************************************
2377
2378                                        ;  TEST 5
2379                                        ;-------------
2380                                    ;;**********************************************************
2381  014220  000004                TST5:   SCOPE
2382  014222  012737  000005  001202        MOV     #5,$TSTNM               ; LOAD THE NO. OF THIS TEST
2383  014230  012737  014360  001442        MOV     #TST6,NEXT              ; POINT TO THE START OF NEXT TEST.
2384  014236  012737  014252  001444        MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2385                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
2386  014244  104410                        MSTCLR                  ;MASTER CLEAR KMC11
2387  014246  012702  000012                MOV     #12,R2          ;SAVE REGISTER ADDRESS FOR TYPEOUT
2388  014252  012761  000040  000004  1$:   MOV     #40,4(R1)       ;LOAD PORT4
2389  014260  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2390  014262  122112                        122112                  ;SET BITS IN LU-12
2391  014264  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2392  014266  021245                        021245                  ;READ LU-12
2393  014270  012705  000040                MOV     #40,R5          ;PUT "EXPECTED" IN R5
2394  014274  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" IN R4
2395  014300  042704  000337                BIC     #337,R4         ;CLEAR UNWANTED BITS
2396  014304  120504                        CMPB    R5,R4           ;IS BITS SET?
2397  014306  001401                        BEQ     2$              ;BR IF YES
2398  014310  104003                        ERROR   3               ;ERROR, BIT 5 IS NOT SET
2399  014312  104405                  2$:   SCOP1                   ;SCOPE SUBTEST (SW09=1)
2400  014314  012737  014322  001444        MOV     #3$,LOCK        ;NEW SCOP1
2401  014322  005061  000004          3$:   CLR     4(R1)           ;LOAD PORT4
2402  014326  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2403  014330  122112                        122112                  ;CLEAR BIT 5 IN LU-12
2404  014332  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2405  014334  021245                        021245                  ;READ LU-12
2406  014336  005005                        CLR     R5              ;PUT "EXPECTED" IN R5
2407  014340  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" IN R4
2408  014344  042704  000337                BIC     #337,R4         ;CLEAR UNWANTED BITS
2409  014350  120504                        CMPB    R5,R4           ;IS BITS CLEAR?
2410  014352  001401                        BEQ     4$              ;BR IF YES
2411  014354  104003                        ERROR   3               ;ERROR, BITS IS NOT CLEAR
2412  014356  104405                  4$:   SCOP1                   ;SCOPE SUBTEST (SW09=1)
2413
2414
2415                                        ;*************************** TEST 6 ****************************
2416                                        ;*LINE UNIT REGISTER WRITE/READ TEST
2417                                        ;*SET BIT1 IN LU REGISTER 17, VERIFY IT IS SET
2418                                        ;*CLEAR BIT1 IN LU REGISTER 17, VERIFY IT IS CLEAR
2419                                        ;;***********************************************************
2420
2421                                        ;  TEST 6
```

```
                                        ;-------------
                                        ;*************************************************************
                                TST6:   SCOPE
014360  000004
014362  012737  000006  001202          MOV     #6,$TSTNM               ; LOAD THE NO. OF THIS TEST
014370  012737  014520  001442          MOV     #TST7,NEXT              ; POINT TO THE START OF NEXT TEST.
014376  012737  014412  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
014404  104410                          MSTCLR                  ;MASTER CLEAR KMC11
014406  012702  000017                  MOV     #17,R2          ;SAVE REGISTER ADDRESS FOR TYPEOUT
014412  012761  000001  000004  1$:     MOV     #1,4(R1)        ;LOAD PORT4
014420  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
014422  122117                          122117                  ;SET BIT1 IN LU-17
014424  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
014426  021365                          021365                  ;READ LU-17
014430  012705  000001                  MOV     #1,R5           ;PUT "EXPECTED" IN R5
014434  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" IN R4
014440  042704  000376                  BIC     #376,R4         ;CLEAR UNWANTED BITS
014444  120504                          CMPB    R5,R4           ;IS BIT1 SET?
014446  001401                          BEQ     2$              ;BR IF YES
014450  104003                          ERROR   3               ;ERROR, BIT 1 IS NOT SET
014452  104405                  2$:     SCOP1                   ;SCOPE SUBTEST (SW09=1)
014454  012737  014462  001444          MOV     #3$,LOCK        ;NEW SCOP1
014462  005061  000004          3$:     CLR     4(R1)           ;LOAD PORT4
014466  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
014470  122117                          122117                  ;CLEAR BIT 1 IN LU-17
014472  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
014474  021365                          021365                  ;READ LU-17
014476  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
014500  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" IN R4
014504  042704  000376                  BIC     #376,R4         ;CLEAR UNWANTED BITS
014510  120504                          CMPB    R5,R4           ;IS BIT1 CLEAR?
014512  001401                          BEQ     4$              ;BR IF YES
014514  104003                          ERROR   3               ;ERROR, BIT1 IS NOT CLEAR
014516  104405                  4$:     SCOP1                   ;SCOPE SUBTEST (SW09=1)


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


                                        ;  TEST 7
                                        ;-------------
                                        ;*************************************************************
                                TST7:   SCOPE
014520  000004
014522  012737  000007  001202          MOV     #7,$TSTNM               ; LOAD THE NO. OF THIS TEST
014530  012737  014730  001442          MOV     #TST10,NEXT             ; POINT TO THE START OF NEXT TEST.
014536  012737  014556  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
014544  104410                          MSTCLR                  ;MASTER CLEAR KMC11
014546  012702  000013                  MOV     #13,R2          ;SAVE REGISTER ADDRESS FOR TYPEOUT
014552  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
014556                          64$:
014556  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
014562  042761  000257  000004          BIC     #257,4(R1)      ;CLEAR UNWANTED BITS
```

```
2478  014570  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2479  014572  122113                          122100!13               ;MOV DATA TO IBUS REGISTER 13
2480  014574  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2481  014576  021265                          21005!<13*20>           ;READ FROM IBUS REGISTER 13
2482  014600  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
2483  014602  042705  000257                  BIC     #257,R5         ;CLEAR UNWANTED BITS
2484  014606  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
2485  014612  042704  000257                  BIC     #257,R4         ;CLEAR UNWANTED BITS
2486  014616  120504                          CMPB    R5,R4           ;DATA CORRECT?
2487  014620  001401                          BEQ     65$             ;BR IF YES
2488  014622  104003                          ERROR   3               ;ERROR
2489  014624  104405                  65$:    SCOP1                   ;SW09=1?
2490  014626  000241                          CLC                     ;CLEAR CARRY
2491  014630  106100                          ROLB    R0              ;SHIFT BIT IN R0
2492  014632  001351                          BNE     64$             ;IF R0=0 THEN DONE
2493  014634  012737  014650  001444          MOV     #67$,LOCK       ;NEW SCOP1
2494  014642  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
2495  014646  005100                  69$:    COM     R0              ;CHANGE TO FLOATING ZERO
2496  014650                          67$:
2497  014650  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
2498  014654  042761  000257  000004          BIC     #257,4(R1)      ;CLEAR UNWANTED BITS
2499  014662  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2500  014664  122113                          122100!13               ;MOV DATA TO IBUS REGISTER 13
2501  014666  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2502  014670  021265                          21005!<13*20>           ;READ FROM IBUS REGISTER 13
2503  014672  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
2504  014674  042705  000257                  BIC     #257,R5         ;CLEAR UNWANTED BITS
2505  014700  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
2506  014704  042704  000257                  BIC     #257,R4         ;CLEAR UNWANTED BITS
2507  014710  120504                          CMPB    R5,R4           ;DATA CORRECT?
2508  014712  001401                          BEQ     68$             ;BR IF YES
2509  014714  104003                          ERROR   3               ;ERROR
2510  014716  104405                  68$:    SCOP1                   ;SW09=1?
2511  014720  005100                          COM     R0              ;CHANGE TO FLOATING 1
2512  014722  000241                          CLC                     ;CLEAR CARRY
2513  014724  106100                          ROLB    R0              ;SHIFT BIT IN R0
2514  014726  001347                          BNE     69$             ;IF R0=0 THEN DONE
2515
2516
2517                                  ;*************************** TEST 10 ****************************
2518                                  ;*LINE UNIT REGISTER WRITE/READ TEST
2519                                  ;*FLOAT A 1 THROUGH LINE UNIT REGISTER 14
2520                                  ;*FLOAT A 0 THROUGH LINE UNIT REGISTER 14
2521                                  ;;************************************************************************
2522
2523                                  ;  TEST 10
2524                                  ;--------------
2525                                  ;;************************************************************************
2526  014730  000004                  TST10:  SCOPE
2527  014732  012737  000010  001202          MOV     #10,$TSTNM              ; LOAD THE NO. OF THIS TEST
2528  014740  012737  015104  001442          MOV     #TST11,NEXT             ; POINT TO THE START OF NEXT TEST.
2529  014746  012737  014766  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
2530                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
2531  014754  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2532  014756  012702  000014                  MOV     #14,R2          ;SAVE REGISTER ADDRESS FOR TYPEOUT
2533  014762  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
```

```
2534  014766                          64$:
2535  014766  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
2536  014772  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2537  014774  122114                          122100!14               ;MOV DATA TO IBUS REGISTER 14
2538  014776  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2539  015000  021305                          21005!(14*20)           ;READ FROM IBUS REGISTER 14
2540  015002  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
2541  015004  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
2542  015010  120504                          CMPB    R5,R4           ;DATA CORRECT?
2543  015012  001401                          BEQ     65$             ;BR IF YES
2544  015014  104003                          ERROR   3               ;ERROR
2545  015016  104405                  65$:    SCOP1                   ;SW09=1?
2546  015020  000241                          CLC                     ;CLEAR CARRY
2547  015022  106100                          ROLB    R0              ;SHIFT BIT IN R0
2548  015024  001360                          BNE     64$             ;IF R0=0 THEN DONE
2549  015026  012737  015042  001444          MOV     #67$,LOCK       ;NEW SCOP1
2550  015034  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
2551  015040  005100                  69$:    COM     R0              ;CHANGE TO FLOATING ZERO
2552  015042                          67$:
2553  015042  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
2554  015046  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2555  015050  122114                          122100!14               ;MOV DATA TO IBUS REGISTER 14
2556  015052  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2557  015054  021305                          21005!(14*20)           ;READ FROM IBUS REGISTER 14
2558  015056  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
2559  015060  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
2560  015064  120504                          CMPB    R5,R4           ;DATA CORRECT?
2561  015066  001401                          BEQ     68$             ;BR IF YES
2562  015070  104003                          ERROR   3               ;ERROR
2563  015072  104405                  68$:    SCOP1                   ;SW09=1?
2564  015074  005100                          COM     R0              ;CHANGE TO FLOATING 1
2565  015076  000241                          CLC                     ;CLEAR CARRY
2566  015100  106100                          ROLB    R0              ;SHIFT BIT IN R0
2567  015102  001356                          BNE     69$             ;IF R0=0 THEN DONE
2568
2569
2570                                  ;*************************** TEST 11 ****************************
2571                                  ;*SWITCH PAC TEST
2572                                  ;*THIS TEST READS SWITCH PAC#1
2573                                  ;*THIS SWITCH PAC CONTAINS THE DDCMP LINE #
2574                                  ;:************************************************************
2575
2576                                  ;  TEST 11
2577                                  ;--------------
2578                                  ;:************************************************************
2579  015104  000004                  TST11:  SCOPE
2580  015106  012737  000011  001202          MOV     #11,$TSTNM              ; LOAD THE NO. OF THIS TEST
2581  015114  012737  015146  001442          MOV     #TST12,NEXT             ; POINT TO THE START OF NEXT TEST.
2582                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
2583  015122  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2584  015124  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2585  015126  021324                          021324                  ;PORT4←LU15
2586  015130  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
2587  015134  113705  002052                  MOVB    STAT2,R5        ;PUT "EXPECTED" IN R5
2588  015140  120504                          CMPB    R5,R4           ;SW OK?
2589  015142  001401                          BEQ     1$              ;BR IF YES
```

```
                                        ERROR   31              ;ERROR, SWITCH PAC READ ERROR
                                1$:

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

                                        ;  TEST 12
                                        ;--------------
                                ;:************************************************************
                                TST12:  SCOPE
                                        MOV     #12,$TSTNM              ; LOAD THE NO. OF THIS TEST
                                        MOV     #TST13,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
                                        MSTCLR                  ;MASTER CLEAR KMC11
                                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
                                        021344                  ;PORT4←LU16
                                        MOV     4(R1),R4        ;PUT "FOUND" IN R4
                                        MOVB    STAT2+1,R5      ;PUT "EXPECTED" IN R5
                                        CMPB    R5,R4           ;SW OK?
                                        BEQ     1$              ;BR IF YES
                                        ERROR   31              ;ERROR, SWITCH PAC READ ERROR
                                1$:

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

                                        ;  TEST 13
                                        ;--------------
                                ;:************************************************************
                                TST13:  SCOPE
                                        MOV     #13,$TSTNM              ; LOAD THE NO. OF THIS TEST
                                        MOV     #TST14,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
                                        MSTCLR                  ;MASTER CLEAR KMC11
                                        CLR     TEMP            ;PREPARE FOR DELAY
                                1$:
                                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
                                        021364                  ;PORT4←LU-17
                                        BIT     #2,4(R1)        ;IS CLOCK BIT SET?
                                        BNE     2$              ;BR IF YES
                                        INC     TEMP            ;DELAY
                                        BNE     1$      ;DELAY FINISHED?
                                        ERROR   4               ;ERROR BIT IS STUCK CLEAR
                                2$:     CLR     TEMP            ;PREPARE FOR DELAY
                                3$:
                                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
                                        021364                  ;PORT4←LU-17
                                        BIT     #2,4(R1)        ;IS CLOCK BIT CLEAR?
```

```
2646  015276  001404                        BEQ     4$              ;BR IF YES
2647  015300  005237  011106                INC     TEMP            ;DELAY
2648  015304  001367                        BNE     3$              ;BR IF DELAY NOT DONE
2649  015306  104004                        ERROR   4               ;ERROR BIT IS STUCK SET
2650  015310                          4$:
2651
2652
2653                                        ;*************************** TEST 14 ****************************
2654                                        ;*OUT DATA SILO TEST
2655                                        ;*SET SOM AND LOAD OUT DATA SILO
2656                                        ;*VERIFY THAT OCOR SET, INDICATING THAT THE
2657                                        ;*CHARACTER IS AT THE BOTTOM OF THE OUT SILO
2658                                        ;:************************************************************
2659
2660                                        ;  TEST 14
2661                                        ;-------------
2662                                    ;;**************************************************************
2663  015310  000004                  TST14:  SCOPE
2664  015312  012737  000014  001202        MOV     #14,$TSTNM              ; LOAD THE NO. OF THIS TEST
2665  015320  012737  015424  001442        MOV     #TST15,NEXT             ; POINT TO THE START OF NEXT TEST.
2666                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
2667  015326  104410                        MSTCLR                  ;MASTER CLEAR KMC11
2668  015330  005061  000004                CLR     4(R1)           ;CLEAR PORT4
2669  015334  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2670  015336  122117                        122117                  ;PUT LINE UNIT IN BITSTUFF MODE
2671  015340  004737  035032                JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
2672  015344  012711  004000                MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
2673  015350  012761  000001  000004        MOV     #1,4(R1)        ;LOAD PORT4 WITH BIT0
2674  015356  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2675  015360  122111                        122111                  ;SET SOM
2676  015362  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2677  015364  122110                        122110                  ;LOAD OUT DATA SILO
2678  015366  104414  000002                TIMER,  2               ;WAIT FOR OCOR
2679  015372  012702  000017                MOV     #17,R2          ;SAVE ADDRESS FOR TYPEOUT
2680  015376  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2681  015400  021364                        021364                  ;PORT4=LU 17
2682  015402  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
2683  015406  042704  000357                BIC     #357,R4         ;CLEAR UNWANTED BITS
2684  015412  012705  000020                MOV     #20,R5          ;PUT "EXPECTED" IN R5
2685  015416  120504                        CMPB    R5,R4           ;IS OCOR SET?
2686  015420  001401                        BEQ     1$              ;BR IF YES
2687  015422  104005                        ERROR   5
2688  015424                          1$:
2689
2690
2691                                        ;*************************** TEST 15 ****************************
2692                                        ;*BITSTUFF TEST OF RTS AND OUT ACTIVE
2693                                        ;*SET SOM AND LOAD OUT DATA SILO
2694                                        ;*SINGLE STEP 2 DATA CLOCKS, VERIFY
2695                                        ;*THAT RTS AND ACTIVE ARE SET
2696                                        ;:************************************************************
2697
2698                                        ;  TEST 15
2699                                        ;-------------
2700                                    ;;**************************************************************
2701  015424  000004                  TST15:  SCOPE
```

```
2702  015426  012737  000015  001202          MOV     #15,$TSTNM              ; LOAD THE NO. OF THIS TEST
2703  015434  012737  015576  001442          MOV     #TST16,NEXT             ; POINT TO THE START OF NEXT TEST.
2704                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
2705  015442  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2706  015444  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
2707  015450  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2708  015452  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
2709  015454  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
2710  015460  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
2711  015464  012761  000001  000004          MOV     #1,4(R1)        ;LOAD PORT4 WITH BIT0
2712  015472  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2713  015474  122111                          122111                  ;SET SOM
2714  015476  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2715  015500  122110                          122110                  ;LOAD OUT DATA SILO
2716  015502  004737  033502                  JSR     PC,OCOR         ;WAIT FOR OCOR
2717  015506  104413  000002                  DATACLK,        2       ;CLOCK DATA FOUR TIMES
2718  015512  012702  000011                  MOV     #11,R2          ;SAVE ADDRESS FOR TYPEOUT
2719  015516  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2720  015520  021224                          021224                  ;PORT4=LU 11
2721  015522  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
2722  015526  042704  000257                  BIC     #257,R4         ;CLEAR UNWANTED BITS
2723  015532  012705  000120                  MOV     #120,R5         ;PUT "EXPECTED" IN R5
2724  015536  120504                          CMPB    R5,R4           ;IS ACTIVE SET?
2725  015540  001401                          BEQ     1$              ;BR IF YES
2726  015542  104005                          ERROR   5
2727  015544                          1$:
2728  015544  012702  000013                  MOV     #13,R2          ;SAVE ADDRESS FOR TYPEOUT
2729  015550  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2730  015552  021264                          021264                  ;PORT4=LU 13
2731  015554  016104  000004                  MOV     4(R1),R4        ;PUT EXPECTED IN R4
2732  015560  042704  000337                  BIC     #337,R4         ;CLEAR UNWANTED BITS
2733  015564  012705  000040                  MOV     #BIT5,R5        ;PUT "EXPECTED" IN R5, RTS SHOULD BE SET
2734  015570  120504                          CMPB    R5,R4           ;IS RTS OK?
2735  015572  001401                          BEQ     2$              ;BR IF YES
2736  015574  104005                          ERROR   5               ;RTS ERROR
2737  015576                          2$:
2738
2739
2740                                          ;*************************** TEST 16 ****************************
2741                                          ;*TEST OF OUT CLEAR
2742                                          ;*SET SOM AND LOAD OUT DATA SILO
2743                                          ;*SINGLE STEP DATA CLOCK, SET OUT CLEAR
2744                                          ;*VERIFY THAT OCOR,RTS, AND ACTIVE ARE CLEARED
2745                                          ;;*************************************************************
2746
2747                                          ;  TEST 16
2748                                          ;--------------
2749                                  ;;*****************************************************************
2750  015576  000004                  TST16:  SCOPE
2751  015600  012737  000016  001202          MOV     #16,$TSTNM              ; LOAD THE NO. OF THIS TEST
2752  015606  012737  016010  001442          MOV     #TST17,NEXT             ; POINT TO THE START OF NEXT TEST.
2753                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
2754  015614  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2755  015616  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
2756  015622  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2757  015624  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
```

```
2758  015626  004737  035032                JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
2759  015632  012711  004000                MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
2760  015636  012761  000001  000004        MOV     #1,4(R1)        ;LOAD PORT4 WITH BIT0
2761  015644  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2762  015646  122111                        122111                  ;SET SOM
2763  015650  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2764  015652  122110                        122110                  ;LOAD OUT DATA SILO
2765  015654  004737  033502                JSR     PC,OCOR         ;WAIT FOR OCOR
2766  015660  104413  000002                DATACLK,        2       ;CLOCK DATA FOUR TIMES
2767  015664  012761  000200  000004        MOV     #BIT7,4(R1)     ;SET BIT7 IN PORT4
2768  015672  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2769  015674  122111                        122111                  ;SET OUT CLEAR
2770  015676  104413  000001                DATACLK,        1       ;GIVE A TICK TO CLEAR RTS
2771  015702  012702  000017                MOV     #17,R2          ;SAVE ADDRESS FOR TYPEOUT
2772  015706  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2773  015710  021364                        021364                  ;PORT4=LU 17
2774  015712  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
2775  015716  042704  000357                BIC     #357,R4         ;CLEAR UNWANTED BITS
2776  015722  005005                        CLR     R5              ;PUT "EXPECTED" IN R5
2777  015724  120504                        CMPB    R5,R4           ;IS OCOR CLEARED?
2778  015726  001401                        BEQ     1$              ;BR IF YES
2779  015730  104005                        ERROR   5
2780  015732                        1$:
2781  015732  012702  000013                MOV     #13,R2          ;SAVE ADDRESS FOR TYPEOUT
2782  015736  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2783  015740  021264                        021264                  ;PORT4=LU 13
2784  015742  016104  000004                MOV     4(R1),R4        ;PUT EXPECTED IN R4
2785  015746  042704  000337                BIC     #337,R4         ;CLEAR UNWANTED BITS
2786  015752  005005                        CLR     R5              ;PUT "EXPECTED" IN R5, RTS SHOULD BE CLEARED
2787  015754  120504                        CMPB    R5,R4           ;IS RTS OK?
2788  015756  001401                        BEQ     2$              ;BR IF YES
2789  015760  104005                        ERROR   5               ;RTS ERROR
2790  015762                        2$:
2791  015762  012702  000011                MOV     #11,R2          ;SAVE ADDRESS FOR TYPEOUT
2792  015766  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2793  015770  021224                        021224                  ;PORT4=LU11
2794  015772  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
2795  015776  012705  000020                MOV     #BIT4,R5        ;ONLY OUT READY SHOULD BE SET
2796  016002  120504                        CMPB    R5,R4           ;IS ACTIVE CLEAR?
2797  016004  001401                        BEQ     3$              ;BR IF YES
2798  016006  104005                        ERROR   5               ;ERROR ACTIVE NOT CLEARED
2799  016010                        3$:
2800
2801
2802                                ;*************************** TEST 17 ****************************
2803                                ;*BITSTUFF TRANSMITTER TEST
2804                                ;*SINGLE CLOCK THE CHARACTER 0
2805                                ;*CHECK FLAG AND DATA IN THE BIT WINDOW
2806                                ;*VERIFY EACH BIT POSITION AS IT
2807                                ;*PASSES THE BIT WINDOW (SI BIT)
2808                                ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2809                                ;*****************************************************************
2810
2811                                ;  TEST 17
2812                                ;-------------
2813                          ;;*****************************************************************
```

```
2814  016010  000004                    TST17:  SCOPE
2815  016012  012737  000017  001202            MOV     #17,$TSTNM              ; LOAD THE NO. OF THIS TEST
2816  016020  012737  016272  001442            MOV     #TST20,NEXT             ; POINT TO THE START OF NEXT TEST.
2817                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
2818  016026  104410                            MSTCLR                  ;MASTER CLEAR KMC11
2819  016030  005061  000004                    CLR     4(R1)           ;CLEAR PORT4
2820  016034  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2821  016036  122117                            122117                  ;PUT LINE UNIT IN BITSTUFF MODE
2822  016040  004737  035032                    JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
2823  016044  005037  035250                    CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
2824  016050  012711  004000                    MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
2825  016054  004737  033634                    JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
2826  016060  012761  000001  000004            MOV     #1,4(R1)        ;SET BIT0 IN PORT4
2827  016066  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2828  016070  122111                            122111                  ;SET SOM!
2829  016072  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2830  016074  122110                            122110                  ;LOAD GARBAGE CHAR
2831  016076  012705  000000                    MOV     #0,R5   ;LOAD CHARACTER IN R5 FOR TYPEOUT
2832  016102  004737  033634                    JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
2833  016106  010561  000004                    MOV     R5,4(R1)        ;LOAD PORT4 WITH CHARACTER
2834  016112  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2835  016114  122110                            122110                  ;LOAD OUT DATA
2836  016116  004737  033502                    JSR     PC,OCOR         ;WAIT FOR OCOR TO SET
2837  016122  005003                            CLR     R3              ;CLEAR BIT COUNTER
2838  016124  010502                            MOV     R5,R2           ;LOAD CHARACTER IN R2
2839  016126  104413  000002                    DATACLK,        2       ;2 TICKS TO SET UP TRANSMITTER
2840  016132  012737  000176  001302            MOV     #^B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
2841  016140  104413  000001            64$:    DATACLK,        1       ;CLOCK FLAG ONCE
2842  016144  106037  001302                    RORB    $TMP2           ;SHIFT SOFT FLAG
2843  016150  103405                            BCS     65$             ;BR IF BIT IS MARK
2844  016152  004737  033450                    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
2845  016156  103006                            BCC     66$             ;BR IF OK
2846  016160  104026                            ERROR   26              ;ERROR IN FLAG CHAR
2847  016162  000404                            BR      66$
2848  016164  004737  033450            65$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
2849  016170  103401                            BCS     66$             ;BR IF OK
2850  016172  104026                            ERROR   26              ;ERROR IN FLAG CHAR
2851  016174  005203                    66$:    INC     R3              ;INC BIT COUNT
2852  016176  022703  000010                    CMP     #10,R3          ;FLAG DONE YET?
2853  016202  001356                            BNE     64$             ;BR IF NO
2854  016204  005003                            CLR     R3              ;CLEAR BIT COUNT
2855  016206  104413  000001            1$:     DATACLK,        1       ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2856  016212  106002                            RORB    R2              ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2857  016214  103005                            BCC     2$              ;BR IF CARRY CLEAR
2858  016216  004737  033450                    JSR     PC,GETSI        ;GET THE WINDOW
2859  016222  103406                            BCS     3$              ;BR IF BIT IS A MARK
2860  016224  104006                            ERROR   6               ;ERROR BIT WAS A SPACE
2861  016226  000404                            BR      3$              ;CONTINE WITH TEST
2862  016230  004737  033450            2$:     JSR     PC,GETSI        ;GET THE WINDOW
2863  016234  103001                            BCC     3$              ;BR IF BIT IS A SPACE
2864  016236  104006                            ERROR   6               ;ERROR BIT WAS A MARK
2865  016240                            3$:
2866  016240  005203                            INC     R3              ;NEXT BIT
2867  016242  022703  000010                    CMP     #10,R3          ;DONE YET?
2868  016246  001357                            BNE     1$              ;BR IF NO
2869  016250  104413  000014                    DATACLK,        14      ;CLOCK TRANSMITTER 14 MORE TICKS
```

```
2870  016254  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2871  016256  021264                        021264                  ;PORT4=LU-13
2872  016260  032761  000040  000004        BIT     #BIT5,4(R1)     ;RTS SHOULD BE CLEAR NOW
2873  016266  001401                        BEQ     4$              ;BR IF YES
2874  016270  104034                        ERROR   34              ;ERROR, RTS NOT CLEAR
2875  016272                        4$:
2876
2877
2878                                        ;************************** TEST 20 **************************
2879                                        ;*BITSTUFF TRANSMITTER TEST
2880                                        ;*SINGLE CLOCK THE CHARACTER 125
2881                                        ;*CHECK FLAG AND DATA IN THE BIT WINDOW
2882                                        ;*VERIFY EACH BIT POSITION AS IT
2883                                        ;*PASSES THE BIT WINDOW (SI BIT)
2884                                        ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2885                                        ;;*********************************************************************
2886
2887                                        ;  TEST 20
2888                                        ;-------------
2889                                    ;;**************************************************************
2890  016272  000004                TST20:  SCOPE
2891  016274  012737  000020  001202        MOV     #20,$TSTNM              ; LOAD THE NO. OF THIS TEST
2892  016302  012737  016554  001442        MOV     #TST21,NEXT             ; POINT TO THE START OF NEXT TEST.
2893                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
2894  016310  104410                        MSTCLR                  ;MASTER CLEAR KMC11
2895  016312  005061  000004                CLR     4(R1)           ;CLEAR PORT4
2896  016316  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2897  016320  122117                        122117                  ;PUT LINE UNIT IN BITSTUFF MODE
2898  016322  004737  035032                JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
2899  016326  005037  035250                CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
2900  016332  012711  004000                MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
2901  016336  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
2902  016342  012761  000001  000004        MOV     #1,4(R1)        ;SET BIT0 IN PORT4
2903  016350  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2904  016352  122111                        122111                  ;SET SOM!
2905  016354  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2906  016356  122110                        122110                  ;LOAD GARBAGE CHAR
2907  016360  012705  000125                MOV     #125,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
2908  016364  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
2909  016370  010561  000004                MOV     R5,4(R1)        ;LOAD PORT4 WITH CHARACTER
2910  016374  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2911  016376  122110                        122110                  ;LOAD OUT DATA
2912  016400  004737  033502                JSR     PC,OCOR         ;WAIT FOR OCOR TO SET
2913  016404  005003                        CLR     R3              ;CLEAR BIT COUNTER
2914  016406  010502                        MOV     R5,R2           ;LOAD CHARACTER IN R2
2915  016410  104413  000002                DATACLK,        2       ;2 TICKS TO SET UP TRANSMITTER
2916  016414  012737  000176  001302        MOV     #^B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
2917  016422  104413  000001        64$:    DATACLK,        1       ;CLOCK FLAG ONCE
2918  016426  106037  001302                RORB    $TMP2           ;SHIFT SOFT FLAG
2919  016432  103405                        BCS     65$             ;BR IF BIT IS MARK
2920  016434  004737  033450                JSR     PC,GETSI        ;LOOK AT BIT WINDOW
2921  016440  103006                        BCC     66$             ;BR IF OK
2922  016442  104026                        ERROR   26              ;ERROR IN FLAG CHAR
2923  016444  000404                        BR      66$
2924  016446  004737  033450        65$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
2925  016452  103401                        BCS     66$             ;BR IF OK
```

```
2926  016454  104026                        ERROR   26              ;ERROR IN FLAG CHAR
2927  016456  005203                66$:    INC     R3              ;INC BIT COUNT
2928  016460  022703  000010                CMP     #10,R3          ;FLAG DONE YET?
2929  016464  001356                        BNE     64$             ;BR IF NO
2930  016466  005003                        CLR     R3              ;CLEAR BIT COUNT
2931  016470  104413  000001          1$:   DATACLK,        1       ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2932  016474  106002                        RORB    R2              ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
2933  016476  103005                        BCC     2$              ;BR IF CARRY CLEAR
2934  016500  004737  033450                JSR     PC,GETSI        ;GET THE WINDOW
2935  016504  103406                        BCS     3$              ;BR IF BIT IS A MARK
2936  016506  104006                        ERROR   6               ;ERROR BIT WAS A SPACE
2937  016510  000404                        BR      3$              ;CONTINE WITH TEST
2938  016512  004737  033450          2$:   JSR     PC,GETSI        ;GET THE WINDOW
2939  016516  103001                        BCC     3$              ;BR IF BIT IS A SPACE
2940  016520  104006                        ERROR   6               ;ERROR BIT WAS A MARK
2941  016522                        3$:
2942  016522  005203                        INC     R3              ;NEXT BIT
2943  016524  022703  000010                CMP     #10,R3          ;DONE YET?
2944  016530  001357                        BNE     1$              ;BR IF NO
2945  016532  104413  000014                DATACLK,        14      ;CLOCK TRANSMITTER 14 MORE TICKS
2946  016536  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2947  016540  021264                        021264                  ;PORT4=LU-13
2948  016542  032761  000040  000004       BIT     #BIT5,4(R1)     ;RTS SHOULD BE CLEAR NOW
2949  016550  001401                        BEQ     4$              ;BR IF YES
2950  016552  104034                        ERROR   34              ;ERROR, RTS NOT CLEAR
2951  016554                        4$:
2952
2953
2954                                        ;*************************** TEST 21 ****************************
2955                                        ;*BITSTUFF TRANSMITTER TEST
2956                                        ;*SINGLE CLOCK THE CHARACTER 252
2957                                        ;*CHECK FLAG AND DATA IN THE BIT WINDOW
2958                                        ;*VERIFY EACH BIT POSITION AS IT
2959                                        ;*PASSES THE BIT WINDOW (SI BIT)
2960                                        ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2961                                        ;;************************************************************
2962
2963                                        ;  TEST 21
2964                                        ;--------------
2965                                  ;;************************************************************
2966  016554  000004                TST21:  SCOPE
2967  016556  012737  000021  001202        MOV     #21,$TSTNM              ; LOAD THE NO. OF THIS TEST
2968  016564  012737  017036  001442        MOV     #TST22,NEXT             ; POINT TO THE START OF NEXT TEST.
2969                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
2970  016572  104410                        MSTCLR                  ;MASTER CLEAR KMC11
2971  016574  005061  000004                CLR     4(R1)           ;CLEAR PORT4
2972  016600  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2973  016602  122117                        122117                  ;PUT LINE UNIT IN BITSTUFF MODE
2974  016604  004737  035032                JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
2975  016610  005037  035250                CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
2976  016614  012711  004000                MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
2977  016620  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
2978  016624  012761  000001  000004       MOV     #1,4(R1)        ;SET BIT0 IN PORT4
2979  016632  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2980  016634  122111                        122111                  ;SET SOM!
2981  016636  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
2982  016640  122110                          122110                  ;LOAD GARBAGE CHAR
2983  016642  012705  000252                  MOV     #252,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
2984  016646  004737  033634                  JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
2985  016652  010561  000004                  MOV     R5,4(R1)        ;LOAD PORT4 WITH CHARACTER
2986  016656  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2987  016660  122110                          122110                  ;LOAD OUT DATA
2988  016662  004737  033502                  JSR     PC,OCOR         ;WAIT FOR OCOR TO SET
2989  016666  005003                          CLR     R3              ;CLEAR BIT COUNTER
2990  016670  010502                          MOV     R5,R2           ;LOAD CHARACTER IN R2
2991  016672  104413  000002                  DATACLK,        2       ;2 TICKS TO SET UP TRANSMITTER
2992  016676  012737  000176  001302          MOV     #^B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
2993  016704  104413  000001          64$:    DATACLK,        1       ;CLOCK FLAG ONCE
2994  016710  106037  001302                  RORB    $TMP2           ;SHIFT SOFT FLAG
2995  016714  103405                          BCS     65$             ;BR IF BIT IS MARK
2996  016716  004737  033450                  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
2997  016722  103006                          BCC     66$             ;BR IF OK
2998  016724  104026                          ERROR   26              ;ERROR IN FLAG CHAR
2999  016726  000404                          BR      66$
3000  016730  004737  033450          65$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
3001  016734  103401                          BCS     66$             ;BR IF OK
3002  016736  104026                          ERROR   26              ;ERROR IN FLAG CHAR
3003  016740  005203                  66$:    INC     R3              ;INC BIT COUNT
3004  016742  022703  000010                  CMP     #10,R3          ;FLAG DONE YET?
3005  016746  001356                          BNE     64$             ;BR IF NO
3006  016750  005003                          CLR     R3              ;CLEAR BIT COUNT
3007  016752  104413  000001          1$:     DATACLK,        1       ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
3008  016756  106002                          RORB    R2              ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
3009  016760  103005                          BCC     2$              ;BR IF CARRY CLEAR
3010  016762  004737  033450                  JSR     PC,GETSI        ;GET THE WINDOW
3011  016766  103406                          BCS     3$              ;BR IF BIT IS A MARK
3012  016770  104006                          ERROR   6               ;ERROR BIT WAS A SPACE
3013  016772  000404                          BR      3$              ;CONTINE WITH TEST
3014  016774  004737  033450          2$:     JSR     PC,GETSI        ;GET THE WINDOW
3015  017000  103001                          BCC     3$              ;BR IF BIT IS A SPACE
3016  017002  104006                          ERROR   6               ;ERROR BIT WAS A MARK
3017  017004                          3$:
3018  017004  005203                          INC     R3              ;NEXT BIT
3019  017006  022703  000010                  CMP     #10,R3          ;DONE YET?
3020  017012  001357                          BNE     1$              ;BR IF NO
3021  017014  104413  000014                  DATACLK,        14      ;CLOCK TRANSMITTER 14 MORE TICKS
3022  017020  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3023  017022  021264                          021264                  ;PORT4=LU-13
3024  017024  032761  000040  000004          BIT     #BIT5,4(R1)     ;RTS SHOULD BE CLEAR NOW
3025  017032  001401                          BEQ     4$              ;BR IF YES
3026  017034  104034                          ERROR   34              ;ERROR, RTS NOT CLEAR
3027  017036                          4$:
3028
3029
3030                                  ;************************** TEST 22 **************************
3031                                  ;*BIT STUFF TEST
3032                                  ;*THIS TEST CHECKS ZERO BIT STUFFING OF
3033                                  ;* THE TRANSMITTER IN THE BIT WINDOW
3034                                  ;:*************************************************************
3035
3036                                  ;  TEST 22
3037                                  ;-------------
```

```
3038                                    ;;*****************************************************************
3039  017036  000004                    TST22:  SCOPE
3040  017040  012737  000022  001202            MOV     #22,$TSTNM              ; LOAD THE NO. OF THIS TEST
3041  017046  012737  017346  001442            MOV     #TST23,NEXT             ; POINT TO THE START OF NEXT TEST.
3042                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
3043  017054  104410                            MSTCLR                  ;MASTER CLEAR KMC11
3044  017056  005061  000004                    CLR     4(R1)           ;CLEAR PORT4
3045  017062  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3046  017064  122117                            122117                  ;PUT LINE UNIT IN BITSTUFF MODE
3047  017066  004737  035032                    JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
3048  017072  012711  004000                    MOV     #BIT11,(R1)     ;SET LU LOOP
3049  017076  004737  033634                    JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
3050  017102  012761  000001  000004            MOV     #1,4(R1)        ;SET BIT0 IN PORT4
3051  017110  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3052  017112  122111                            122111                  ;SET SOM!
3053  017114  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3054  017116  122110                            122110                  ;LOAD GARBAGE CHAR
3055  017120  004537  034770                    JSR     R5,MESLD        ;LOAD OUT SILO DATA
3056  017124  035276                            STUFDT          ;MESSAGE ADDRESSS
3057  017126  000024                            20.                     ;NUMBER OF CHARACTERS
3058  017130  012704  035276                    MOV     #STUFDT,R4      ;R4=CHARACTER POINTER
3059  017134  005003                            CLR     R3              ;R3= BIT COUNTER
3060  017136  012700  000006                    MOV     #6,R0           ;BIT COUNTER FOR FLAG CHARACTER
3061  017142  104413  000002                    DATACLK,        2       ;SET UP TRANSMITTER
3062  017146  012737  000176  001302            MOV     #^B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
3063  017154  104413  000001          64$:      DATACLK,        1       ;CLOCK FLAG ONCE
3064  017160  106037  001302                    RORB    $TMP2           ;SHIFT SOFT FLAG
3065  017164  103405                            BCS     65$             ;BR IF BIT IS MARK
3066  017166  004737  033450                    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
3067  017172  103006                            BCC     66$             ;BR IF OK
3068  017174  104026                            ERROR   26              ;ERROR IN FLAG CHAR
3069  017176  000404                            BR      66$
3070  017200  004737  033450          65$:      JSR     PC,GETSI        ;LOOK AT BIT WINDOW
3071  017204  103401                            BCS     66$             ;BR IF OK
3072  017206  104026                            ERROR   26              ;ERROR IN FLAG CHAR
3073  017210  005203                    66$:    INC     R3              ;INC BIT COUNT
3074  017212  022703  000010                    CMP     #10,R3          ;FLAG DONE YET?
3075  017216  001356                            BNE     64$             ;BR IF NO
3076  017220  005003                            CLR     R3              ;CLEAR BIT COUNT
3077  017222  012700  000024                    MOV     #20.,R0         ;R0=CHARACTER COUNTER
3078  017226  005037  035250                    CLR     BITCON          ;CLEAR BIT STUFF COUNTER
3079  017232  112405                    3$:     MOVB    (R4)+,R5        ;LOAD CHARACTER IN R5
3080  017234  010502                            MOV     R5,R2           ;LOAD CHARACTER IN R2
3081  017236  104413  000001          4$:       DATACLK,        1       ;SHIFT DTAT ONCE
3082  017242  106002                            RORB    R2              ;SHIFT SOFT DATA
3083  017244  103407                            BCS     5$              ;BR IF CARRY SET
3084  017246  005037  035250                    CLR     BITCON          ;CLEAR BIT STUFF COUNTER
3085  017252  004737  033450                    JSR     PC,GETSI        ;LOOK AT WINDOW
3086  017256  103010                            BCC     6$              ;BR IF SPACE
3087  017260  104006                            ERROR   6               ;ERROR, WINDOW WAS A MARK
3088  017262  000406                            BR      6$              ;CONTINUE
3089  017264  005237  035250          5$:       INC     BITCON          ;ADD 1 TO BIT STUFF COUNTER
3090  017270  004737  033450                    JSR     PC,GETSI        ;LOOK AT WINDOW
3091  017274  103401                            BCS     6$              ;BR IF MARK
3092  017276  104006                            ERROR   6               ;ERROR, WINDOW WAS A SPACE
3093  017300  022737  000005  035250    6$:     CMP     #5,BITCON       ;HAVE THERE BEEN 5 1'S IN A ROW
```

```
3094  017306  001010                        BNE     7$              ;BR IF NO
3095  017310  005037  035250                CLR     BITCON          ;IF YES CLR BIT STUFF COUNTER
3096  017314  104413  000001                DATACLK,        1       ;AND CLOCK TRANSMITTER ONCE
3097  017320  004737  033450                JSR     PC,GETSI        ;CHECK WINDOW FOR A ZEOR STUFF!!
3098  017324  103001                        BCC     7$              ;BR IF WINDOW IS A SPACE
3099  017326  104030                        ERROR   30              ;ERROR, TRANSMITTER DID NOT STUFF A ZERO
3100  017330  005203              7$:       INC     R3              ;BUMP BIT COUNTER
3101  017332  022703  000010                CMP     #10,R3          ;DONE THIS CHARACTER YET?
3102  017336  001337                        BNE     4$              ;BR IF NO
3103  017340  005003                        CLR     R3              ;RESTART BIT COUNTER AT ZERO
3104  017342  005300                        DEC     R0              ;DEC CHARACTER COUNTER
3105  017344  001332                        BNE     3$              ;BR IF NOT DONE YET
3106  017346                        8$:
3107
3108
3109                                        ;************************** TEST 23 **************************
3110                                        ;*BITSTUFF TRANSMITTER TEST
3111                                        ;*SINGLE CLOCK THE CHARACTER 377
3112                                        ;*CHECK FLAG AND DATA IN THE BIT WINDOW
3113                                        ;*VERIFY EACH BIT POSITION AS IT
3114                                        ;*PASSES THE BIT WINDOW (SI BIT)
3115                                        ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
3116                                        ;;*********************************************************
3117
3118                                        ;  TEST 23
3119                                        ;-------------
3120                                        ;;*********************************************************
3121  017346  000004              ↑ST23:    SCOPE
3122  017350  012737  000023  001202        MOV     #23,$TSTNM              ; LOAD THE NO. OF THIS TEST
3123  017356  012737  017654  001442        MOV     #TST24,NEXT             ; POINT TO THE START OF NEXT TEST.
3124                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
3125  017364  104410                        MSTCLR                  ;MASTER CLEAR KMC11
3126  017366  005061  000004                CLR     4(R1)           ;CLEAR PORT4
3127  017372  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3128  017374  122117                        122117                  ;PUT LINE UNIT IN BITSTUFF MODE
3129  017376  004737  035032                JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
3130  017402  005037  035250                CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
3131  017406  012711  004000                MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
3132  017412  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
3133  017416  012761  000001  000004        MOV     #1,4(R1)        ;SET BIT0 IN PORT4
3134  017424  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3135  017426  122111                        122111                  ;SET SOM!
3136  017430  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3137  017432  122110                        122110                  ;LOAD GARBAGE CHAR
3138  017434  012705  000377                MOV     #377,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
3139  017440  010537  017612                MOV     R5,5$           ;LOAD CHAR FOR STUFF CHECK
3140  017444  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
3141  017450  010561  000004                MOV     R5,4(R1)        ;LOAD PORT4 WITH CHARACTER
3142  017454  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3143  017456  122110                        122110                  ;LOAD OUT DATA
3144  017460  004737  033502                JSR     PC,OCOR         ;WAIT FOR OCOR TO SET
3145  017464  005003                        CLR     R3              ;CLEAR BIT COUNTER
3146  017466  010502                        MOV     R5,R2           ;LOAD CHARACTER IN R2
3147  017470  104413  000002                DATACLK,        2       ;2 TICKS TO SET UP TRANSMITTER
3148  017474  012737  000176  001302        MOV     #↑B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
3149  017502  104413  000001        64$:    DATACLK,        1       ;CLOCK FLAG ONCE
```

```
3150  017506  106037  001302          RORB    $TMP2           ;SHIFT SOFT FLAG
3151  017512  103405                  BCS     65$             ;BR IF BIT IS MARK
3152  017514  004737  033450          JSR     PC,GETSI        ;LOOK AT BIT WINDOW
3153  017520  103006                  BCC     66$             ;BR IF OK
3154  017522  104026                  ERROR   26              ;ERROR IN FLAG CHAR
3155  017524  000404                  BR      66$
3156  017526  004737  033450  65$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
3157  017532  103401                  BCS     66$             ;BR IF OK
3158  017534  104026                  ERROR   26              ;ERROR IN FLAG CHAR
3159  017536  005203          66$:    INC     R3              ;INC BIT COUNT
3160  017540  022703  000010          CMP     #10,R3          ;FLAG DONE YET?
3161  017544  001356                  BNE     64$             ;BR IF NO
3162  017546  005003                  CLR     R3              ;CLEAR BIT COUNT
3163  017550  005037  035250          CLR     BITCON          ;CLEAR STUFF COUNT
3164  017554  104413  000001  1$:     DATACLK,        1       ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
3165  017560  106002                  RORB    R2              ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
3166  017562  103005                  BCC     2$              ;BR IF CARRY CLEAR
3167  017564  004737  033450          JSR     PC,GETSI        ;GET THE WINDOW
3169  017570  103406                  BCS     3$              ;BR IF BIT IS A MARK
3169  017572  104006                  ERROR   6               ;ERROR BIT WAS A SPACE
3170  017574  000404                  BR      3$              ;CONTINE WITH TEST
3171  017576  004737  033450  2$:     JSR     PC,GETSI        ;GET THE WINDOW
3172  017602  103001                  BCC     3$              ;BR IF BIT IS A SPACE
3173  017604  104006                  ERROR   6               ;ERROR BIT WAS A MARK
3174  017606                  3$:
3175  017606  004537  035132          JSR     R5,STFFCK       ;CHECK FOR BIT STUFF
3176  017612  000377          5$:     377                     ;DATA CHARACTER
3177  017614  000001                  1                       ;SHIFT COUNT
3178  017616  010237  017612          MOV     R2,5$           ;LOAD CHAR FOR STUFF CHECK
3179  017622  005203                  INC     R3              ;NEXT BIT
3180  017624  022703  000010          CMP     #10,R3          ;DONE YET?
3181  017630  001351                  BNE     1$              ;BR IF NO
3182  017632  104413  000014          DATACLK,        14      ;CLOCK TRANSMITTER 14 MORE TICKS
3183  017636  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3184  017640  021264                  021264                  ;PORT4←LU-13
3185  017642  032761  000040  000004  BIT     #BIT5,4(R1)     ;RTS SHOULD BE CLEAR NOW
3186  017650  001401                  BEQ     4$              ;BR IF YES
3187  017652  104034                  ERROR   34              ;ERROR, RTS NOT CLEAR
3188  017654                  4$:
3189
3190
3191                                  ;************************** TEST 24 ****************************
3192                                  ;*BITSTUFF TRANSMITTER TEST
3193                                  ;*SINGLE CLOCK A BINARY COUNT PATTERN
3194                                  ;*VERIFY EACH BIT POSITION AS IT
3195                                  ;*PASSES THE BIT WINDOW (SI BIT)
3196                                  ;*ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
3197                                  ;*AND R5 CONTAINS THE CHARACTER THAT FAILED
3198                                  ;;*************************************************************
3199
3200                                  ;  TEST 24
3201                                  ;--------------
3202                                  ;;*************************************************************
3203  017654  000004          TST24:  SCOPE
3204  017656  012737  000024  001202          MOV     #24,$TSTNM              ; LOAD THE NO. OF THIS TEST
3205  017664  012737  020206  001442          MOV     #TST25,NEXT             ; POINT TO THE START OF NEXT TEST.
```

```
                                                        ;R1 CONTAINS BASE KMC11 ADDRESS
017672  104410                  MSTCLR                  ;MASTER CLEAR KMC11
017674  005061  000004          CLR     4(R1)           ;CLEAR PORT4
017700  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017702  122117                  122117                  ;PUT LINE UNIT IN BITSTUFF MODE
017704  004737  035032          JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
017710  005037  035250          CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
017714  012711  004000          MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
017720  005003                  CLR     R3              ;R3 CONTAINS BIT COUNT
017722  005004                  CLR     R4              ;R4 CONTAINS CHAR TO BE LOADED IN SILO
017724  005005                  CLR     R5              ;R5 CONTAINS CHARACTER CURRENTLY BEING SHIFTED OUT
017726  004737  033634          JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
017732  012761  000001  000004  MOV     #1,4(R1)        ;SET BIT0 IN PORT4
017740  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017742  122111                  122111                  ;SET SOM!
017744  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017746  122110                  122110                  ;LOAD GARBAGE CHAR
017750  004737  033634          JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
017754  010461  000004          MOV     R4,4(R1)        ;LOAD PORT4 WITH CHARACTER
017760  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
017762  122110                  122110                  ;LOAD OUT DATA
017764  005204                  INC     R4              ;INCREMENT TO NEXT CHARACTER
017766  004737  033634          JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
017772  010461  000004          MOV     R4,4(R1)        ;LOAD PORT4 WITH CHARACTER
017776  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020000  122110                  122110                  ;LOAD OUT DATA
020002  004737  033502          JSR     PC,OCOR         ;WAIT FOR OCOR TO SET
020006  104413  000002          DATACLK,        2       ;2 TICKS TO SET UP TRANSMITTER
020012  012737  000176  001302  MOV     #^B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
020020  104413  000001  64$:    DATACLK,        1       ;CLOCK FLAG ONCE
020024  106037  001302          RORB    $TMP2           ;SHIFT SOFT FLAG
020030  103405                  BCS     65$             ;BR IF BIT IS MARK
020032  004737  033450          JSR     PC,GETSI        ;LOOK AT BIT WINDOW
020036  103006                  BCC     66$             ;BR IF OK
020040  104026                  ERROR   26              ;ERROR IN FLAG CHAR
020042  000404                  BR      66$
020044  004737  033450  65$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
020050  103401                  BCS     66$             ;BR IF OK
020052  104026                  ERROR   26              ;ERROR IN FLAG CHAR
020054  005203          66$:    INC     R3              ;INC BIT COUNT
020056  022703  000010          CMP     #10,R3          ;FLAG DONE YET?
020062  001356                  BNE     64$             ;BR IF NO
020064  005003                  CLR     R3              ;CLEAR BIT COUNT
020066  005037  035250          CLR     BITCON          ;CLEAR BIT STUFF COUNTER
020072  005003          4$:     CLR     R3              ;CLEAR BIT COUNTER
020074  010502                  MOV     R5,R2           ;LOAD CHARACTER IN R2
020076  010237  020140          MOV     R2,6$                   ;LOAD CHAR FOR STUFF CHECK
020102  104413  000001  1$:     DATACLK,        1       ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
020106  106002                  RORB    R2              ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
020110  103005                  BCC     2$              ;BR IF CARRY CLEAR
020112  004737  033450          JSR     PC,GETSI        ;GET THE WINDOW
020116  103406                  BCS     3$              ;BR IF BIT IS A MARK
020120  104006                  ERROR   6               ;ERROR BIT WAS A SPACE
020122  000404                  BR      3$              ;CONTINE WITH TEST
020124  004737  033450  2$:     JSR     PC,GETSI        ;GET THE WINDOW
020130  103001                  BCC     3$              ;BR IF BIT IS A SPACE
```

```
020132  104006                          ERROR   6               ;ERROR BIT WAS A MARK
020134                          3$:
020134  004537  035132                  JSR     R5,STFFCK       ;CHECK FOR BIT STUFF
020140  000000                  6$:     0                       ;DATA CHARACTER
020142  000001                          1                       ;SHIFT COUNT
020144  010237  020140                  MOV     R2,6$           ;LOAD CHAR FOR STUFF CHECK
020150  005203                          INC     R3              ;NEXT BIT
020152  022703  000010                  CMP     #10,R3          ;DONE YET?
020156  001351                          BNE     1$              ;BR IF NO
020160  005204                          INC     R4              ;NEXT CHARACTER
020162  004737  033634                  JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
020166  010461  000004                  MOV     R4,4(R1)        ;LOAD PORT4 WITH CHARACTER
020172  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020174  122110                          122110                  ;LOAD OUT DATA
020176  005205                          INC     R5              ;NEXT CHARACTER
020200  022705  000400                  CMP     #400,R5         ;DONE YET?
020204  001332                          BNE     4$              ;BR IF NO
020206                          5$:
```

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

                        ;  TEST 25
                        ;-------------
                        ;;************************************************************
020206  000004          TST25:  SCOPE
020210  012737  000025  001202          MOV     #25,$TSTNM              ; LOAD THE NO. OF THIS TEST
020216  012737  020474  001442          MOV     #TST26,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
020224  104410                          MSTCLR                  ;MASTER CLEAR KMC11
020226  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
020232  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020234  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
020236  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
020242  012711  004000                  MOV     #BIT11,(R1)     ;SET LU LOOP
020246  012700  000005                  MOV     #5,R0           ;FLAG COUNT
020252  005003                          CLR     R3              ;CLEAR BIT COUNTER
020254  004737  033634          1$:     JSR     PC,OUTRDY       ;WAIT FOR OUT-READY
020260  012761  000001  000004          MOV     #1,4(R1)        ;SET BIT0 IN PORT4
020266  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020270  122111                          122111                  ;SET SOM!
020272  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020274  122110                          122110                  ;LOAD GARBAGE CHAR
020276  005300                          DEC     R0              ;DEC COUNT
020300  001365                          BNE     1$              ;LOAD ANOTHER
020302  004737  033634                  JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
020306  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
020312  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
020314  122110                          122110                  ;LOAD A ZERO
020316  004737  033502                  JSR     PC,OCOR         ;WAIT
```

```
3318  020322  012700  000005          MOV     #5,R0           ;R0 = FLAG COUNT
3319  020326  104413  000002          DATACLK,        2       ;SET UP TRANSMITTER
3320  020332                  2$:
3321  020332  012737  000176  001302  MOV     #^B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
3322  020340  104413  000001  64$:    DATACLK,        1       ;CLOCK FLAG ONCE
3323  020344  106037  001302          RORB    $TMP2           ;SHIFT SOFT FLAG
3324  020350  103405                  BCS     65$             ;BR IF BIT IS MARK
3325  020352  004737  033450          JSR     PC,GETSI        ;LOOK AT BIT WINDOW
3326  020356  103006                  BCC     66$             ;BR IF OK
3327  020360  104026                  ERROR   26              ;ERROR IN FLAG CHAR
3328  020362  000404                  BR      66$
3329  020364  004737  033450  65$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
3330  020370  103401                  BCS     66$             ;BR IF OK
3331  020372  104026                  ERROR   26              ;ERROR IN FLAG CHAR
3332  020374  005203  66$:    INC     R3              ;INC BIT COUNT
3333  020376  022703  000010          CMP     #10,R3          ;FLAG DONE YET?
3334  020402  001356                  BNE     64$             ;BR IF NO
3335  020404  005003                  CLR     R3              ;CLEAR BIT COUNT
3336  020406  005300                  DEC     R0              ;DEC COUNT
3337  020410  001350                  BNE     2$              ;BR IF NOT DONE
3338  020412  005003                  CLR     R3              ;R3 = BIT COUNT
3339  020414  005005                  CLR     R5              ;R5 = "EXPECTED"
3340  020416  104413  000001  3$:     DATACLK,        1       ;CLOCK ONCE
3341  020422  004737  033450          JSR     PC,GETSI        ;GO LOOK AT WINDOW
3342  020426  103001                  BCC     4$              ;BR IF A SPACE
3343  020430  104006                  ERROR   6               ;ERROR, A MARK WAS SEEN
3344  020432  005203  4$:     INC     R3              ;INC BIT COUNT
3345  020434  022703  000010          CMP     #10,R3          ;DONE YET?
3346  020440  001366                  BNE     3$              ;BR IF NO
3347  020442  005003                  CLR     R3              ;CLEAR BIT COUNT
3348  020444  012705  000377          MOV     #377,R5         ;R5 = "EXPECTED"
3349  020450  104413  000001  5$:     DATACLK,        1       ;CLOCK ONCE
3350  020454  004737  033450          JSR     PC,GETSI        ;LOOK AT WINDOW
3351  020460  103401                  BCS     6$              ;BR IF A MARY
3352  020462  104033                  ERROR   33              ;ERROR, A SPACE WAS SEEN
3353  020464  005203  6$:     INC     R3              ;INC BIT COUNT
3354  020466  022703  000010          CMP     #10,R3          ;DONE YET?
3355  020472  001366                  BNE     5$              ;BR IF NO
3356
3357
3358                          ;*************************** TEST 26 ****************************
3359                          ;*LEADING ZEROS TEST
3360                          ;*VERIFY THAT THE SETTING OF SOM AND EOM TOGETHER
3361                          ;*AND THEN SOM ALONE WILL GENERATE 16 LEADING ZEROS
3362                          ;*AND A FLAG,THE CHECK IS MADE USING THE BIT WINDOW
3363                          ;:*************************************************************
3364
3365                          ;  TEST 26
3366                          ;-------------
3367                          ;;*************************************************************
3368  020474  000004  TST26:  SCOPE
3369  020476  012737  000026  001202  MOV     #26,$TSTNM              ; LOAD THE NO. OF THIS TEST
3370  020504  012737  020714  001442  MOV     #TST27,NEXT             ; POINT TO THE START OF NEXT TEST.
3371                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3372  020512  104410                  MSTCLR                  ;MASTER CLEAR KMC11
3373  020514  005061  000004          CLR     4(R1)           ;CLEAR PORT4
```

```
  3374  020520  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
  3375  020522  122117                          122117                  ;SET TO BITSTUFF MODE
  3376  020524  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
  3377  020530  012711  004000                  MOV     #BIT11,(R1)     ;SET LU LOOP
  3378  020534  004737  033634                  JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
  3379  020540  012761  000003  000004          MOV     #3,4(R1)        ;LOAD PORT4
  3380  020546  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
  3381  020550  122111                          122111                  ;SET SOM & EOM
  3382  020552  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
  3383  020554  122110                          122110                  ;GARBAGE CHARACTER
  3384  020556  012761  000001  000004          MOV     #1,4(R1)        ;LOAD PORT4
  3385  020564  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
  3386  020566  122111                          122111                  ;SET SOM
  3387  020570  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
  3388  020572  122110                          122110                  ;GARBAGE CHAR
  3389  020574  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
  3390  020576  122110                          122110                  ;GARBAGE CHAR
  3391  020600  004737  033502                  JSR     PC,OCOR         ;WAIT FOR OCOR
  3392  020604  005000                          CLR     R0              ;R0 = BIT COUNT
  3393  020606  104413  000002                  DATACLK,2               ;SET UP TRANSMITTER
  3394  020612  104413  000001          1$:     DATACLK,1               ;SINGLE CLOCK TRANSMITTER
  3395  020616  004737  033450                  JSR     PC,GETSI        ;LOOK AT BITWINDOW
  3396  020622  103001                          BCC     .+4
  3397  020624  104041                          ERROR   41              ;ERROR WINDOW WAS A MARK
  3398  020626  005200                          INC     R0
  3399  020630  022700  000020                  CMP     #16.,R0         ;16 ZEROS YET?
  3400  020634  001366                          BNE     1$              ;BR IF NO
  3401  020636  005003                          CLR     R3              ;R3 = BIT COUNT
  3402  020640  012737  000176  001302          MOV     #^B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
  3403  020646  104413  000001          64$:    DATACLK,        1       ;CLOCK FLAG ONCE
  3404  020652  106037  001302                  RORB    $TMP2           ;SHIFT SOFT FLAG
  3405  020656  103405                          BCS     65$             ;BR IF BIT IS MARK
  3406  020660  004737  033450                  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
  3407  020664  103006                          BCC     66$             ;BR IF OK
  3408  020666  104026                          ERROR   26              ;ERROR IN FLAG CHAR
  3409  020670  000404                          BR      66$
  3410  020672  004737  033450          65$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
  3411  020676  103401                          BCS     66$             ;BR IF OK
  3412  020700  104026                          ERROR   26              ;ERROR IN FLAG CHAR
  3413  020702  005203                  66$:    INC     R3              ;INC BIT COUNT
  3414  020704  022703  000010                  CMP     #10,R3          ;FLAG DONE YET?
  3415  020710  001356                          BNE     64$             ;BR IF NO
  3416  020712  005003                          CLR     R3              ;CLEAR BIT COUNT
  3417
  3418
  3419                                  ;************************** TEST 27 **************************
  3420                                  ;*BITSTUFF STRIP FLAG TEST
  3421                                  ;*SET LU LOOP, SINGLE STEP 5 FLAGS,
  3422                                  ;*VERIFY THAT IN ACTIVE DOES NOT SET
  3423                                  ;;*********************************************************************
  3424
  3425                                  ;  TEST 27
  3426                                  ;-------------
  3427                                  ;;*********************************************************************
  3428  020714  000004                  TST27:  SCOPE
  3429  020716  012737  000027  001202          MOV     #27,$TSTNM              ; LOAD THE NO. OF THIS TEST
```

```
3430  020724  012737  021016  001442          MOV     #TST30,NEXT             ; POINT TO THE START OF NEXT TEST.
3431                                          ;R1 CONTAINS BASE KMC11 ADDRESS
3432  020732  104410                         MSTCLR                          ;MASTER CLEAR KMC11
3433  020734  005061  000004                 CLR     4(R1)                   ;CLEAR PORT4
3434  020740  104412                         ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3435  020742  122117                         122117                          ;PUT LINE UNIT IN BITSTUFF MODE
3436  020744  004737  035032                 JSR     PC,CLRIO                ;DO THIS AFTER MODE IS SET
3437  020750  012711  004000                 MOV     #BIT11,(R1)             ;SET LU LOOP
3438  020754  012702  000012                 MOV     #12,R2                  ;SAVE LU REG FOR TYPEOUT
3439  020760  004737  033520                 JSR     PC,SYNC                 ;SINGLE CLOCK 5 SYNC CHARACTERS
3440  020764  000005                         5
3441  020766  104413  000054                 DATACLK,        54
3442  020772  104412                         ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3443  020774  021244                         021244                          ;PORT4←LU12
3444  020776  016104  000004                 MOV     4(R1),R4                ;PUT "FOUND" IN R4
3445  021002  042704  000277                 BIC     #277,R4                 ;CLEAR UNWANTED BITS
3446  021006  005005                         CLR     R5                      ;PUT "EXPECTED" IN R5
3447  021010  120504                         CMPB    R5,R4                   ;IS ACTIVE CLEAR?
3448  021012  001401                         BEQ     1$                      ;BR IF YES
3449  021014  104040                         ERROR   40                      ;ERROR ACTIVE IS NOT CLEAR
3450  021016                          1$:
3451
3452
3453                                          ;************************** TEST 30 **************************
3454                                          ;*BITSTUFF IN ACTIVE TEST
3455                                          ;*SET LU LOOP, SINGLE STEP 5 FLAGS AND A NON-FLAG (301)
3456                                          ;*VERIFY THAT IN ACTIVE IS SET
3457                                          ;;*********************************************************
3458
3459                                          ;  TEST 30
3460                                          ;--------------
3461                                          ;;*********************************************************
3462  021016  000004                  TST30:  SCOPE
3463  021020  012737  000030  001202          MOV     #30,$TSTNM              ; LOAD THE NO. OF THIS TEST
3464  021026  012737  021122  001442          MOV     #TST31,NEXT             ; POINT TO THE START OF NEXT TEST.
3465                                          ;R1 CONTAINS BASE KMC11 ADDRESS
3466  021034  104410                         MSTCLR                          ;MASTER CLEAR KMC11
3467  021036  005061  000004                 CLR     4(R1)                   ;CLEAR PORT4
3468  021042  104412                         ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3469  021044  122117                         122117                          ;PUT LINE UNIT IN BITSTUFF MODE
3470  021046  004737  035032                 JSR     PC,CLRIO                ;DO THIS AFTER MODE IS SET
3471  021052  012711  004000                 MOV     #BIT11,(R1)             ;SET LU LOOP
3472  021056  012702  000012                 MOV     #12,R2                  ;SAVE LU REG FOR TYPEOUT
3473  021062  004737  033520                 JSR     PC,SYNC                 ;SINGLE CLOCK 5 SYNC CHARACTERS
3474  021066  000005                         5
3475  021070  104413  000064                 DATACLK,        64
3476  021074  104412                         ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3477  021076  021244                         021244                          ;PORT4←LU12
3478  021100  016104  000004                 MOV     4(R1),R4                ;PUT "FOUND" IN R4
3479  021104  042704  000277                 BIC     #277,R4                 ;CLEAR UNWANTED BITS
3480  021110  012705  000100                 MOV     #BIT6,R5                ;PUT "EXPECTED" IN R5
3481  021114  120504                         CMPB    R5,R4                   ;IS ACTIVE SET?
3482  021116  001401                         BEQ     1$                      ;BR IF YES
3483  021120  104040                         ERROR   40                      ;ERROR ACTIVE IS NOT SET
3484  021122                          1$:
3485
```

```
3486
3487                                    ;************************** TEST 31 **************************
3488                                    ;*BITSTUFF IN ACTIVE TEST
3489                                    ;*SET LINE UNIT LOOP,SINGLE STEP ONE FLAG AND A CHAR (301)
3490                                    ;*VERIFY THAT IN ACTIVE IS SET
3491                                    ;;*********************************************************************
3492
3493                                    ;  TEST 31
3494                                    ;-------------
3495                                ;;*********************************************************************
3496  021122  000004                TST31:  SCOPE
3497  021124  012737  000031  001202        MOV     #31,$TSTNM              ; LOAD THE NO. OF THIS TEST
3498  021132  012737  021260  001442        MOV     #TST32,NEXT             ; POINT TO THE START OF NEXT TEST.
3499                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
3500  021140  104410                       MSTCLR                  ;MASTER CLEAR KMC11
3501  021142  005061  000004               CLR     4(R1)           ;CLEAR PORT4
3502  021146  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3503  021150  122117                       122117                  ;PUT LINE UNIT IN BITSTUFF MODE
3504  021152  004737  035032               JSR     PC,CLRIO        ;MUST DO THIS AFTER MODE IS SET
3505  021156  012711  004000               MOV     #BIT11,(R1)
3506  021162  012702  000012               MOV     #12,R2          ;SAVE REG ADDRESS FOR TYPEOUT
3507  021166  004737  033634               JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
3508  021172  012761  000001  000004       MOV     #1,4(R1)        ;LOAD PORT4
3509  021200  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3510  021202  122111                       122111                  ;SET SOM
3511  021204  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3512  021206  122110                       122110                  ;LOAD GARBAGE CHAR
3513  021210  012761  000301  000004       MOV     #301,4(R1)      ;LOAD PORT4
3514  021216  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3515  021220  122110                       122110                  ;LOAD OUT DATA
3516  021222  004737  033502               JSR     PC,OCOR         ;WAIT FOR OCOR
3517  021226  104413  000023               DATACLK,        23      ;SINGLE CLOCK THE DATA
3518  021232  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3519  021234  021244                       021244                  ;PORT4+LU-12
3520  021236  016104  000004               MOV     4(R1),R4        ;PUT "FOUND" IN R4
3521  021242  042704  000277               BIC     #277,R4         ;CLEAR UNWANTED BITS
3522  021246  012705  000100               MOV     #BIT6,R5                ;PUT "EXPECTED" IN R5
3523  021252  120504                       CMPB    R5,R4           ;IS IN ACTIVE SET?
3524  021254  001401                       BEQ     1$
3525  021256  104040                       ERROR   40              ;ERROR, IN ACTIVE NOT SET
3526  021260                        1$:
3527
3528
3529                                    ;************************** TEST 32 **************************
3530                                    ;*BITSTUFF IN ACTIVE TEST
3531                                    ;*SET LU LOOP, SINGLE STEP 2 FLAGS AND A NON-FLAG (301)
3532                                    ;*VERIFY THAT IN ACTIVE IS SET
3533                                    ;;*********************************************************************
3534
3535                                    ;  TEST 32
3536                                    ;-------------
3537                                ;;*********************************************************************
3538  021260  000004                TST32:  SCOPE
3539  021262  012737  000032  001202        MOV     #32,$TSTNM              ; LOAD THE NO. OF THIS TEST
3540  021270  012737  021364  001442        MOV     #TST33,NEXT             ; POINT TO THE START OF NEXT TEST.
3541                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
```

```
3542  021276  104410                  MSTCLR                  ;MASTER CLEAR KMC11
3543  021300  005061  000004          CLR     4(R1)           ;CLEAR PORT4
3544  021304  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3545  021306  122117                  122117                  ;PUT LINE UNIT IN BITSTUFF MODE
3546  021310  004737  035032          JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
3547  021314  012711  004000          MOV     #BIT11,(R1)     ;SET LU LOOP
3548  021320  012702  000012          MOV     #12,R2          ;SAVE LU REG FOR TYPEOUT
3549  021324  004737  033520          JSR     PC,SYNC         ;SINGLE CLOCK 2 SYNC CHARACTERS
3550  021330  000002                  2
3551  021332  104413  000033          DATACLK,        33
3552  021336  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3553  021340  021244                  021244                  ;PORT4←LU12
3554  021342  016104  000004          MOV     4(R1),R4        ;PUT "FOUND" IN R4
3555  021346  042704  000277          BIC     #277,R4         ;CLEAR UNWANTED BITS
3556  021352  012705  000100          MOV     #BIT6,R5                ;PUT "EXPECTED" IN R5
3557  021356  120504                  CMPB    R5,R4           ;IS ACTIVE SET?
3558  021360  001401                  BEQ     1$              ;BR IF YES
3559  021362  104040                  ERROR   40              ;ERROR ACTIVE IS NOT SET
3560  021364                  1$:
3561
3562
3563                                  ;************************** TEST 33 **************************
3564                                  ;*IN CLEAR TEST
3565                                  ;*SYNC UP RECEIVER AND TRANSMIT A CHARACTER
3566                                  ;*WAIT FOR IN RDY, THEN SET IN CLEAR
3567                                  ;*VERIFY THAT IN ACTIVE AND IN RDY ARE CLEARED
3568                                  ;;*********************************************************************
3569
3570                                  ;  TEST 33
3571                                  ;-------------
3572                                  ;;*********************************************************************
3573  021364  000004          TST33:  SCOPE
3574  021366  012737  000033  001202  MOV     #33,$TSTNM              ; LOAD THE NO. OF THIS TEST
3575  021374  012737  021570  001442  MOV     #TST34,NEXT             ; POINT TO THE START OF NEXT TEST.
3576                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3577  021402  104410                  MSTCLR                  ;MASTER CLEAR KMC11
3578  021404  005061  000004          CLR     4(R1)           ;CLEAR PORT4
3579  021410  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3580  021412  122117                  122117                  ;PUT LINE UNIT IN BITSTUFF MODE
3581  021414  004737  035032          JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
3582  021420  012702  000012          MOV     #12,R2          ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3583  021424  012711  004000          MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
3584  021430  012761  000001  000004  MOV     #1,4(R1)        ;SET BIT0 IN PORT4
3585  021436  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3586  021440  122111                  122111                  ;SET SOM!
3587  021442  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3588  021444  122110                  122110                  ;LOAD GARBAGE CHAR
3589  021446  004737  034000          JSR     PC,CHARSD       ;LOAD SILO WITH CHARACTER
3590  021452  000026                  26                      ;CHARACTER
3591  021454  104413  000033          DATACLK,        33      ;SINGLE CLOCK THE DATA
3592  021460  104414  000002          TIMER,  2               ;WAIT FOR INRDY
3593  021464  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3594  021466  021244                  021244                  ;PORT4←LU 12
3595  021470  016104  000004          MOV     4(R1),R4        ;PUT "FOUND" IN R4
3596  021474  042704  000357          BIC     #357,R4         ;CLEAR UNWANTED BITS
3597  021500  012705  000020          MOV     #BIT4,R5                ;PUT "EXPECTED" IN R5
```

```
3598  021504  120504                        CMPB    R5,R4           ;IS INRDY SET?
3599  021506  001401                        BEQ     1$
3600  021510  104040                        ERROR   40              ;ERROR, INRDY IS NOT SET
3601  021512                          1$:
3602  021512  012761  000200  000004        MOV     #BIT7,4(R1)     ;LOAD PORT4
3603  021520  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3604  021522  122112                        122112                  ;SET IN CLEAR
3605  021524  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3606  021526  021244                        021244                  ;PORT4=LU 12
3607  021530  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
3608  021534  042704  000277                BIC     #277,R4         ;CLEAR UNWANTED BITS
3609  021540  005005                        CLR     R5              ;PUT "EXPECTED" IN R5
3610  021542  120504                        CMPB    R5,R4           ;IS IN ACTIVE CLEAR?
3611  021544  001401                        BEQ     2$
3612  021546  104040                        ERROR   40              ;ERROR, IN ACTIVE IS NOT CLEAR
3613  021550                          2$:
3614  021550  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
3615  021554  042704  000357                BIC     #357,R4         ;CLEAR UNWANTED BITS
3616  021560  005005                        CLR     R5              ;PUT "EXPECTED" IN R5
3617  021562  120504                        CMPB    R5,R4           ;IS INRDY CLEARED?
3618  021564  001401                        BEQ     3$
3619  021566  104040                        ERROR   40              ;ERROR, INRDY IS NOT CLEARED
3620  021570                          3$:
3621
3622
3623                                        ;************************** TEST 34 **************************
3624                                        ;*BITSTUFF BASIC RECEICER TEST
3625                                        ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 0
3626                                        ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3627                                        ;;*********************************************************************
3628
3629                                        ;  TEST 34
3630                                        ;--------------
3631                                    ;;**********************************************************************
3632  021570  000004                  TST34:  SCOPE
3633  021572  012737  000034  001202        MOV     #34,$TSTNM              ; LOAD THE NO. OF THIS TEST
3634  021600  012737  021736  001442        MOV     #TST35,NEXT             ; POINT TO THE START OF NEXT TEST.
3635                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3636  021606  104410                        MSTCLR                  ;MASTER CLEAR KMC11
3637  021610  005061  000004                CLR     4(R1)           ;CLEAR PORT4
3638  021614  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3639  021616  122117                        122117                  ;PUT LINE UNIT IN BITSTUFF MODE
3640  021620  004737  035032                JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
3641  021624  012702  000012                MOV     #12,R2          ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3642  021630  012711  004000                MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
3643  021634  012761  000001  000004        MOV     #1,4(R1)        ;SET BIT0 IN PORT4
3644  021642  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3645  021644  122111                        122111                  ;SET SOM!
3646  021646  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3647  021650  122110                        122110                  ;LOAD GARBAGE CHAR
3648  021652  004737  034000                JSR     PC,CHARSD       ;LOAD SILO WITH CHARACTER
3649  021656  000000                        0                       ;CHARACTER
3650  021660  104413  000033                DATACLK,        33      ;SINGLE CLOCK THE DATA
3651  021664  104414  000002                TIMER,  2               ;WAIT FOR INRDY
3652  021670  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3653  021672  021244                        021244                  ;PORT4=LU 12
```

```
3654  021674  016104  000004              MOV     4(R1),R4        ;PUT "FOUND" IN R4
3655  021700  042704  000357              BIC     #357,R4         ;CLEAR UNWANTED BITS
3656  021704  012705  000020              MOV     #BIT4,R5                ;PUT "EXPECTED" IN R5
3657  021710  120504                      CMPB    R5,R4           ;IS INRDY SET?
3658  021712  001401                      BEQ     1$
3659  021714  104040                      ERROR   40              ;ERROR, INRDY IS NOT SET
3660  021716                      1$:
3661  021716  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3662  021720  021204                      021204                  ;PORT4<-IN DATA
3663  021722  016104  000004              MOV     4(R1),R4        ;PUT "FOUND" IN R4
3664  021726  005005                      CLR     R5              ;PUT "EXPECTED" IN R5
3665  021730  120504                      CMPB    R5,R4           ;WAS A 0 RECEIVED?
3666  021732  001401                      BEQ     2$
3667  021734  104010                      ERROR   10              ;ERROR, RECEIVED DATA IS WRONG
3668  021736                      2$:
3669
3670
3671                                      ;************************** TEST 35 **************************
3672                                      ;#BITSTUFF BASIC RECEICER TEST
3673                                      ;#SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 125
3674                                      ;#VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3675                                      ;;*************************************************************
3676
3677                                      ;  TEST 35
3678                                      ;--------------
3679                              ;;*****************************************************************
3680  021736  000004              TST35:  SCOPE
3681  021740  012737  000035  001202      MOV     #35,$TSTNM              ; LOAD THE NO. OF THIS TEST
3682  021746  012737  022106  001442      MOV     #TST36,NEXT             ; POINT TO THE START OF NEXT TEST.
3683                                                              ;R1 CONTAINS BASE KMC11 ADDRESS
3684  021754  104410                      MSTCLR                  ;MASTER CLEAR KMC11
3685  021756  005061  000004              CLR     4(R1)           ;CLEAR PORT4
3686  021762  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3687  021764  122117                      122117                  ;PUT LINE UNIT IN BITSTUFF MODE
3688  021766  004737  035032              JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
3689  021772  012702  000012              MOV     #12,R2          ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3690  021776  012711  004000              MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
3691  022002  012761  000001  000004      MOV     #1,4(R1)        ;SET BIT0 IN PORT4
3692  022010  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3693  022012  122111                      122111                  ;SET SOM!
3694  022014  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3695  022016  122110                      122110                  ;LOAD GARBAGE CHAR
3696  022020  004737  034000              JSR     PC,CHARSD       ;LOAD SILO WITH CHARACTER
3697  022024  000125                      125                     ;CHARACTER
3698  022026  104413  000033              DATACLK,        33      ;SINGLE CLOCK THE DATA
3699  022032  104414  000002              TIMER,  2               ;WAIT FOR INRDY
3700  022036  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3701  022040  021244                      021244                  ;PORT4<-LU 12
3702  022042  016104  000004              MOV     4(R1),R4        ;PUT "FOUND" IN R4
3703  022046  042704  000357              BIC     #357,R4         ;CLEAR UNWANTED BITS
3704  022052  012705  000020              MOV     #BIT4,R5                ;PUT "EXPECTED" IN R5
3705  022056  120504                      CMPB    R5,R4           ;IS INRDY SET?
3706  022060  001401                      BEQ     1$
3707  022062  104040                      ERROR   40              ;ERROR, INRDY IS NOT SET
3708  022064                      1$:
3709  022064  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
3710  022066  021204                          021204                  ;PORT4←IN DATA
3711  022070  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
3712  022074  012705  000125                  MOV     #125,R5         ;PUT "EXPECTED" IN R5
3713  022100  120504                          CMPB    R5,R4           ;WAS A 125 RECEIVED?
3714  022102  001401                          BEQ     2$
3715  022104  104010                          ERROR   10              ;ERROR, RECEIVED DATA IS WRONG
3716  022106                          2$:
3717
3718                                          ;************************** TEST 36 **************************
3719                                          ;#BITSTUFF BASIC RECEICER TEST
3720                                          ;#SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 252
3721                                          ;#VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3722                                          ;;*************************************************************
3723
3724                                          ;  TEST 36
3725                                          ;--------------
3726                                          ;;*************************************************************
3727                              TST36:      SCOPE
3728  022106  000004
3729  022110  012737  000036  001202          MOV     #36,$TSTNM              ; LOAD THE NO. OF THIS TEST
3730  022116  012737  022256  001442          MOV     #TST37,NEXT             ; POINT TO THE START OF NEXT TEST.
3731                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3732  022124  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3733  022126  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
3734  022132  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3735  022134  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
3736  022136  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
3737  022142  012702  000012                  MOV     #12,R2          ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3738  022146  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
3739  022152  012761  000001  000004          MOV     #1,4(R1)        ;SET BIT0 IN PORT4
3740  022160  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3741  022162  122111                          122111                  ;SET SOM!
3742  022164  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3743  022166  122110                          122110                  ;LOAD GARBAGE CHAR
3744  022170  004737  034000                  JSR     PC,CHARSD       ;LOAD SILO WITH CHARACTER
3745  022174  000252                          252                     ;CHARACTER
3746  022176  104413  000033                  DATACLK,        33      ;SINGLE CLOCK THE DATA
3747  022202  104414  000002                  TIMER,  2               ;WAIT FOR INRDY
3748  022206  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3749  022210  021244                          021244                  ;PORT4←LU 12
3750  022212  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
3751  022216  042704  000357                  BIC     #357,R4         ;CLEAR UNWANTED BITS
3752  022222  012705  000020                  MOV     #BIT4,R5                ;PUT "EXPECTED" IN R5
3753  022226  120504                          CMPB    R5,R4           ;IS INRDY SET?
3754  022230  001401                          BEQ     1$
3755  022232  104040                          ERROR   40              ;ERROR, INRDY IS NOT SET
3756  022234                          1$:
3757  022234  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3758  022236  021204                          021204                  ;PORT4←IN DATA
3759  022240  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
3760  022244  012705  000252                  MOV     #252,R5         ;PUT "EXPECTED" IN R5
3761  022250  120504                          CMPB    R5,R4           ;WAS A 252 RECEIVED?
3762  022252  001401                          BEQ     2$
3763  022254  104010                          ERROR   10              ;ERROR, RECEIVED DATA IS WRONG
3764  022256                          2$:
3765
```

```
                                        ;*************************** 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
                                        ;-------------
                                        ;:************************************************************
022256  000004                  TST37:  SCOPE
022260  012737  000037  001202          MOV     #37,$TSTNM              ; LOAD THE NO. OF THIS TEST
022266  012737  022426  001442          MOV     #TST40,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
022274  104410                          MSTCLR                  ;MASTER CLEAR KMC11
022276  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
022302  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022304  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
022306  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
022312  012702  000012                  MOV     #12,R2          ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
022316  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
022322  012761  000001  000004          MOV     #1,4(R1)        ;SET BIT0 IN PORT4
022330  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022332  122111                          122111                  ;SET SOM!
022334  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022336  122110                          122110                  ;LOAD GARBAGE CHAR
022340  004737  034000                  JSR     PC,CHARSD       ;LOAD SILO WITH CHARACTER
022344  000377                          377                     ;CHARACTER
022346  104413  000034                  DATACLK,        34      ;SINGLE CLOCK THE DATA
022352  104414  000002                  TIMER,  2               ;WAIT FOR INRDY
022356  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022360  021244                          021244                  ;PORT4←LU 12
022362  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
022366  042704  000357                  BIC     #357,R4         ;CLEAR UNWANTED BITS
022372  012705  000020                  MOV     #BIT4,R5                ;PUT "EXPECTED" IN R5
022376  120504                          CMPB    R5,R4           ;IS INRDY SET?
022400  001401                          BEQ     1$
022402  104040                          ERROR   40              ;ERROR, INRDY IS NOT SET
022404                          1$:
022404  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022406  021204                          021204                  ;PORT4←IN DATA
022410  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
022414  012705  000377                  MOV     #377,R5         ;PUT "EXPECTED" IN R5
022420  120504                          CMPB    R5,R4           ;WAS A 377 RECEIVED?
022422  001401                          BEQ     2$
022424  104010                          ERROR   10              ;ERROR, RECEIVED DATA IS WRONG
022426                          2$:


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


                                        ;  TEST 40
```

```
                                        ;---------------
                                    ;;**********************************************************************
022426  000004                    TST40:  SCOPE
022430  012737  000040  001202            MOV     #40,$TSTNM              ; LOAD THE NO. OF THIS TEST
022436  012737  022602  001442            MOV     #TST41,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
022444  104410                            MSTCLR                          ;MASTER CLEAR KMC11
022446  005061  000004                    CLR     4(R1)                   ;CLEAR PORT4
022452  104412                            ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022454  122117                            122117                          ;PUT LINE UNIT IN BITSTUFF MODE
022456  004737  035032                    JSR     PC,CLRIO                ;DO THIS AFTER MODE IS SET
022462  005037  034304                    CLR     SCHAR                   ;START BINARY COUNT AT ZERO
022466  005137  034304                    COM     SCHAR                   ;IF BITSTUFF SCHAR IS MINUS NUMBER
022472  005037  035250                    CLR     BITCON                  ;START  1'S COUNT AT 0
022476  005037  034306                    CLR     STUFLG                  ;CLEAR BITSTUFF FLAG
022502  005002                            CLR     R2                      ;R2 IS "EXPECTED" DATA
022504  012703  000073                    MOV     #73,R3                  ;R3 IS CHARACTER COUNT
022510  012711  004000                    MOV     #BIT11,(R1)             ;SET LINE UNIT LOOP
022514  004737  034044                    JSR     PC,SILOLD               ;LOAD SILO WITH COUNT PATTERN
022520  104413  000023                    DATACLK,        23              ;SYNC RECEIVER AND GET IT ACTIVE
022524  104413  000730            1$:     DATACLK,        730             ;CLOCK IN 73 CHARACTERS
022530  004737  034310            4$:     JSR     PC,INRDY                ;WAIT FOR INRDY
022534  104412                            ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022536  021204                            021204                          ;PORT4->IN DATA
022540  016104  000004                    MOV     4(R1),R4                ;PUT "FOUND" IN R4
022544  010205                            MOV     R2,R5                   ;PUT "EXPECTED" IN R5
022546  120504                            CMPB    R5,R4                   ;IS DATA CORRECT?
022550  001401                            BEQ     2$                      ;BR IF YES
022552  104010                            ERROR   10                      ;DATA ERROR
022554  005202                    2$:     INC     R2                      ;NEXT CHARACTER
022556  022702  000400                    CMP     #400,R2                 ;ALL DONE?
022562  001407                            BEQ     3$                      ;BR IF YES
022564  005303                            DEC     R3                      ;DECREMENT CHARACTER COUNT
022566  001360                            BNE     4$                      ;BR IF SILO NOT EMPTY
022570  004737  034044                    JSR     PC,SILOLD               ;LOAD SILO WITH MORE OF COUNT PATTERN
022574  012703  000073                    MOV     #73,R3                  ;RELOAD CHARACTER COUNT
022600  000751                            BR      1$                      ;CONTINUE
022602                            3$:
                                    ;**************************** TEST 41 ****************************
                                    ;*BITSTUFF DATA TEST
                                    ;*THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
                                    ;*CHECKING EACH CHARACTER AS IT IS RECEIVED
                                    ;*THIS TEST IS EXACTLY THE SAME AS THE LAST TEST,
                                    ;*EXCEPT LINE UNIT LOOP IS SET IN LU REGISTER 12
                                    ;;**********************************************************************
                                    ;  TEST 41
                                        ;-------------
                                    ;;**********************************************************************
022602  000004                    TST41:  SCOPE
022604  012737  000041  001202            MOV     #41,$TSTNM              ; LOAD THE NO. OF THIS TEST
022612  012737  022766  001442            MOV     #TST42,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
022620  104410                            MSTCLR                          ;MASTER CLEAR KMC11
```

```
3878  022622  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
3879  022626  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3880  022630  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
3881  022632  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
3882  022636  005037  034304                  CLR     SCHAR           ;START BINARY COUNT AT ZERO
3883  022642  005137  034304                  COM     SCHAR           ;IF BITSTUFF SCHAR IS MINUS NUMBER
3884  022646  005037  035250                  CLR     BITCON          ;START  1'S COUNT AT 0
3885  022652  005037  034306                  CLR     STUFLG          ;CLEAR BITSTUFF FLAG
3886  022656  005002                          CLR     R2              ;R2 IS "EXPECTED" DATA
3887  022660  012703  000073                  MOV     #73,R3          ;R3 IS CHARACTER COUNT
3888  022664  005011                          CLR     (R1)            ;CLEAR LU LOOP IN MAINT REG
3889  022666  012761  000040  000004          MOV     #BITS,4(R1)     ;LOAD PORT4
3890  022674  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3891  022676  122112                          122112                  ;SET LU LOOP IN LU REG 12
3892  022700  004737  034044                  JSR     PC,SILOLD       ;LOAD SILO WITH COUNT PATTERN
3893  022704  104413  000023                  DATACLK,        23      ;SYNC RECEIVER AND GET IT ACTIVE
3894  022710  104413  000730          1$:     DATACLK,        730     ;CLOCK IN 73 CHARACTERS
3895  022714  004737  034310          4$:     JSR     PC,INRDY        ;WAIT FOR INRDY
3896  022720  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3897  022722  021204                          021204                  ;PORT4->IN DATA
3898  022724  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
3899  022730  010205                          MOV     R2,R5           ;PUT "EXPECIED" IN R5
3900  022732  120504                          CMPB    R5,R4           ;IS DATA CORRECT?
3901  022734  001401                          BEQ     2$              ;BR IF YES
3902  022736  104010                          ERROR   10              ;DATA ERROR
3903  022740  005202          2$:             INC     R2              ;NEXT CHARACTER
3904  022742  022702  000400                  CMP     #400,R2         ;ALL DONE?
3905  022746  001407                          BEQ     3$              ;BR IF YES
3906  022750  005303                          DEC     R3              ;DECREMENT CHARACTER COUNT
3907  022752  001360                          BNE     4$              ;BR IF SILO NOT EMPTY
3908  022754  004737  034044                  JSR     PC,SILOLD       ;LOAD SILO WITH MORE OF COUNT PATTERN
3909  022760  012703  000073                  MOV     #73,R3          ;RELOAD CHARACTER COUNT
3910  022764  000751                          BR      1$              ;CONTINUE
3911  022766                          3$:
3912
3913
3914                                          ;*************************** TEST 42 ****************************
3915                                          ;*RECEIVER ABORT TEST
3916                                          ;*SINGLE CLOCK 3 FLAGS, A 301, ANOTHER 301 AND 10 EXTRA
3917                                          ;*CLOCK TICKS, VERIFY THAT A 301 AND A BLOCK END
3918                                          ;*WERE RECEIVED INDICATING THAT THE RECEIVER RECOGINIZED
3919                                          ;*THE ABORT SEQUENCE (8 CONTIGUOUS 1'S)
3920                                          ;;*************************************************************
3921
3922                                          ;  TEST 42
3923                                          ;--------------
3924                                  ;;*****************************************************************
3925  022766  000004                  TST42:  SCOPE
3926  022770  012737  000042  001202          MOV     #42,$TSTNM              ; LOAD THE NO. OF THIS TEST
3927  022776  012737  023130  001442          MOV     #TST43,NEXT             ; POINT TO THE START OF NEXT TEST.
3928                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3929  023004  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3930  023006  005061  000004                  CLR     4(R1)
3931  023012  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3932  023014  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
3933  023016  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
```

```
3934  023022  012711  004000              MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
3935  023026  004737  033666              JSR     PC,CHAR         ;LOAD SILO WITH 3 FLAGS
3936  023032  000301                      301                     ;AND A 301
3937  023034  004737  033634              JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
3938  023040  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3939  023042  122110                      122110                  ;LOAD 2ND 301 CHARACTER
3940  023044  104413  000073              DATACLK,        73      ;CLOCK THE 301 IN AND 10 EXTRA TICKS
3941  023050  004737  034310              JSR     PC,INRDY        ;WAIT FOR INRDY
3942  023054  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3943  023056  021204                      021204                  ;PORT4->IN DATA
3944  023060  016104  000004              MOV     4(R1),R4        ;PUT "FOUND" IN R4
3945  023064  012705  000301              MOV     #301,R5         ;PUT "EXPECTED" IN R5
3946  023070  120504                      CMPB    R5,R4           ;WAS A 301 RECEIVED?
3947  023072  001401                      BEQ     1$
3948  023074  104010                      ERROR   10              ;ERROR FIRST CHARACTER INCORRECT
3949  023076  004737  034310      1$:     JSR     PC,INRDY        ;WAIT FOR INRDY
3950  023102  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3951  023104  021244                      021244                  ;READ LU-12
3952  023106  016104  000004              MOV     4(R1),R4        ;PUT "FOUND" IN R4
3953  023112  042704  000375              BIC     #375,R4         ;CLEAR UNWANTED BITS
3954  023116  012705  000002              MOV     #2,R5           ;PUT "EXPECTED" IN R5
3955  023122  120504                      CMPB    R5,R4           ;IS BLOCK END SET?
3956  023124  001401                      BEQ     3$              ;BR IF YES
3957  023126  104032                      ERROR   32              ;ERROR, BLOCK END NOT SET
3958  023130                      3$:
3959
3960
3961                                      ;*************************** TEST 43 ****************************
3962                                      ;*CABLE TURNAROUND TEST
3963                                      ;*CLEAR LINE UNIT LOOP, SET DTR
3964                                      ;*VERIFY THAT RING AND MODEM READY ARE SET
3965                                      ;*CLEAR DTR, VERIFY THAT RING AND MRDY ARE CLEARED
3966                                      ;:************************************************************
3967
3968                                      ;  TEST 43
3969                                      ;-------------
3970                                  ;;**************************************************************
3971  023130  000004              TST43:  SCOPE
3972  023132  012737  000043  001202      MOV     #43,$TSTNM              ; LOAD THE NO. OF THIS TEST
3973  023140  012737  023326  001442      MOV     #TST44,NEXT             ; POINT TO THE START OF NEXT TEST.
3974                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3975  023146  104410                      MSTCLR                  ;MASTER CLEAR KMC11
3976  023150  032737  020000  002050      BIT     #BIT13,STAT1    ;IS LINE UNIT M8202?
3977  023156  001004                      BNE     .+12            ;BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
3978  023160  032737  040000  002050      BIT     #BIT14,STAT1    ;IS TURNAROUND CONNECTOR ON?
3979  023166  001457                      BEQ     2$              ;SKIP TEST IF NO
3980  023170  005011                      CLR     (R1)            ;CLEAR LINE UNIT LOOP
3981  023172  012761  000100  000004      MOV     #100,4(R1)      ;LOAD PORT4
3982  023200  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3983  023202  122113                      122113                  ;SET DTR
3984  023204  104414  000002              TIMER,  2               ;WAIT
3985  023210  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3986  023212  021264                      021264                  ;PORT4<-LU13
3987  023214  016104  000004              MOV     4(R1),R4        ;PUT "FOUND" IN R4
3988  023220  042704  000023              BIC     #23,R4          ;CLEAR UNWANTED BITS
3989  023224  012705  000310              MOV     #310,R5         ;PUT "EXPECTED" IN R5
```

```
3990  023230  032737  020000  002050          BIT     #BIT13,STAT1    ;IS LINE UNIT M8202?
3991  023236  001402                          BEQ     .+6             ;BR IF NO
3992  023240  042705  000200                  BIC     #BIT7,R5        ;NO RING ON M8202
3993  023244  120504                          CMPB    R5,R4           ;ARE RING AND MRDY SET?
3994  023246  001401                          BEQ     1$
3995  023250  104011                          ERROR   11              ;ERROR, RING OR MRDY NOT SET
3996  023252  005061  000004          1$:     CLR     4(R1)           ;CLEAR PORT4
3997  023256  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3998  023260  122113                          122113                  ;CLEAR DTR
3999  023262  104414  000002                  TIMER,  2
4000  023266  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4001  023270  021264                          021264                  ;PORT4=LU13
4002  023272  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
4003  023276  042704  000023                  BIC     #23,R4          ;CLEAR UNWANTED BITS
4004  023302  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
4005  023304  032737  020000  002050          BIT     #BIT13,STAT1    ;IS LINE UNIT M8202?
4006  023312  001402                          BEQ     .+6             ;BR IF NO
4007  023314  052705  000010                  BIS     #BIT3,R5        ;MRDY SET ON M8202
4008  023320  120504                          CMPB    R5,R4           ;ARE RING AND MRDY CLEAR?
4009  023322  001401                          BEQ     2$
4010  023324  104011                          ERROR   11              ;ERROR, RING OR MRDY NOT CLEAR
4011  023326                          2$:
4012
4013                                  ;************************** TEST 44 **************************
4014                                  ;*CABLE TURNAROUND TEST
4015                                  ;*CLEAR LINE UNIT LOOP, LOAD OUT DATA SILO
4016                                  ;*VERIFY THAT ALL MODEM SIGNALS ARE SET
4017                                  ;;*********************************************************
4018
4019
4020                                  ;  TEST 44
4021                                  ;-------------
4022                                  ;;*********************************************************
4023  023326  000004          TST44:  SCOPE
4024  023330  012737  000044  001202          MOV     #44,$TSTNM              ; LOAD THE NO. OF THIS TEST
4025  023336  012737  023506  001442          MOV     #TST45,NEXT             ; POINT TO THE START OF NEXT TEST.
4026                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4027  023344  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4028  023346  032737  020000  002050          BIT     #BIT13,STAT1    ;IS LINE UNIT M8202?
4029  023354  001004                          BNE     .+12            ;BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
4030  023356  032737  040000  002050          BIT     #BIT14,STAT1    ;IS TURNAROUND CONNECTOR ON?
4031  023364  001450                          BEQ     1$              ;SKIP TEST IF NO
4032  023366  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
4033  023372  012761  000100  000004          MOV     #100,   4(R1)           ;LOAD PORT4
4034  023400  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4035  023402  122113                          122113                  ;CLEAR ALL MODEM SIGNALS,EXCEPT DTR
4036  023404  104414  000002                  TIMER,  2               ;WAIT
4037  023410  012761  000001  000004          MOV     #1,4(R1)        ;LOAD PORT4
4038  023416  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4039  023420  122111                          122111                  ;SET SOM
4040  023422  004537  034770                  JSR     R5,MESLD        ;FILL OUT DATA SILO
4041  023426  035252                          MESDAT                  ;WITH 64 CHARACTERS
4042  023430  000100                          64.
4043  023432  012700  000050                  MOV     #50,R0          ;PREPARE FOR DELAY
4044  023436  005011                          CLR     (R1)            ;CLEAR LINE UNIT LOOP
4045  023440                          2$:
```

```
4046  023440  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4047  023442  021264                          021264                  ;PORT4=LU13
4048  023444  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
4049  023450  042704  000023                  BIC     #23,R4          ;CLEAR UNWANTED BITS
4050  023454  012705  000354                  MOV     #354,R5         ;PUT "EXPECTED" IN R5
4051  023460  032737  020000  002050          BIT     #BIT13,STAT1    ;IS LINE UNIT M8202?
4052  023466  001402                          BEQ     .+6             ;BR IF NO
4053  023470  042705  000200                  BIC     #BIT7,R5        ;NO RING ON M8202
4054  023474  120504                          CMPB    R5,R4           ;COMPARE EXPECTED AND FOUND
4055  023476  001403                          BEQ     1$              ;BR IF OK
4056  023500  005300                          DEC     R0              ;DEC DELAY COUNT
4057  023502  001356                          BNE     2$              ;BR IF NOT ZERO
4058  023504  104011                          ERROR   11              ;ERROR, ALL SIGNALS ARE NOT SET
4059  023506                          1$:
4060
4061
4062                                          ;*************************** TEST 45 ****************************
4063                                          ;*TEST OF CRC OPERATION
4064                                          ;*USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
4065                                          ;*0, VERIFY THE LSB OF THE BCC ON EACH SHIFT
4066                                          ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
4067                                          ;;************************************************************
4068
4069                                          ;  TEST 45
4070                                          ;--------------
4071                                  ;;**********************************************************
4072  023506  000004                  TST45:  SCOPE
4073  023510  012737  000045  001202          MOV     #45,$TSTNM              ; LOAD THE NO. OF THIS TEST
4074  023516  012737  024052  001442          MOV     #TST46,NEXT             ; POINT TO THE START OF NEXT TEST.
4075  023524  012737  023560  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
4076                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4077  023532  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4078  023534  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
4079  023540  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4080  023542  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
4081  023544  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
4082  023550  005037  035250                  CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
4083  023554  012711  004000                  MOV     #BIT11,(R1)     ;SET LU LOOP
4084  023560  004737  035032          64$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
4085  023564  005000                          CLR     R0              ;START SHIFT COUNTER AT ZERO
4086  023566  012737  102010  034466          MOV     #CRC.CCITT,XPOLY;LOAD POLYNOMIAL FOR SOFTWARE BCC
4087  023574  012737  000000  023640          MOV     #0,66$          ;LOAD CHAR FOR SOFTWARE BCC
4088  023602  005037  023642                  CLR     67$             ;CLEAR OLD SOFTWARE BCC
4089  023606  005137  023642                  COM     67$             ;START AT -1
4090  023612  004737  034472                  JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
4091  023616  000000                          0                       ;AND THE CHARACTER 0
4092  023620  104413  000021                  DATACLK,        21      ;GET TRANSMITTER ACTIVE
4093  023624  104413  000001          65$:    DATACLK,        1       ;SHIFT BCC ONCE
4094  023630  005200                          INC     R0              ;BUMP SHIFT COUNT
4095  023632  004537  034344                  JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
4096  023636  000001                          1                       ;ONE SHIFT
4097  023640  000000          66$:    0                               ;DATA CHARACTER
4098  023642  000000          67$:    0                               ;OLD BCC
4099  023644  103405                          BCS     68$             ;BR IF SOFT BCC LSB IS SET
4100  023646  004737  034604                  JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
4101  023652  103006                          BCC     69$             ;BR IF HARD BCC LSB IS CLEAR
```

```
4102  023654  104012                          ERROR   12              ;ERROR, BCC LSB IS SET
4103  023656  000404                          BR      69$             ;CONTINUE
4104  023660  004737  034604          68$:    JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
4105  023664  103401                          BCS     69$             ;BR IF HARD BCC LSB IS SET
4106  023666  104016                          ERROR   16              ;ERROR, HARD BCC LSB IS CLEAR
4107  023670                          69$:
4108  023670  006037  023640                  ROR     66$             ;SHIFT SOFT DATA
4109  023674  013737  034470  023642          MOV     CALBCC,67$      ;LOAD OLD SOFT BCC
4110  023702  022700  000010                  CMP     #10,R0          ;DONE YET?
4111  023706  001346                          BNE     65$             ;BR IF NOT DONE
4112  023710  104405                          SCOP1                   ;SCOPE SUBTEST (SW09=1)
4113  023712  012737  023720  001444          MOV     #71$,LOCK       ;NEW SCOPE1
4114  023720  004737  035032          71$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
4115  023724  005000                          CLR     R0              ;START SHIFT COUNTER AT ZERO
4116  023726  012737  102010  034466          MOV     #CRC.CCITT,XPOLY;LOAD POLYNOMIAL FOR SOFTWARE BCC
4117  023734  012737  000000  024000          MOV     #0,73$          ;LOAD CHAR FOR SOFTWARE BCC
4118  023742  005037  024002                  CLR     74$             ;CLEAR OLD SOFTWARE BCC
4119  023746  005137  024002                  COM     74$             ;START AT -1
4120  023752  004737  034472                  JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
4121  023756  000000                          0                       ;AND THE CHARACTER 0
4122  023760  104413  000032                  DATACLK,        32      ;GET RECEIVER ACTIVE
4123  023764  104413  000001          72$:    DATACLK,        1       ;SHIFT BCC ONCE
4124  023770  005200                          INC     R0              ;BUMP SHIFT COUNT
4125  023772  004537  034344                  JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
4126  023776  000001                          1                       ;ONE SHIFT
4127  024000  000000          73$:    0                               ;DATA CHARACTER
4128  024002  000000          74$:    0                               ;OLD BCC
4129  024004  103405                          BCS     75$             ;BR IF SOFT BCC LSB IS SET
4130  024006  004737  034616                  JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
4131  024012  103006                          BCC     76$             ;BR IF HARD BCC LSB IS CLEAR
4132  024014  104013                          ERROR   13              ;ERROR, BCC LSB IS SET
4133  024016  000404                          BR      76$             ;CONTINUE
4134  024020  004737  034616          75$:    JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
4135  024024  103401                          BCS     76$             ;BR IF HARD BCC LSB IS SET
4136  024026  104017                          ERROR   17              ;ERROR, BCC LSB IS CLEAR
4137  024030                          76$:
4138  024030  006037  024000                  ROR     73$             ;SHIFT SOFT DATA
4139  024034  013737  034470  024002          MOV     CALBCC,74$      ;LOAD OLD SOFT BCC
4140  024042  022700  000010                  CMP     #10,R0          ;DONE YET?
4141  024046  001346                          BNE     72$             ;BR IF NOT DONE
4142  024050  104405                          SCOP1                   ;SCOPE SUBTEST (SW09=1)
4143  024052                          77$:
4144
4145
4146                                          ;*************************** TEST 46 ****************************
4147                                          ;*TEST OF CRC OPERATION
4148                                          ;*USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
4149                                          ;*377, VERIFY THE LSB OF THE BCC ON EACH SHIFT
4150                                          ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
4151                                          ;;************************************************************
4152
4153                                          ;  TEST 46
4154                                          ;--------------
4155                                          ;;************************************************************
4156  024052  000004          TST46:  SCOPE
4157  024054  012737  000046  001202          MOV     #46,$TSTNM              ; LOAD THE NO. OF THIS TEST
```

```
4158  024062  012737  024444  001442          MOV     #TST47,NEXT             ; POINT TO THE START OF NEXT TEST.
4159  024070  012737  024124  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
4160                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
4161  024076  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4162  024100  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
4163  024104  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4164  024106  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
4165  024110  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
4166  024114  005037  035250                  CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
4167  024120  012711  004000                  MOV     #BIT11,(R1)     ;SET LU LOOP
4168  024124  004737  035032          64$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
4169  024130  005000                          CLR     R0              ;START SHIFT COUNTER AT ZERO
4170  024132  012737  102010  034466          MOV     #CRC.CCITT,XPOLY;LOAD POLYNOMIAL FOR SOFTWARE BCC
4171  024140  012737  000377  024224          MOV     #377,66$;       ;LOAD CHAR FOR SOFTWARE BCC
4172  024146  005037  024226                  CLR     67$             ;CLEAR OLD SOFTWARE BCC
4173  024152  005137  024226                  COM     67$             ;START AT -1
4174  024156  004737  034472                  JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
4175  024162  000377                          377                     ;AND THE CHARACTER 377
4176  024164  104413  000021                  DATACLK,        21      ;GET TRANSMITTER ACTIVE
4177  024170  005037  035250                  CLR     BITCON          ;CLEAR BIT COUNTER
4178  024174  005037  024210                  CLR     60$
4179  024200  104413  000001          65$:    DATACLK,        1       ;SHIFT BCC ONCE
4180  024204  004537  035132                  JSR     R5,STFFCK       ;CHECK FOR STUFFING ZEROS
4181  024210  000000          60$:    0                               ;CHARACTER
4182  024212  000001                          1                       ;SHIFT COUNT
4183  024214  005200                          INC     R0              ;BUMP SHIFT COUNT
4184  024216  004537  034344                  JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
4185  024222  000001                          1                       ;ONE SHIFT
4186  024224  000000          66$:    0                               ;DATA CHARACTER
4187  024226  000000          67$:    0                               ;OLD BCC
4188  024230  103405                          BCS     68$             ;BR IF SOFT BCC LSB IS SET
4189  024232  004737  034604                  JSR     PC,GETQ0        ;GET HARDWARE TRANSMITTER BCC LSB
4190  024236  103006                          BCC     69$             ;BR IF HARD BCC LSB IS CLEAR
4191  024240  104012                          ERROR   12              ;ERROR, BCC LSB IS SET
4192  024242  000404                          BR      69$             ;CONTINUE
4193  024244  004737  034604          68$:    JSR     PC,GETQ0        ;GET HARDWARE TRANSMITTER BCC LSB
4194  024250  103401                          BCS     69$             ;BR IF HARD BCC LSB IS SET
4195  024252  104016                          ERROR   16              ;ERROR, HARD BCC LSB IS CLEAR
4196  024254                          69$:
4197  024254  013737  024224  024210          MOV     66$,60$
4198  024262  006037  024224                  ROR     66$             ;SHIFT SOFT DATA
4199  024266  013737  034470  024226          MOV     CALBCC,67$      ;LOAD OLD SOFT BCC
4200  024274  022700  000010                  CMP     #10,R0          ;DONE YET?
4201  024300  001337                          BNE     65$             ;BR IF NOT DONE
4202  024302  104405                          SCOP1                   ;SCOPE SUBTEST (SW09=1)
4203  024304  012737  024312  001444          MOV     #71$,LOCK       ;NEW SCOPE1
4204  024312  004737  035032          71$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
4205  024316  005000                          CLR     R0              ;START SHIFT COUNTER AT ZERO
4206  024320  012737  102010  034466          MOV     #CRC.CCITT,XPOLY;LOAD POLYNOMIAL FOR SOFTWARE BCC
4207  024326  012737  000377  024372          MOV     #377,73$;       ;LOAD CHAR FOR SOFTWARE BCC
4208  024334  005037  024374                  CLR     74$             ;CLEAR OLD SOFTWARE BCC
4209  024340  005137  024374                  COM     74$             ;START AT -1
4210  024344  004737  034472                  JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
4211  024350  000377                          377                     ;AND THE CHARACTER 377
4212  024352  104413  000033                  DATACLK,        33      ;GET RECEIVER ACTIVE
4213  024356  104413  000001          72$:    DATACLK,        1       ;SHIFT BCC ONCE
```

```
4214  024362  005200                  INC     R0              ;BUMP SHIFT COUNT
4215  024364  004537  034344          JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
4216  024370  000001                  1                       ;ONE SHIFT
4217  024372  000000          73$:    0                       ;DATA CHARACTER
4218  024374  000000          74$:    0                       ;OLD BCC
4219  024376  103405                  BCS     75$             ;BR IF SOFT BCC LSB IS SET
4220  024400  004737  034616          JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
4221  024404  103006                  BCC     76$             ;BR IF HARD BCC LSB IS CLEAR
4222  024406  104013                  ERROR   13              ;ERROR, BCC LSB IS SET
4223  024410  000404                  BR      76$             ;CONTINUE
4224  024412  004737  034616  75$:    JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
4225  024416  103401                  BCS     76$             ;BR IF HARD BCC LSB IS SET
4226  024420  104017                  ERROR   17              ;ERROR, BCC LSB IS CLEAR
4227  024422                  76$:
4228  024422  006037  024372          ROR     73$             ;SHIFT SOFT DATA
4229  024426  013737  034470  024374  MOV     CALBCC,74$      ;LOAD OLD SOFT BCC
4230  024434  022700  000010          CMP     #10,R0          ;DONE YET?
4231  024440  001346                  BNE     72$             ;BR IF NOT DONE
4232  024442  104405                  SCOP1                   ;SCOPE SUBTEST (SW09=1)
4233  024444                  77$:
4234
4235
4236                                  ;************************** TEST 47 **************************
4237                                  ;*TEST OF CRC OPERATION
4238                                  ;#USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
4239                                  ;#125, VERIFY THE LSB OF THE BCC ON EACH SHIFT
4240                                  ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
4241                                  ;;**********************************************************
4242
4243                                  ;  TEST 47
4244                                  ;--------------
4245                          ;;**********************************************************
4246  024444  000004          TST47:  SCOPE
4247  024446  012737  000047  001202  MOV     #47,$TSTNM              ; LOAD THE NO. OF THIS TEST
4248  024454  012737  025010  001442  MOV     #TST50,NEXT             ; POINT TO THE START OF NEXT TEST.
4249  024462  012737  024516  001444  MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
4250                                                          ;R1 CONTAINS BASE KMC11 ADDRESS
4251  024470  104410                  MSTCLR                  ;MASTER CLEAR KMC11
4252  024472  005061  000004          CLR     4(R1)           ;CLEAR PORT4
4253  024476  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4254  024500  122117                  122117                  ;PUT LINE UNIT IN BITSTUFF MODE
4255  024502  004737  035032          JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
4256  024506  005037  035250          CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
4257  024512  012711  004000          MOV     #BIT11,(R1)     ;SET LU LOOP
4258  024516  004737  035032  64$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
4259  024522  005000                  CLR     R0              ;START SHIFT COUNTER AT ZERO
4260  024524  012737  102010  034466  MOV     #CRC.CCITT,XPOLY;LOAD POLYNOMIAL FOR SOFTWARE BCC
4261  024532  012737  000125  024576  MOV     #125,66$;       ;LOAD CHAR FOR SOFTWARE BCC
4262  024540  005037  024600          CLR     67$             ;CLEAR OLD SOFTWARE BCC
4263  024544  005137  024600          COM     67$             ;START AT -1
4264  024550  004737  034472          JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
4265  024554  000125                  125                     ;AND THE CHARACTER 125
4266  024556  104413  000021          DATACLK,        21      ;GET TRANSMITTER ACTIVE
4267  024562  104413  000001  65$:    DATACLK,        1       ;SHIFT BCC ONCE
4268  024566  005200                  INC     R0              ;BUMP SHIFT COUNT
4269  024570  004537  034344          JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
```

```
4270  024574  000001                          1                       ;ONE SHIFT
4271  024576  000000                  66$:    0                       ;DATA CHARACTER
4272  024600  000000                  67$:    0                       ;OLD BCC
4273  024602  103405                          BCS     68$             ;BR IF SOFT BCC LSB IS SET
4274  024604  004737  034604                  JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
4275  024610  103006                          BCC     69$             ;BR IF HARD BCC LSB IS CLEAR
4276  024612  104012                          ERROR   12              ;ERROR, BCC LSB IS SET
4277  024614  000404                          BR      69$             ;CONTINUE
4278  024616  004737  034604          68$:    JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
4279  024622  103401                          BCS     69$             ;BR IF HARD BCC LSB IS SET
4280  024624  104016                          ERROR   16              ;ERROR, HARD BCC LSB IS CLEAR
4281  024626                          69$:
4282  024626  006037  024576                  ROR     66$             ;SHIFT SOFT DATA
4283  024632  013737  034470  024600          MOV     CALBCC,67$      ;LOAD OLD SOFT BCC
4284  024640  022700  000010                  CMP     #10,R0          ;DONE YET?
4285  024644  001346                          BNE     65$             ;BR IF NOT DONE
4286  024646  104405                          SCOP1                   ;SCOPE SUBTEST (SW09=1)
4287  024650  012737  024656  001444          MOV     #71$,LOCK       ;NEW SCOPE1
4288  024656  004737  035032          71$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
4289  024662  005000                          CLR     R0              ;START SHIFT COUNTER AT ZERO
4290  024664  012737  102010  034466          MOV     #CRC.CCITT,XPOLY;LOAD POLYNOMIAL FOR SOFTWARE BCC
4291  024672  012737  000125  024736          MOV     #125,73$;       ;LOAD CHAR FOR SOFTWARE BCC
4292  024700  005037  024740                  CLR     74$             ;CLEAR OLD SOFTWARE BCC
4293  024704  005137  024740                  COM     74$             ;START AT -1
4294  024710  004737  034472                  JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
4295  024714  000125                          125                     ;AND THE CHARACTER 125
4296  024716  104413  000032                  DATACLK,        32      ;GET RECEIVER ACTIVE
4297  024722  104413  000001          72$:    DATACLK,        1       ;SHIFT BCC ONCE
4298  024726  005200                          INC     R0              ;BUMP SHIFT COUNT
4299  024730  004537  034344                  JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
4300  024734  000001                          1                       ;ONE SHIFT
4301  024736  000000                  73$:    0                       ;DATA CHARACTER
4302  024740  000000                  74$:    0                       ;OLD BCC
4303  024742  103405                          BCS     75$             ;BR IF SOFT BCC LSB IS SET
4304  024744  004737  034616                  JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
4305  024750  103006                          BCC     76$             ;BR IF HARD BCC LSB IS CLEAR
4306  024752  104013                          ERROR   13              ;ERROR, BCC LSB IS SET
4307  024754  000404                          BR      76$             ;CONTINUE
4308  024756  004737  034616          75$:    JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
4309  024762  103401                          BCS     76$             ;BR IF HARD BCC LSB IS SET
4310  024764  104017                          ERROR   17              ;ERROR, BCC LSB IS CLEAR
4311  024766                          76$:
4312  024766  006037  024736                  ROR     73$             ;SHIFT SOFT DATA
4313  024772  013737  034470  024740          MOV     CALBCC,74$      ;LOAD OLD SOFT BCC
4314  025000  022700  000010                  CMP     #10,R0          ;DONE YET?
4315  025004  001346                          BNE     72$             ;BR IF NOT DONE
4316  025006  104405                          SCOP1                   ;SCOPE SUBTEST (SW09=1)
4317  025010                          77$:
4318
4319
4320                                  ;************************** TEST 50 **************************
4321                                  ;*TEST OF CRC OPERATION
4322                                  ;*USING THE CRC.CCITT POLYNOMIAL, SINGLE CLOCK THE CHARACTER
4323                                  ;*252, VERIFY THE LSB OF THE BCC ON EACH SHIFT
4324                                  ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
4325                                  ;:************************************************************
```

```
                                        ;  TEST 50
                                        ;--------------
                                        ;;*****************************************************************
025010  000004                  TST50:  SCOPE
025012  012737  000050  001202          MOV     #50,$TSTNM              ; LOAD THE NO. OF THIS TEST
025020  012737  025354  001442          MOV     #TST51,NEXT             ; POINT TO THE START OF NEXT TEST.
025026  012737  025062  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
025034  104410                          MSTCLR                  ;MASTER CLEAR KMC11
025036  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
025042  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025044  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
025046  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
025052  005037  035250                  CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
025056  012711  004000                  MOV     #BIT11,(R1)     ;SET LU LOOP
025062  004737  035032          64$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
025066  005000                  64$:    CLR     R0              ;START SHIFT COUNTER AT ZERO
025070  012737  102010  034466          MOV     #CRC.CCITT,XPOLY;LOAD POLYNOMIAL FOR SOFTWARE BCC
025076  012737  000252  025142          MOV     #252,66$;       ;LOAD CHAR FOR SOFTWARE BCC
025104  005037  025144                  CLR     67$             ;CLEAR OLD SOFTWARE BCC
025110  005137  025144                  COM     67$             ;START AT -1
025114  004737  034472                  JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
025120  000252                          252                     ;AND THE CHARACTER 252
025122  104413  000021                  DATACLK,        21      ;GET TRANSMITTER ACTIVE
025126  104413  000001          65$:    DATACLK,        1       ;SHIFT BCC ONCE
025132  005200                  65$:    INC     R0              ;BUMP SHIFT COUNT
025134  004537  034344                  JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
025140  000001                          1                       ;ONE SHIFT
025142  000000                  66$:    0                       ;DATA CHARACTER
025144  000000                  67$:    0                       ;OLD BCC
025146  103405                          BCS     68$             ;BR IF SOFT BCC LSB IS SET
025150  004737  034604                  JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
025154  103006                          BCC     69$             ;BR IF HARD BCC LSB IS CLEAR
025156  104012                          ERROR   12              ;ERROR, BCC LSB IS SET
025160  000404                          BR      69$             ;CONTINUE
025162  004737  034604          68$:    JSR     PC,GETQO        ;GET HARDWARE TRANSMITTER BCC LSB
025166  103401                          BCS     69$             ;BR IF HARD BCC LSB IS SET
025170  104016                          ERROR   16              ;ERROR, HARD BCC LSB IS CLEAR
025172                          69$:
025172  006037  025142                  ROR     66$             ;SHIFT SOFT DATA
025176  013737  034470  025144          MOV     CALBCC,67$      ;LOAD OLD SOFT BCC
025204  022700  000010                  CMP     #10,R0          ;DONE YET?
025210  001346                          BNE     65$             ;BR IF NOT DONE
025212  104405                          SCOP1                   ;SCOPE SUBTEST (SW09=1)
025214  012737  025222  001444          MOV     #71$,LOCK       ;NEW SCOPE1
025222  004737  035032          71$:    JSR     PC,CLRIO        ;CLEAR BCC REGISTERS
025226  005000                  71$:    CLR     R0              ;START SHIFT COUNTER AT ZERO
025230  012737  102010  034466          MOV     #CRC.CCITT,XPOLY;LOAD POLYNOMIAL FOR SOFTWARE BCC
025236  012737  000252  025302          MOV     #252,73$;       ;LOAD CHAR FOR SOFTWARE BCC
025244  005037  025304                  CLR     74$             ;CLEAR OLD SOFTWARE BCC
025250  005137  025304                  COM     74$             ;START AT -1
025254  004737  034472                  JSR     PC,BCCLD        ;LOAD OUT SILO WITH 2 SYNCS
025260  000252                          252                     ;AND THE CHARACTER 252
025262  104413  000032                  DATACLK,        32      ;GET RECEIVER ACTIVE
025266  104413  000001          72$:    DATACLK,        1       ;SHIFT BCC ONCE
```

```
4382  025272  005200                        INC     R0              ;BUMP SHIFT COUNT
4383  025274  004537  034344                JSR     R5,SIMBCC       ;CALCULATE SOFTWARE BCC LSB
4384  025300  000001                        1                       ;ONE SHIFT
4385  025302  000000                73$:    0                       ;DATA CHARACTER
4386  025304  000000                74$:    0                       ;OLD BCC
4387  025306  103405                        BCS     75$             ;BR IF SOFT BCC LSB IS SET
4388  025310  004737  034616                JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
4389  025314  103006                        BCC     76$             ;BR IF HARD BCC LSB IS CLEAR
4390  025316  104013                        ERROR   13              ;ERROR, BCC LSB IS SET
4391  025320  000404                        BR      76$             ;CONTINUE
4392  025322  004737  034616        75$:    JSR     PC,GETQI        ;GET HARDWARE RECEIVER BCC LSB
4393  025326  103401                        BCS     76$             ;BR IF HARD BCC LSB IS SET
4394  025330  104017                        ERROR   17              ;ERROR, BCC LSB IS CLEAR
4395  025332                        76$:
4396  025332  006037  025302                ROR     73$             ;SHIFT SOFT DATA
4397  025336  013737  034470  025304        MOV     CALBCC,74$      ;LOAD OLD SOFT BCC
4398  025344  022700  000010                CMP     #10,R0          ;DONE YET?
4399  025350  001346                        BNE     72$             ;BR IF NOT DONE
4400  025352  104405                        SCOP1                   ;SCOPE SUBTEST (SW09=1)
4401  025354                        77$:
4402
4403
4404                                  ;*************************** TEST 51 ***************************
4405                                  ;*TRANSMITTER CRC TEST
4406                                  ;*USING THE CRC.CCITT POLYNOMINAL, SINGLE CLOCK A BINARY
4407                                  ;*COUNT PATTERN, VERIFY THE LSB OF THE TRANSMITTER BCC ON EACH SHIFT
4408                                  ;:*************************************************************
4409
4410                                  ;  TEST 51
4411                                  ;-------------
4412                                  ;;*************************************************************
4413  025354  000004        TST51:    SCOPE
4414  025356  012737  000051  001202        MOV     #51,$TSTNM              ; LOAD THE NO. OF THIS TEST
4415  025364  012737  025676  001442        MOV     #TST52,NEXT             ; POINT TO THE START OF NEXT TEST.
4416                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
4417  025372  104410                        MSTCLR                  ;MASTER CLEAR KMC11
4418  025374  005061  000004                CLR     4(R1)           ;CLEAR PORT4
4419  025400  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4420  025402  122117                        122117                  ;PUT LINE UNIT IN BITSTUFF MODE
4421  025404  004737  035032                JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
4422  025410  005037  035250                CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
4423  025414  012711  004000                MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
4424  025420  005003                        CLR     R3              ;ZERO BIT COUNT
4425  025422  005004                        CLR     R4              ;R4 CONTAINS CHAR TO BE LOADED IN SILO
4426  025424  005005                        CLR     R5              ;R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
4427  025426  005037  025554                CLR     4$              ;CLEAR SOFT BCC
4428  025432  005137  025554                COM     4$              ;START AT -1
4429  025436  012737  102010  034466        MOV     #CRC.CCITT,XPOLY        ;LOAD POLYNOMINAL
4430  025444  004737  034634                JSR     PC,SYNLD        ;LOAD SILO WITH 2 SYNCS, SOM SET
4431  025450  010461  000004                MOV     R4,4(R1)        ;PORT4<-CHAR
4432  025454  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4433  025456  122110                        122110                  ;LOAD OUT DATA
4434  025460  005204                        INC     R4              ;INCREMENT TO NEXT CHARACTER
4435  025462  010461  000004                MOV     R4,4(R1)        ;PORT4<-CHAR
4436  025466  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4437  025470  122110                        122110                  ;LOAD OUT DATA
```

```
 4438  025472  005204                        INC     R4              ;INCREMENT TO NEXT CHARACTER
 4439  025474  010461  000004                MOV     R4,4(R1)        ;PORT4←CHAR
 4440  025500  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4441  025502  122110                        122110                  ;LOAD OUT DATA
 4442  025504  004737  033502                JSR     PC,OCOR         ;WAIT FOR OCOR
 4443  025510  104413  000021                DATACLK,21              ;CLOCK DATA
 4444  025514  010537  025540                MOV     R5,10$          ;START WITH ZERO
 4445  025520  012700  000001                MOV     #1,R0           ;START COUNT AT 1
 4446  025524  010537  025552        1$:     MOV     R5,3$           ;LOAD CHAR FOR SOFT CRC
 4447  025530  104413  000001        2$:     DATACLK,1               ;SHIFT BCC ONCE
 4448  025534  004537  035132                JSR     R5,STFFCK       ;CHECK BIT STUFFING
 4449  025540  000000                10$:    0                       ;CHARACTER
 4450  025542  000001                        1                       ;SHIFT COUNT
 4451  025544  004537  034344                JSR     R5,SIMBCC       ;CALCULATE SOFT BCC
 4452  025550  000001                        1                       ;SOFT SHIFT COUNT
 4453  025552  000000                3$:     0                       ;SOFT CHARACTER
 4454  025554  000000                4$:     0                       ;OLD SOFT BCC
 4455  025556  103405                        BCS     5$              ;BR IF SOFT BCC LSB IS SET
 4456  025560  004737  034604                JSR     PC,GETQO                ;GET HARDWARE TRANSMITTER BCC LSB
 4457  025564  103006                        BCC     6$              ;BR IF OK (CLEARED)
 4458  025566  104020                        ERROR   20              ;ERROR, BCC LSB WAS SET
 4459  025570  000404                        BR      6$              ;CONTINUE WITH TEST
 4460  025572  004737  034604        5$:     JSR     PC,GETQO                ;GET HARDWARE TRANSMITTER BCC LSB
 4461  025576  103401                        BCS     6$              ;BR IF OK (SET)
 4462  025600  104021                        ERROR   21              ;ERROR, BCC LSB WAS CLEAR
 4463
 4464  025602                        6$:
 4465  025602  006037  025540                ROR     10$             ;SHIFT CHAR FOR STUFF CHECK
 4466  025606  005300                        DEC     R0              ;DEC STUFF CHECK SHIFT COUNT
 4467  025610  001004                        BNE     11$             ;BR IF NOT DONE THIS CHARACTER
 4468  025612  012700  000010                MOV     #10,R0          ;RESET BIT COUNT TO 10
 4469  025616  010537  025540                MOV     R5,10$          ;LOAD NEXT CHAR FOR STUFF CHECK
 4470  025622                        11$:
 4471  025622  006037  025552                ROR     3$              ;SHIFT SOFT DATA
 4472  025626  013737  034470  025554        MOV     CALBCC,4$       ;LOAD OLD SOFT BCC
 4473  025634  005203                        INC     R3              ;INCREMENT BIT COUNTER
 4474  025636  022703  000010                CMP     #10,R3          ;DONE A FULL CHARACTER YET?
 4475  025642  001332                        BNE     2$              ;BR IF NO
 4476  025644  005003                        CLR     R3              ;RESTART BIT COUNTER
 4477  025646  005204                        INC     R4              ;INCREMENT DATA FOR SILO
 4478  025650  022704  000400                CMP     #400,R4         ;DONE BINARY COUNT YET?
 4479  025654  003404                        BLE     9$              ;BR IF YES
 4480  025656  010461  000004                MOV     R4,4(R1)        ;PORT4←DATA
 4481  025662  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4482  025664  122110                        122110                  ;LOAD OUT DATA
 4483  025666  005205                9$:     INC     R5              ;INCREMENT DATA
 4484  025670  022705  000400                CMP     #400,R5         ;DONE BINARY PATTERN YET?
 4485  025674  001313                        BNE     1$              ;BR IF NO
 4486  025676                        7$:
 4487
 4488
 4489                                ;************************** TEST 52 **************************
 4490                                ;*RECEIVER CRC TEST
 4491                                ;*USING THE CRC.CCITT POLYNOMINAL, SINGLE CLOCK A BINARY
 4492                                ;*COUNT PATTERN, VERIFY THE LSB OF THE RECEIVER BCC ON EACH SHIFT
 4493                                ;;*************************************************************
```

```
                                        ;  TEST 52
                                        ;-------------
                                    ;;**********************************************************************
025676  000004                    TST52:  SCOPE
025700  012737  000052  001202            MOV     #52,$TSTNM              ; LOAD THE NO. OF THIS TEST
025706  012737  026234  001442            MOV     #TST53,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
025714  104410                            MSTCLR                          ;MASTER CLEAR KMC11
025716  005061  000004                    CLR     4(R1)                   ;CLEAR PORT4
025722  104412                            ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025724  122117                            122117                          ;PUT LINE UNIT IN BITSTUFF MODE
025726  004737  035032                    JSR     PC,CLRIO                ;DO THIS AFTER MODE IS SET
025732  005037  035250                    CLR     BITCON                  ;CONSECUTIVE 1'S COUNTER INIT TO 0
025736  012711  004000                    MOV     #BIT11,(R1)             ;SET LINE UNIT LOOP
025742  005003                            CLR     R3                      ;ZERO BIT COUNT
025744  005004                            CLR     R4                      ;R4 CONTAINS CHAR TO BE LOADED IN SILO
025746  005005                            CLR     R5                      ;R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
025750  005037  026102                    CLR     4$                      ;CLEAR SOFT BCC
025754  005137  026102                    COM     4$                      ;START AT -1
025760  012737  102010  034466            MOV     #CRC.CCITT,XPOLY                ;LOAD POLYNOMINAL
025766  004737  034634                    JSR     PC,SYNLD                ;LOAD SILO WITH 2 SYNCS, SOM SET
025772  010461  000004                    MOV     R4,4(R1)                ;PORT4<-CHAR
025776  104412                            ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026000  122110                            122110                          ;LOAD OUT DATA
026002  005204                            INC     R4                      ;INCREMENT TO NEXT CHARACTER
026004  010461  000004                    MOV     R4,4(R1)                ;PORT4<-CHAR
026010  104412                            ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026012  122110                            122110                          ;LOAD OUT DATA
026014  005204                            INC     R4                      ;INCREMENT TO NEXT CHARACTER
026016  010461  000004                    MOV     R4,4(R1)                ;PORT4<-CHAR
026022  104412                            ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026024  122110                            122110                          ;LOAD OUT DATA
026026  004737  033502                    JSR     PC,OCOR                 ;WAIT FOR OCOR
026032  104413  000032                    DATACLK,32                      ;CLOCK DATA
026036  010537  026066                    MOV     R5,10$                  ;START WITH ZERO
026042  005237  026066                    INC     10$                     ;TRANSMITTER IS ONE CHAR AHEAD
026046  012700  000010                    MOV     #10,R0                  ;R0 = CHAR COUNT
026052  010537  026100            1$:     MOV     R5,3$                   ;LOAD CHAR FOR SOFT CRC
026056  104413  000001            2$:     DATACLK,1                       ;SHIFT BCC ONCE
026062  004537  035132                    JSR     R5,STFFCK               ;CHECK BIT STUFFING
026066  000000                    10$:    0                               ;CHARACTER
026070  000001                            1                               ;SHIFT COUNT
026072  004537  034344                    JSR     R5,SIMBCC               ;CALCULATE SOFT BCC
026076  000001                            1                               ;SOFT SHIFT COUNT
026100  000000                    3$:     0                               ;SOFT CHARACTER
026102  000000                    4$:     0                               ;OLD SOFT BCC
026104  103405                            BCS     5$                      ;BR IF SOFT BCC LSB IS SET
026106  004737  034616                    JSR     PC,GETQI                        ;GET HARDWARE RECEIVER BCC LSB
026112  103006                            BCC     6$                      ;BR IF OK (CLEARED)
026114  104022                            ERROR   22                      ;ERROR, BCC LSB WAS SET
026116  000404                            BR      6$                      ;CONTINUE WITH TEST
026120  004737  034616            5$:     JSR     PC,GETQI                        ;GET HARDWARE RECEIVER BCC LSB
026124  103401                            BCS     6$                      ;BR IF OK (SET)
026126  104023                            ERROR   23                      ;ERROR, BCC LSB WAS CLEAR
```

```
4550  026130                          6$:
4551  026130  006037  026066                  ROR     10$             ;SHIFT CHAR FOR STUFF CHECK
4552  026134  005300                          DEC     R0              ;DEC STUFF CHECK SHIFT COUNT
4553  026136  001010                          BNE     11$             ;BR IF NOT DONE THIS CHARACTER
4554  026140  012700  000010                  MOV     #10,R0          ;RESET BIT COUNT TO 10
4555  026144  010537  026066                  MOV     R5,10$          ;LOAD NEXT CHAR FOR STUFF CHECK
4556  026150  005237  026066                  INC     10$             ;TRANSMITTER IS 2 CHAR AHEAD
4557  026154  005237  026066                  INC     10$             ;
4558  026160                          11$:
4559  026160  006037  026100                  ROR     3$              ;SHIFT SOFT DATA
4560  026164  013737  034470  026102          MOV     CALBCC,4$       ;LOAD OLD SOFT BCC
4561  026172  005203                          INC     R3              ;INCREMENT BIT COUNTER
4562  026174  022703  000010                  CMP     #10,R3          ;DONE A FULL CHARACTER YET?
4563  026200  001326                          BNE     2$              ;BR IF NO
4564  026202  005003                          CLR     R3              ;RESTART BIT COUNTER
4565  026204  005204                          INC     R4              ;INCREMENT DATA FOR SILO
4566  026206  022704  000400                  CMP     #400,R4         ;DONE BINARY COUNT YET?
4567  026212  003404                          BLE     9$              ;BR IF YES
4568  026214  010461  000004                  MOV     R4,4(R1)        ;PORT4<-DATA
4569  026220  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4570  026222  122110                          122110                  ;LOAD OUT DATA
4571  026224  005205                  9$:     INC     R5              ;INCREMENT DATA
4572  026226  022705  000400                  CMP     #400,R5         ;DONE BINARY PATTERN YET?
4573  026232  001307                          BNE     1$              ;BR IF NO
4574  026234                          7$:
4575
4576
4577                                          ;*************************** TEST 53 ****************************
4578                                          ;*TRANSMITTER BITSTUFF CRC TEST
4579                                          ;*THIS TEST TRANSMITS A FOUR CHARACTER MESSAGE WITH CRC
4580                                          ;*BOTH DATA AND THE BCC ARE VERIFIED IN THE BIT
4581                                          ;*WINDOW. THE FOUR CHARACTERS ARE 0,125,252,377
4582                                          ;*THE TRANSMITTER IS CHECKED FOR GOING TO A MARK STATE AFTER THE BCC
4583                                          ;;************************************************************
4584
4585                                          ;  TEST 53
4586                                          ;--------------
4587                                  ;;******************************************************************
4588  026234  000004                  TST53:  SCOPE
4589  026236  012737  000053  001202          MOV     #53,$TSTNM              ; LOAD THE NO. OF THIS TEST
4590  026244  012737  026736  001442          MOV     #TST54,NEXT             ; POINT TO THE START OF NEXT TEST.
4591                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4592  026252  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4593  026254  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
4594  026260  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4595  026262  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
4596  026264  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
4597  026270  005037  035250                  CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
4598
4599                                          ;LOAD OUT DATA SILO
4600
4601  026274  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
4602  026300  012704  035252                  MOV     #MESDAT,R4      ;LOAD POINTER TO DATA
4603  026304  005037  026414                  CLR     10$             ;CLEAR SOFT BCC
4604  026310  005137  026414                  COM     10$             ;START AT -1
4605  026314  012700  000004                  MOV     #4,R0           ;LOAD CHARACTER COUNT
```

```
4606  026320  004737  034634                  JSR     PC,SYNLD        ;LOAD 2 FLAG CHARACTERS IN OUT SILO
4607  026324  004737  033634                  JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
4608  026330  004537  034770                  JSR     R5,MESLD        ;LOAD SILO WITH 4 CHAR MESS
4609  026334  035252                          MESDAT                  ;ADDRESS OF MESSAGE
4610  026336  000004                          4                       ;NUMBER OF CHARACTERS
4611  026340  004737  034744                  JSR     PC,EOM          ;LOAD GARBAGE CHARACTER, WITH EOM SET
4612  026344  004737  034744                  JSR     PC,EOM
4613  026350  004737  033502                  JSR     PC,OCOR         ;WAIT FOR OCOR
4614  026354  005003                          CLR     R3              ;CLEAR BIT COUNTER
4615  026356  104413  000022                  DATACLK,22              ;CLOCK DATA
4616  026362  112405                  12$:    MOVB    (R4)+,R5        ;LOAD R5 WITH CHAR
4617  026364  010502                          MOV     R5,R2           ;LOAD R2 WITH CHAR
4618
4619                                          ;CHECK FIRST FOUR CHARACTER MESSAGE
4620                                          ;IN THE BIT WINDOW (0,125,252,377)
4621
4622  026366  010537  026462                  MOV     R5,71$          ;LOAD FOR STUFF CHECK
4623  026372  012737  102010  034466          MOV     #CRC.CCITT,XPOLY        ;LOAD POLYNOMIAL
4624  026400  010537  026412                  MOV     R5,67$          ;LOAD SOFT CHAR FOR BCC
4625  026404  004537  034344                  JSR     R5,SIMBCC       ;CALCULATE SOFT BCC
4626  026410  000010                          10                      ;SHIFT COUNT
4627  026412  000000                  67$:    0                       ;CHARACTER
4628  026414  000000                  10$:    0                       ;OLD BCC
4629  026416  013737  034470  026414          MOV     CALBCC,10$      ;LOAD SOFT BCC FOR NEXT SHIFT
4630  026424  104413  000001          64$:    DATACLK,        1       ;SHIFT DATA IN TO BIT WINDOW
4631  026430  106002                          RORB    R2              ;SHIFT SOFT DATA
4632  026432  103005                          BCC     65$             ;BR IF A SPACE
4633  026434  004737  033450                  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
4634  026440  103406                          BCS     66$             ;BR IF OK (MARK)
4635  026442  104006                          ERROR   6               ;ERROR, BIT WINDOW WAS A SPACE
4636  026444  000404                          BR      66$             ;CONTINUE
4637  026446  004737  033450          65$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
4638  026452  103001                          BCC     66$             ;BR IF OK (SPACE)
4639  026454  104006                          ERROR   6               ;ERROR, BIT WINDOW WAS A MARK
4640  026456                          66$:
4641  026456  004537  035132                  JSR     R5,STFFCK
4642  026462  000000                  71$:    0
4643  026464  000001                          1
4644  026466  110237  026462                  MOVB    R2,71$          ;SHIFT FOR NEXT STUFF CHECK
4645  026472  005203                          INC     R3              ;BUMP BIT COUNTER
4646  026474  022703  000010                  CMP     #10,R3          ;DONE FULL 8 BITS YET
4647  026500  001351                          BNE     64$             ;BR IF NO
4648  026502  005003                          CLR     R3              ;CLEAR BIT COUNTER
4649  026504  005300                          DEC     R0              ;DEC CHARACTER COUNT
4650  026506  001325                          BNE     12$             ;BR IF NOT DONE YET
4651
4652                                          ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4653
4654  026510  005137  034470                  COM     CALBCC          ;ADJUST BCC FOR SDLC
4655  026514  013700  034470                  MOV     CALBCC,R0       ;PUT BCC IN R0
4656  026520  010037  026562                  MOV     R0,72$          ;LOAD BCC FOR STUFF CHECK
4657  026524  104413  000001          68$:    DATACLK,1               ;SHIFT HARDWARE BCC
4658  026530  006000                          ROR     R0              ;SHIFT SOFT BCC
4659  026532  103005                          BCC     69$             ;BR IF CARRY CLEAR
4660  026534  004737  033450                  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
4661  026540  103406                          BCS     70$             ;BR IF OK (MARK)
```

```
 4662  026542  104014                        ERROR   14              ;ERROR, CRC WRONG (SPACE)
 4663  026544  000404                        BR      70$             ;CONTINUE
 4664  026546  004737  033450        69$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4665  026552  103001                        BCC     70$             ;BR IF OK (SPACE)
 4666  026554  104014                        ERROR   14              ;ERROR, CRC WRONG (MARK)
 4667  026556                        70$:
 4668  026556  004537  035132                JSR     R5,STFFCK       ;CHECK BCC CHAR FOR ZERO STUFFS
 4669  026562  000000                72$:    0                       ;CHARACTER
 4670  026564  000001                        1                       ;SHIFT COUNT
 4671  026566  010037  026562                MOV     R0,72$          ;SHIFT SOFTBCC ONCE
 4672  026572  005203                        INC     R3              ;BUMP BIT COUNTER
 4673  026574  022703  000020                CMP     #20,R3          ;FINISHED BCC YET?
 4674  026600  001351                        BNE     68$             ;BR IF NO
 4675  026602  005003                        CLR     R3              ;CLEAR BIT COUNTER
 4676
 4677                                        ;CHECK FOR FLAG TO FOLLOW BCC
 4678
 4679  026604  012737  000176  001302        MOV     #↑B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
 4680  026612  104413  000001        73$:    DATACLK,        1       ;CLOCK FLAG ONCE
 4681  026616  106037  001302                RORB    $TMP2           ;SHIFT SOFT FLAG
 4682  026622  103405                        BCS     74$             ;BR IF BIT IS MARK
 4683  026624  004737  033450                JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4684  026630  103006                        BCC     75$             ;BR IF OK
 4685  026632  104026                        ERROR   26              ;ERROR IN FLAG CHAR
 4686  026634  000404                        BR      75$
 4687  026636  004737  033450        74$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4688  026642  103401                        BCS     75$             ;BR IF OK
 4689  026644  104026                        ERROR   26              ;ERROR IN FLAG CHAR
 4690  026646  005203                75$:    INC     R3              ;INC BIT COUNT
 4691  026650  022703  000010                CMP     #10,R3          ;FLAG DONE YET?
 4692  026654  001356                        BNE     73$             ;BR IF NO
 4693  026656  005003                        CLR     R3              ;CLEAR BIT COUNT
 4694
 4695                                        ;CHECK TO SEE IF TRANSMITTER IS MARKING
 4696
 4697  026660  104413  000001        2$:     DATACLK,        1       ;CLOCK TRANSMITTER
 4698  026664  004737  033450                JSR     PC,GETSI        ;LOOK AT WINDOW
 4699  026670  103401                        BCS     3$              ;IT SHOULD BE MARKING
 4700  026672  104024                        ERROR   24              ;ERROR, BIT WAS A SPACE
 4701  026674  005203                3$:     INC     R3              ;BUMP BIT COUNTER
 4702  026676  022703  000007                CMP     #7,R3           ;DONE YET
 4703  026702  001366                        BNE     2$              ;BR IF NO
 4704  026704  104413  000010                DATACLK,        10      ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
 4705  026710  005003                        CLR     R3              ;CLEAR BIT COUNTER
 4706  026712  104413  000001        4$:     DATACLK,        1       ;SHIFT OUT NEXT BIT
 4707  026716  004737  033450                JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4708  026722  103401                        BCS     .+4             ;BR IF IT IS A MARK
 4709  026724  104024                        ERROR   24              ;ERROR, TRANSMITTER IS NOT MARKING
 4710  026726  005203                        INC     R3              ;INC BIT COUNT
 4711  026730  022703  000020                CMP     #20,R3          ;DONE YET?
 4712  026734  001366                        BNE     4$              ;BR IF NO
 4713  026736                        5$:
 4714
 4715
 4716                                        ;***************************** TEST 54 *****************************
 4717                                        ;*RECEIVER BITSTUFF CRC TEST
```

```
4718                                    ;*THIS TEST CLOCKS A FOUR CHARACTER MESSAGE WITH BCC
4719                                    ;*AND VERIFYS CORRECT DATA RECEPTION AND BCC MATCH
4720                                    ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4721                                    ;;*****************************************************************
4722
4723                                    ;  TEST 54
4724                                    ;--------------
4725                                    ;;*****************************************************************
4726  026736  000004                TST54:  SCOPE
4727  026740  012737  000054  001202        MOV     #54,$TSTNM              ; LOAD THE NO. OF THIS TEST
4728  026746  012737  027160  001442        MOV     #TST55,NEXT             ; POINT TO THE START OF NEXT TEST.
4729                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4730  026754  104410                        MSTCLR                  ;MASTER CLEAR KMC11
4731  026756  005061  000004                CLR     4(R1)           ;CLEAR PORT4
4732  026762  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4733  026764  122117                        122117                  ;PUT LINE UNIT IN BITSTUFF MODE
4734  026766  004737  035032                JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
4735  026772  012711  004000                MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
4736  026776  012702  035252                MOV     #MESDAT,R2      ;LOAD POINTER TO DATA
4737  027002  012700  000004                MOV     #4,R0           ;LOAD CHARACTER COUNT
4738  027006  004737  034634                JSR     PC,SYNLD        ;LOAD 2 FLAG CHARACTERS IN OUT SILO
4739  027012  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
4740  027016  004537  034770                JSR     R5,MESLD        ;LOAD SILO WITH 4 CHAR MESS
4741  027022  035252                        MESDAT                  ;ADDRESS OF MESSAGE
4742  027024  000004                        4                       ;NUMBER OF CHARACTERS
4743  027026  004737  034744                JSR     PC,EOM          ;LOAD GARBAGE CHARACTER, WITH EOM SET
4744  027032  004737  034744                JSR     PC,EOM
4745  027036  004737  033502                JSR     PC,OCOR         ;WAIT FOR OCOR
4746  027042  104413  000115                DATACLK,11$             ;CLOCK DATA
4747  027046  004737  034310          3$:   JSR     PC,INRDY        ;WAIT FOR INRDY
4748  027052  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4749  027054  021204                        021204                  ;GET IN DATA
4750  027056  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
4751  027062  112205                        MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
4752  027064  120504                        CMPB    R5,R4           ;COMPARE RECEIVED DATA
4753  027066  001401                        BEQ     1$              ;BR IF OK
4754  027070  104010                        ERROR   10              ;DATA ERROR
4755  027072  005300                1$:     DEC     R0              ;DEC CHARACTER COUNT
4756  027074  001364                        BNE     3$              ;BR IF NOT DONE YET
4757
4758                                    ;CHECK TO SEE THAT IN BCC MATCH IS SET
4759
4760  027076  004737  034310                JSR     PC,INRDY        ;WAIT FOR INRDY
4761  027102  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4762  027104  021204                        021204                  ;GET FIRST HALF OF CRC
4763  027106  116137  000004  001302        MOVB    4(R1),$TMP2     ;PUT IN $TMP2
4764  027114  042737  177400  001302        BIC     #177400,$TMP2   ;CLEAR HI BYTE
4765  027122  004737  034310                JSR     PC,INRDY        ;WAIT FOR INRDY
4766  027126  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4767  027130  021244                        021244
4768  027132  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
4769  027136  042704  000374                BIC     #374,R4         ;CLEAR UNWANTED BITS
4770  027142  012705  000003                MOV     #3,R5           ;PUT "EXPECTED" IN R5
4771  027146  120504                        CMPB    R5,R4           ;ARE IN BCC MATCH AND BLOCK END SET?
4772  027150  001401                        BEQ     25$
4773  027152  104042                        ERROR   42              ;IN BCC MATCH ERROR
```

```
4774  027154                          25$:
4775  027154  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4776  027156  021204                        021204                  ;GET LAST HALF
4777  027160                          2$:
4778
4779
4780                                  ;*************************** TEST 55 ****************************
4781                                  ;*BITSTUFF EOM FUNCTION TEST
4782                                  ;*THIS TEST LOADS OUT SILO WITH: 2 FLAGS,4 CHAR MESSAGE,EOM
4783                                  ;*4 CHARACTER MESS,EOM. THE DATA STREAM IS CHECKED TO BE
4784                                  ;*4 CHAR,BCC,FLAG,4 CHAR,BCC,FLAG,MARKS. THIS TEST VERIFYS THAT
4785                                  ;*THE CHARCTERS LOADED WITH EOM SET ARE LOST
4786                                  ;*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
4787                                  ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4788                                  ;*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
4789                                  ;;*************************************************************
4790
4791                                  ;  TEST 55
4792                                  ;--------------
4793                                  ;;*************************************************************
4794  027160  000004                  TST55:  SCOPE
4795  027162  012737  000055  001202          MOV     #55,$TSTNM              ; LOAD THE NO. OF THIS TEST
4796  027170  012737  030560  001442          MOV     #TST56,NEXT             ; POINT TO THE START OF NEXT TEST.
4797                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4798  027176  104410                        MSTCLR                  ;MASTER CLEAR KMC11
4799  027200  005061  000004                CLR     4(R1)           ;CLEAR PORT4
4800  027204  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4801  027206  122117                        122117                  ;PUT LINE UNIT IN BITSTUFF MODE
4802  027210  004737  035032                JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
4803  027214  005037  035250                CLR     BITCON          ;CONSECUTIVE 1'S COUNTER INIT TO 0
4804
4805                                  ;LOAD OUT DATA SILO
4806
4807  027220  012711  004000                MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
4808  027224  012704  035252                MOV     #MESDAT,R4      ;LOAD POINTER TO DATA
4809  027230  005037  027360                CLR     10$             ;CLEAR SOFT BCC
4810  027234  005137  027360                COM     10$             ;START AT -1
4811  027240  012700  000004                MOV     #4,R0           ;LOAD CHARACTER COUNT
4812  027244  004737  034634                JSR     PC,SYNLD        ;LOAD 2 FLAG CHARACTERS IN OUT SILO
4813  027250  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
4814  027254  004537  034770                JSR     R5,MESLD        ;LOAD SILO WITH 4 CHAR MESS
4815  027260  035252                        MESDAT                  ;ADDRESS OF MESSAGE
4816  027262  000004                        4                       ;NUMBER OF CHARACTERS
4817  027264  004737  034744                JSR     PC,EOM          ;LOAD GARBAGE CHARACTER, WITH EOM SET
4818  027270  004737  034744                JSR     PC,EOM
4819  027274  004537  034770                JSR     R5,MESLD        ;LOAD FOUR MORE CHARACTERS
4820  027300  035252                        MESDAT                  ;ADDRESS OF MESSAGE
4821  027302  000004                        4                       ;NUMBER OF CHACTERS
4822  027304  004737  034744                JSR     PC,EOM          ;SET EOM
4823  027310  004737  034744                JSR     PC,EOM          ;SET EOM
4824  027314  004737  033502                JSR     PC,OCOR         ;WAIT FOR OCOR
4825  027320  005003                        CLR     R3              ;CLEAR BIT COUNTER
4826  027322  104413  000022                DATACLK,22              ;CLOCK DATA
4827  027326  112405                  12$:  MOVB    (R4)+,R5        ;LOAD R5 WITH CHAR
4828  027330  010502                        MOV     R5,R2           ;LOAD R2 WITH CHAR
4829
```

```
4830                                        ;CHECK FIRST FOUR CHARACTER MESSAGE
4831                                        ;IN THE BIT WINDOW (0,125,252,377)
4832
4833  027332  010537  027426                MOV     R5,71$              ;LOAD FOR STUFF CHECK
4834  027336  012737  102010  034466        MOV     #CRC.CCITT,XPOLY            ;LOAD POLYNOMIAL
4835  027344  010537  027356                MOV     R5,67$              ;LOAD SOFT CHAR FOR BCC
4836  027350  004537  034344                JSR     R5,SIMBCC           ;CALCULATE SOFT BCC
4837  027354  000010                        10                          ;SHIFT COUNT
4838  027356  000000              67$:      0                           ;CHARACTER
4839  027360  000000              10$:      0                           ;OLD BCC
4840  027362  013737  034470  027360        MOV     CALBCC,10$          ;LOAD SOFT BCC FOR NEXT SHIFT
4841  027370  104413  000001        64$:    DATACLK,        1           ;SHIFT DATA IN TO BIT WINDOW
4842  027374  106002                        RORB    R2                  ;SHIFT SOFT DATA
4843  027376  103005                        BCC     65$                 ;BR IF A SPACE
4844  027400  004737  033450                JSR     PC,GETSI            ;LOOK AT BIT WINDOW
4845  027404  103406                        BCS     66$                 ;BR IF OK (MARK)
4846  027406  104006                        ERROR   6                   ;ERROR, BIT WINDOW WAS A SPACE
4847  027410  000404                        BR      66$                 ;CONTINUE
4848  027412  004737  033450        65$:    JSR     PC,GETSI            ;LOOK AT BIT WINDOW
4849  027416  103001                        BCC     66$                 ;BR IF OK (SPACE)
4850  027420  104006                        ERROR   6                   ;ERROR, BIT WINDOW WAS A MARK
4851  027422                        66$:
4852  027422  004537  035132                JSR     R5,STFFCK
4853  027426  000000              71$:      0
4854  027430  000001                        1
4855  027432  110237  027426                MOVB    R2,71$              ;SHIFT FOR NEXT STUFF CHECK
4856  027436  005203                        INC     R3                  ;BUMP BIT COUNTER
4857  027440  022703  000010                CMP     #10,R3              ;DONE FULL 8 BITS YET
4858  027444  001351                        BNE     64$                 ;BR IF NO
4859  027446  005003                        CLR     R3                  ;CLEAR BIT COUNTER
4860  027450  005300                        DEC     R0                  ;DEC CHARACTER COUNT
4861  027452  001325                        BNE     12$                 ;BR IF NOT DONE YET
4862
4863                                        ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4864
4865  027454  005137  034470                COM     CALBCC              ;ADJUST BCC FOR SDLC
4866  027460  013700  034470                MOV     CALBCC,R0           ;PUT BCC IN R0
4867  027464  010037  027526                MOV     R0,72$              ;LOAD BCC FOR STUFF CHECK
4868  027470  104413  000001        68$:    DATACLK,1                   ;SHIFT HARDWARE BCC
4869  027474  006000                        ROR     R0                  ;SHIFT SOFT BCC
4870  027476  103005                        BCC     69$                 ;BR IF CARRY CLEAR
4871  027500  004737  033450                JSR     PC,GETSI            ;LOOK AT BIT WINDOW
4872  027504  103406                        BCS     70$                 ;BR IF OK (MARK)
4873  027506  104014                        ERROR   14                  ;ERROR, CRC WRONG (SPACE)
4874  027510  000404                        BR      70$                 ;CONTINUE
4875  027512  004737  033450        69$:    JSR     PC,GETSI            ;LOOK AT BIT WINDOW
4876  027516  103001                        BCC     70$                 ;BR IF OK (SPACE)
4877  027520  104014                        ERROR   14                  ;ERROR, CRC WRONG (MARK)
4878  027522                        70$:
4879  027522  004537  035132                JSR     R5,STFFCK           ;CHECK BCC CHAR FOR ZERO STUFFS
4880  027526  000000              72$:      0                           ;CHARACTER
4881  027530  000001                        1                           ;SHIFT COUNT
4882  027532  010037  027526                MOV     R0,72$              ;SHIFT SOFTBCC ONCE
4883  027536  005203                        INC     R3                  ;BUMP BIT COUNTER
4884  027540  022703  000020                CMP     #20,R3              ;FINISHED BCC YET?
4885  027544  001351                        BNE     68$                 ;BR IF NO
```

```
                                        CLR     R3              ;CLEAR BIT COUNTER
                                        ;CHECK FOR FLAG TO FOLLOW BCC
027550  012737  000176  001302        MOV     #↑B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
027556  104413  000001          73$:  DATACLK,        1         ;CLOCK FLAG ONCE
027562  106037  001302              RORB    $TMP2             ;SHIFT SOFT FLAG
027566  103405                      BCS     74$               ;BR IF BIT IS MARK
027570  004737  033450              JSR     PC,GETSI          ;LOOK AT BIT WINDOW
027574  103006                      BCC     75$               ;BR IF OK
027576  104026                      ERROR   26                ;ERROR IN FLAG CHAR
027600  000404                      BR      75$
027602  004737  033450          74$:  JSR     PC,GETSI          ;LOOK AT BIT WINDOW
027606  103401                      BCS     75$               ;BR IF OK
027610  104026                      ERROR   26                ;ERROR IN FLAG CHAR
027612  005203          75$:        INC     R3                ;INC BIT COUNT
027614  022703  000010              CMP     #10,R3            ;FLAG DONE YET?
027620  001356                      BNE     73$               ;BR IF NO
027622  005003                      CLR     R3                ;CLEAR BIT COUNT
027624  012700  000004              MOV     #4,R0             ;RESET CHARACTER COUNTER
027630  012704  035252              MOV     #MESDAT,R4        ;LOAD MESSAGE POINTER
027634  005037  027676              CLR     11$               ;CLR SOFT BCC
027640  005137  027676              COM     11$               ;ADJUST TO -1 FOR SDLC
027644  112405          13$:        MOVB    (R4)+,R5          ;LOAD CHAR IN R5
027646  010502                      MOV     R5,R2             ;LOAD CHAR IN R2
                                        ;CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
027650  010537  027744              MOV     R5,83$            ;LOAD FOR STUFF CHECK
027654  012737  102010  034466      MOV     #CRC.CCITT,XPOLY        ;LOAD POLYNOMIAL
027662  010537  027674              MOV     R5,79$            ;LOAD SOFT CHAR FOR BCC
027666  004537  034344              JSR     R5,SIMBCC         ;CALCULATE SOFT BCC
027672  000010                      10                        ;SHIFT COUNT
027674  000000          79$:        0                         ;CHARACTER
027676  000000          11$:        0                         ;OLD BCC
027700  013737  034470  027676      MOV     CALBCC,11$        ;LOAD SOFT BCC FOR NEXT SHIFT
027706  104413  000001          76$:  DATACLK,        1         ;SHIFT DATA IN TO BIT WINDOW
027712  106002                      RORB    R2                ;SHIFT SOFT DATA
027714  103005                      BCC     77$               ;BR IF A SPACE
027716  004737  033450              JSR     PC,GETSI          ;LOOK AT BIT WINDOW
027722  103406                      BCS     78$               ;BR IF OK (MARK)
027724  104006                      ERROR   6                 ;ERROR, BIT WINDOW WAS A SPACE
027726  000404                      BR      78$               ;CONTINUE
027730  004737  033450          77$:  JSR     PC,GETSI          ;LOOK AT BIT WINDOW
027734  103001                      BCC     78$               ;BR IF OK (SPACE)
027736  104006                      ERROR   6                 ;ERROR, BIT WINDOW WAS A MARK
027740                          78$:
027740  004537  035132              JSR     R5,STFFCK
027744  000000          83$:        0
027746  000001                      1
027750  110237  027744              MOVB    R2,83$            ;SHIFT FOR NEXT STUFF CHECK
027754  005203                      INC     R3                ;BUMP BIT COUNTER
027756  022703  000010              CMP     #10,R3            ;DONE FULL 8 BITS YET
027762  001351                      BNE     76$               ;BR IF NO
027764  005003                      CLR     R3                ;CLEAR BIT COUNTER
027766  005300                      DEC     R0                ;DEC CHARACTER COUNT
```

```
 4942  027770  001325                        BNE     13$             ;BR IF NOT DONE YET
 4943
 4944                                        ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
 4945
 4946  027772  005137  034470                COM     CALBCC          ;ADJUST BCC FOR SDLC
 4947  027776  013700  034470                MOV     CALBCC,R0       ;PUT BCC IN R0
 4948  030002  010037  030044                MOV     R0,B4$          ;LOAD BCC FOR STUFF CHECK
 4949  030006  104413  000001        B0$:    DATACLK,1               ;SHIFT HARDWARE BCC
 4950  030012  006000                        ROR     R0              ;SHIFT SOFT BCC
 4951  030014  103005                        BCC     B1$             ;BR IF CARRY CLEAR
 4952  030016  004737  033450                JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4953  030022  103406                        BCS     B2$             ;BR IF OK (MARK)
 4954  030024  104014                        ERROR   14              ;ERROR, CRC WRONG (SPACE)
 4955  030026  000404                        BR      B2$             ;CONTINUE
 4956  030030  004737  033450        B1$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4957  030034  103001                        BCC     B2$             ;BR IF OK (SPACE)
 4958  030036  104014                        ERROR   14              ;ERROR, CRC WRONG (MARK)
 4959  030040                        B2$:
 4960  030040  004537  035132                JSR     R5,STFFCK       ;CHECK BCC CHAR FOR ZERO STUFFS
 4961  030044  000000        B4$:    0                               ;CHARACTER
 4962  030046  000001                        1                       ;SHIFT COUNT
 4963  030050  010037  030044                MOV     R0,B4$          ;SHIFT SOFTBCC ONCE
 4964  030054  005203                        INC     R3              ;BUMP BIT COUNTER
 4965  030056  022703  000020                CMP     #20,R3          ;FINISHED BCC YET?
 4966  030062  001351                        BNE     B0$             ;BR IF NO
 4967  030064  005003                        CLR     R3              ;CLEAR BIT COUNTER
 4968
 4969                                        ;CHECK FOR FLAG TO FOLLOW BCC
 4970
 4971  030066  012737  000176  001302        MOV     #^B<01111110>,STMP2 ;PUT FLAG CHARACTER IN STMP2
 4972  030074  104413  000001        B5$:    DATACLK,        1       ;CLOCK FLAG ONCE
 4973  030100  106037  001302                RORB    STMP2           ;SHIFT SOFT FLAG
 4974  030104  103405                        BCS     B6$             ;BR IF BIT IS MARK
 4975  030106  004737  033450                JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4976  030112  103006                        BCC     B7$             ;BR IF OK
 4977  030114  104026                        ERROR   26              ;ERROR IN FLAG CHAR
 4978  030116  000404                        BR      B7$
 4979  030120  004737  033450        B6$:    JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 4980  030124  103401                        BCS     B7$             ;BR IF OK
 4981  030126  104026                        ERROR   26              ;ERROR IN FLAG CHAR
 4982  030130  005203                B7$:    INC     R3              ;INC BIT COUNT
 4983  030132  022703  000010                CMP     #10,R3          ;FLAG DONE YET?
 4984  030136  001356                        BNE     B5$             ;BR IF NO
 4985  030140  005003                        CLR     R3              ;CLEAR BIT COUNT
 4986
 4987                                        ;CHECK TO SEE IF TRANSMITTER IS MARKING
 4988
 4989  030142  104413  000001        2$:     DATACLK,        1       ;CLOCK TRANSMITTER
 4990  030146  004737  033450                JSR     PC,GETSI        ;LOOK AT WINDOW
 4991  030152  103401                        BCS     3$              ;IT SHOULD BE MARKING
 4992  030154  104024                        ERROR   24              ;ERROR, BIT WAS A SPACE
 4993  030156  005203                3$:     INC     R3              ;BUMP BIT COUNTER
 4994  030160  022703  000007                CMP     #7,R3           ;DONE YET
 4995  030164  001366                        BNE     2$              ;BR IF NO
 4996  030166  104413  000010                DATACLK,        10      ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
 4997  030172  005003                        CLR     R3              ;CLEAR BIT COUNTER
```

```
 4998  030174  104413  000001          4$:     DATACLK,        1       ;SHIFT OUT NEXT BIT
 4999  030200  004737  033450                  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
 5000  030204  103401                          BCS     .+4             ;BR IF IT IS A MARK
 5001  030206  104024                          ERROR   24              ;ERROR, TRANSMITTER IS NOT MARKING
 5002  030210  005203                          INC     R3              ;INC BIT COUNT
 5003  030212  022703  000020                  CMP     #20,R3          ;DONE YET?
 5004  030216  001366                          BNE     4$              ;BR IF NO
 5005
 5006                                          ;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
 5007                                          ;WAS RECEIVED CORRECTLY (0,125,252,377)
 5008
 5009  030220  104413  000001                  DATACLK,        1       ;GET LAST BIT IN RECEIVER
 5010  030224  012703  000004                  MOV     #4,R3           ;R3=CHARACTER COUNT
 5011  030230  012702  035252                  MOV     #MESDAT,R2      ;LOAD MESSAGE POINTER IN R2
 5012  030234  004737  034310          40$:    JSR     PC,INRDY        ;WAIT FOR INRDY
 5013  030240  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 5014  030242  021204                          021204
 5015  030244  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
 5016  030250  112205                          MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
 5017  030252  120504                          CMPB    R5,R4           ;IS RECEIVED DATA CORRECT?
 5018  030254  001401                          BEQ     41$             ;BR IF YES
 5019  030256  104010                          ERROR   10              ;RECEIVE DATA ERROR
 5020  030260  005303          41$:    DEC     R3              ;DEC CHARACTER COUNT
 5021  030262  001364                          BNE     40$             ;BR IF NOT DONE YET
 5022
 5023                                          ;CHECK TO SEE THAT IN BCC MATCH IS SET
 5024                                          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
 5025
 5026  030264  004737  034310                  JSR     PC,INRDY        ;WAIT FOR INRDY
 5027  030270  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 5028  030272  021204                          021204                  ;GET FIRST HALF OF CRC
 5029  030274  116137  000004  001302          MOVB    4(R1),$TMP2     ;PUT IN $TMP2
 5030  030302  042737  177400  001302          BIC     #177400,$TMP2   ;CLEAR HI BYTE
 5031  030310  004737  034310                  JSR     PC,INRDY        ;WAIT FOR INRDY
 5032  030314  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 5033  030316  021244                          021244
 5034  030320  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
 5035  030324  042704  000374                  BIC     #374,R4         ;CLEAR UNWANTED BITS
 5036  030330  012705  000003                  MOV     #3,R5           ;PUT "EXPECTED" IN R5
 5037  030334  120504                          CMPB    R5,R4           ;ARE IN BCC MATCH AND BLOCK END SET?
 5038  030336  001401                          BEQ     50$
 5039  030340  104042                          ERROR   42              ;IN BCC MATCH ERROR
 5040  030342                          50$:
 5041  030342  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 5042  030344  021204                          021204                  ;GET LAST HALF
 5043  030346  116137  000004  001301          MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
 5044  030354  042737  000377  001300          BIC     #377,$TMP1      ;CLEAR LO BYTE
 5045  030362  053737  001300  001302          BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
 5046  030370  023737  034470  001302          CMP     CALBCC,$TMP2    ;IS IT CORRECT?
 5047  030376  001401                          BEQ     42$             ;BR IF OK
 5048  030400  104027                          ERROR   27
 5049
 5050                                          ;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
 5051                                          ;WAS RECEIVED CORRECTLY (0,125,252,377)
 5052
 5053  030402  012703  000004          42$:    MOV     #4,R3           ;R3=CHARACTER COUNT
```

```
5054  030406  012702  035252                 MOV     #MESDAT,R2      ;LOAD MESSAGE POINTER IN R2
5055  030412  004737  034310          43$:   JSR     PC,INRDY        ;WAIT FOR INRDY
5056  030416  104412                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5057  030420  021204                         021204
5058  030422  016104  000004                 MOV     4(R1),R4        ;PUT "FOUND" IN R4
5059  030426  112205                         MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
5060  030430  120504                         CMPB    R5,R4           ;IS RECEIVED DATA CORRECT?
5061  030432  001401                         BEQ     44$             ;BR IF YES
5062  030434  104010                         ERROR   10              ;RECEIVE DATA ERROR
5063  030436  005303                  44$:   DEC     R3              ;DEC CHARACTER COUNT
5064  030440  001364                         BNE     43$             ;BR IF NOT DONE YET
5065
5066                                         ;CHECK TO SEE THAT IN BCC MATCH IS SET
5067                                         ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5068
5069  030442  004737  034310                 JSR     PC,INRDY        ;WAIT FOR INRDY
5070  030446  104412                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5071  030450  021204                         021204                  ;GET FIRST HALF OF CRC
5072  030452  116137  000004  001302         MOVB    4(R1),$TMP2     ;PUT IN $TMP2
5073  030460  042737  177400  001302         BIC     #177400,$TMP2   ;CLEAR HI BYTE
5074  030466  004737  034310                 JSR     PC,INRDY        ;WAIT FOR INRDY
5075  030472  104412                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5076  030474  021244                         021244
5077  030476  016104  000004                 MOV     4(R1),R4        ;PUT "FOUND" IN R4
5078  030502  042704  000374                 BIC     #374,R4         ;CLEAR UNWANTED BITS
5079  030506  012705  000003                 MOV     #3,R5           ;PUT "EXPECTED" IN R5
5080  030512  120504                         CMPB    R5,R4           ;ARE IN BCC MATCH AND BLOCK END SET?
5081  030514  001401                         BEQ     51$
5082  030516  104042                         ERROR   42              ;IN BCC MATCH ERROR
5083  030520                          51$:
5084  030520  104412                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5085  030522  021204                         021204                  ;GET LAST HALF
5086  030524  116137  000004  001301         MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
5087  030532  042737  000377  001300         BIC     #377,$TMP1      ;CLEAR LO BYTE
5088  030540  053737  001300  001302         BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
5089  030546  023737  034470  001302         CMP     CALBCC,$TMP2    ;IS IT CORRECT?
5090  030554  001401                         BEQ     5$              ;BR IF OK
5091  030556  104027                         ERROR   27
5092  030560                          5$:
5093
5094
5095                                         ;*************************** TEST 56 ****************************
5096                                         ;*BITSTUFF EOM FUNCTION TEST
5097                                         ;*THIS TEST LOADS OUT SILO WITH: 2 FLAGS,4 CHAR MESSAGE,EOM
5098                                         ;*SOM,4 CHAR MESS,EOM. THE DATA STREAM IS CHECKED TO BE
5099                                         ;*4 CHAR,BCC,FLAG,4 CHAR,BCC,FLAG,MARKS. THIS TEST VERIFYS THAT
5100                                         ;*THE CHARCTERS LOADED WITH EOM SET ARE LOST
5101                                         ;*ALSO THAT THE CHAR LOADED WITH SOM IS NOT IN THE BCC
5102                                         ;*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
5103                                         ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
5104                                         ;*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
5105                                         ;:****************************************************************
5106
5107                                         ;  TEST 56
5108                                         ;--------------
5109                                 ;;*****************************************************************
```

```
5110  030560  000004                    TST56:  SCOPE
5111  030562  012737  000056  001202            MOV     #56,$TSTNM              ; LOAD THE NO. OF THIS TEST
5112  030570  012737  032240  001442            MOV     #TST57,NEXT             ; POINT TO THE START OF NEXT TEST.
5113                                            ;R1 CONTAINS BASE KMC11 ADDRESS
5114  030576  104410                            MSTCLR                          ;MASTER CLEAR KMC11
5115  030600  005061  000004                    CLR     4(R1)                   ;CLEAR PORT4
5116  030604  104412                            ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5117  030606  122117                            122117                          ;PUT LINE UNIT IN BITSTUFF MODE
5118  030610  004737  035032                    JSR     PC,CLRIO                ;DO THIS AFTER MODE IS SET
5119  030614  005037  035250                    CLR     BITCON                  ;CONSECUTIVE 1'S COUNTER INIT TO 0
5120
5121                                            ;LOAD OUT DATA SILO
5122
5123  030620  012711  004000                    MOV     #BIT11,(R1)             ;SET LINE UNIT LOOP
5124  030624  012704  035252                    MOV     #MESDAT,R4              ;LOAD POINTER TO DATA
5125  030630  005037  030764                    CLR     10$                     ;CLEAR SOFT BCC
5126  030634  005137  030764                    COM     10$                     ;START AT -1
5127  030640  012700  000004                    MOV     #4,R0                   ;LOAD CHARACTER COUNT
5128  030644  004737  034634                    JSR     PC,SYNLD                ;LOAD 2 FLAG CHARACTERS IN OUT SILO
5129  030650  004737  033634                    JSR     PC,OUTRDY               ;WAIT FOR OUTRDY
5130  030654  004537  034770                    JSR     R5,MESLD                ;LOAD SILO WITH 4 CHAR MESS
5131  030660  035252                            MESDAT                          ;ADDRESS OF MESSAGE
5132  030662  000004                            4                               ;NUMBER OF CHARACTERS
5133  030664  004737  034744                    JSR     PC,EOM                  ;LOAD GARBAGE CHARACTER, WITH EOM SET
5134  030670  004737  034744                    JSR     PC,EOM
5135  030674  004737  034714                    JSR     PC,SOM                  ;LOAD GARBAGE CHAR WITH SOM SET
5136  030700  004537  034770                    JSR     R5,MESLD                ;LOAD FOUR MORE CHARACTERS
5137  030704  035252                            MESDAT                          ;ADDRESS OF MESSAGE
5138  030706  000004                            4                               ;NUMBER OF CHACTERS
5139  030710  004737  034744                    JSR     PC,EOM                  ;SET EOM
5140  030714  004737  034744                    JSR     PC,EOM                  ;SET EOM
5141  030720  004737  033502                    JSR     PC,OCOR                 ;WAIT FOR OCOR
5142  030724  005003                            CLR     R3                      ;CLEAR BIT COUNTER
5143  030726  104413  000022                    DATACLK,22                      ;CLOCK DATA
5144  030732  112405                  12$:      MOVB    (R4)+,R5                ;LOAD R5 WITH CHAR
5145  030734  010502                            MOV     R5,R2                   ;LOAD R2 WITH CHAR
5146
5147                                            ;CHECK FIRST FOUR CHARACTER MESSAGE
5148                                            ;IN THE BIT WINDOW (0,125,252,377)
5149
5150  030736  010537  031032                    MOV     R5,71$                  ;LOAD FOR STUFF CHECK
5151  030742  012737  102010  034466            MOV     #CRC.CCITT,XPOLY                ;LOAD POLYNOMIAL
5152  030750  010537  030762                    MOV     R5,67$                  ;LOAD SOFT CHAR FOR BCC
5153  030754  004537  034344                    JSR     R5,SIMBCC               ;CALCULATE SOFT BCC
5154  030760  000010                            10                              ;SHIFT COUNT
5155  030762  000000                  67$:      0                               ;CHARACTER
5156  030764  000000                  10$:      0                               ;OLD BCC
5157  030766  013737  034470  030764            MOV     CALBCC,10$              ;LOAD SOFT BCC FOR NEXT SHIFT
5158  030774  104413  000001          64$:      DATACLK,        1               ;SHIFT DATA IN TO BIT WINDOW
5159  031000  106002                            RORB    R2                      ;SHIFT SOFT DATA
5160  031002  103005                            BCC     65$                     ;BR IF A SPACE
5161  031004  004737  033450                    JSR     PC,GETSI                ;LOOK AT BIT WINDOW
5162  031010  103406                            BCS     66$                     ;BR IF OK (MARK)
5163  031012  104006                            ERROR   6                       ;ERROR, BIT WINDOW WAS A SPACE
5164  031014  000404                            BR      66$                     ;CONTINUE
5165  031016  004737  033450          65$:      JSR     PC,GETSI                ;LOOK AT BIT WINDOW
```

```
5166  031022  103001                        BCC     66$             ;BR IF OK (SPACE)
5167  031024  104006                        ERROR   6               ;ERROR, BIT WINDOW WAS A MARK
5168  031026                          66$:
5169  031026  004537  035132                JSR     R5,STFFCK
5170  031032  000000                  71$:  0
5171  031034  000001                        1
5172  031036  110237  031032                MOVB    R2,71$          ;SHIFT FOR NEXT STUFF CHECK
5173  031042  005203                        INC     R3              ;BUMP BIT COUNTER
5174  031044  022703  000010                CMP     #10,R3          ;DONE FULL 8 BITS YET
5175  031050  001351                        BNE     64$             ;BR IF NO
5176  031052  005003                        CLR     R3              ;CLEAR BIT COUNTER
5177  031054  005300                        DEC     R0              ;DEC CHARACTER COUNT
5178  031056  001325                        BNE     12$             ;BR IF NOT DONE YET
5179
5180                                        ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
5181
5182  031060  005137  034470                COM     CALBCC          ;ADJUST BCC FOR SDLC
5183  031064  013700  034470                MOV     CALBCC,R0       ;PUT BCC IN R0
5184  031070  010037  031132                MOV     R0,72$          ;LOAD BCC FOR STUFF CHECK
5185  031074  104413  000001          68$:  DATACLK,1               ;SHIFT HARDWARE BCC
5186  031100  006000                        ROR     R0              ;SHIFT SOFT BCC
5187  031102  103005                        BCC     69$             ;BR IF CARRY CLEAR
5188  031104  004737  033450                JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5189  031110  103406                        BCS     70$             ;BR IF OK (MARK)
5190  031112  104014                        ERROR   14              ;ERROR, CRC WRONG (SPACE)
5191  031114  000404                        BR      70$             ;CONTINUE
5192  031116  004737  033450          69$:  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5193  031122  103001                        BCC     70$             ;BR IF OK (SPACE)
5194  031124  104014                        ERROR   14              ;ERROR, CRC WRONG (MARK)
5195  031126                          70$:
5196  031126  004537  035132                JSR     R5,STFFCK       ;CHECK BCC CHAR FOR ZERO STUFFS
5197  031132  000000                  72$:  0                       ;CHARACTER
5198  031134  000001                        1                       ;SHIFT COUNT
5199  031136  010037  031132                MOV     R0,72$          ;SHIFT SOFTBCC ONCE
5200  031142  005203                        INC     R3              ;BUMP BIT COUNTER
5201  031144  022703  000020                CMP     #20,R3          ;FINISHED BCC YET?
5202  031150  001351                        BNE     68$             ;BR IF NO
5203  031152  005003                        CLR     R3              ;CLEAR BIT COUNTER
5204
5205                                        ;CHECK FOR FLAG TO FOLLOW BCC
5206
5207  031154  012737  000176  001302        MOV     #^B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
5208  031162  104413  000001          73$:  DATACLK,        1       ;CLOCK FLAG ONCE
5209  031166  106037  001302                RORB    $TMP2           ;SHIFT SOFT FLAG
5210  031172  103405                        BCS     74$             ;BR IF BIT IS MARK
5211  031174  004737  033450                JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5212  031200  103006                        BCC     75$             ;BR IF OK
5213  031202  104026                        ERROR   26              ;ERROR IN FLAG CHAR
5214  031204  000404                        BR      75$
5215  031206  004737  033450          74$:  JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5216  031212  103401                        BCS     75$             ;BR IF OK
5217  031214  104026                        ERROR   26              ;ERROR IN FLAG CHAR
5218  031216  005203                  75$:  INC     R3              ;INC BIT COUNT
5219  031220  022703  000010                CMP     #10,R3          ;FLAG DONE YET?
5220  031224  001356                        BNE     73$             ;BR IF NO
5221  031226  005003                        CLR     R3              ;CLEAR BIT COUNT
```

```
5222
5223                                        ;CHECK FOR ANOTHER FLAG CAUSED BY THE SOM
5224
5225  031230  012737  000176  001302        MOV     #^B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
5226  031236  104413  000001          76$:  DATACLK,          1         ;CLOCK FLAG ONCE
5227  031242  106037  001302                RORB    $TMP2             ;SHIFT SOFT FLAG
5228  031246  103405                        BCS     77$               ;BR IF BIT IS MARK
5229  031250  004737  033450                JSR     PC,GETSI          ;LOOK AT BIT WINDOW
5230  031254  103006                        BCC     78$               ;BR IF OK
5231  031256  104026                        ERROR   26                ;ERROR IN FLAG CHAR
5232  031260  000404                        BR      78$
5233  031262  004737  033450          77$:  JSR     PC,GETSI          ;LOOK AT BIT WINDOW
5234  031266  103401                        BCS     78$               ;BR IF OK
5235  031270  104026                        ERROR   26                ;ERROR IN FLAG CHAR
5236  031272  005203                  78$:  INC     R3                ;INC BIT COUNT
5237  031274  022703  000010                CMP     #10,R3            ;FLAG DONE YET?
5238  031300  001356                        BNE     76$               ;BR IF NO
5239  031302  005003                        CLR     R3                ;CLEAR BIT COUNT
5240  031304  012700  000004                MOV     #4,R0             ;RESET CHARACTER COUNTER
5241  031310  012704  035252                MOV     #MESDAT,R4        ;LOAD MESSAGE POINTER
5242  031314  005037  031356                CLR     11$               ;CLR SOFT BCC
5243  031320  005137  031356                COM     11$               ;ADJUST TO -1 FOR SDLC
5244  031324  112405                  13$:  MOVB    (R4)+,R5          ;LOAD CHAR IN R5
5245  031326  010502                        MOV     R5,R2             ;LOAD CHAR IN R2
5246
5247                                        ;CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
5248
5249  031330  010537  031424                MOV     R5,86$            ;LOAD FOR STUFF CHECK
5250  031334  012737  102010  034466        MOV     #CRC.CCITT,XPOLY          ;LOAD POLYNOMIAL
5251  031342  010537  031354                MOV     R5,82$            ;LOAD SOFT CHAR FOR BCC
5252  031346  004537  034344                JSR     R5,SIMBCC         ;CALCULATE SOFT BCC
5253  031352  000010                        10                        ;SHIFT COUNT
5254  031354  000000                  82$:  0                         ;CHARACTER
5255  031356  000000                  11$:  0                         ;OLD BCC
5256  031360  013737  034470  031356        MOV     CALBCC,11$        ;LOAD SOFT BCC FOR NEXT SHIFT
5257  031366  104413  000001          79$:  DATACLK,          1         ;SHIFT DATA IN TO BIT WINDOW
5258  031372  106002                        RORB    R2                ;SHIFT SOFT DATA
5259  031374  103005                        BCC     80$               ;BR IF A SPACE
5260  031376  004737  033450                JSR     PC,GETSI          ;LOOK AT BIT WINDOW
5261  031402  103406                        BCS     81$               ;BR IF OK (MARK)
5262  031404  104006                        ERROR   6                 ;ERROR, BIT WINDOW WAS A SPACE
5263  031406  000404                        BR      81$               ;CONTINUE
5264  031410  004737  033450          80$:  JSR     PC,GETSI          ;LOOK AT BIT WINDOW
5265  031414  103001                        BCC     81$               ;BR IF OK (SPACE)
5266  031416  104006                        ERROR   6                 ;ERROR, BIT WINDOW WAS A MARK
5267  031420                            81$:
5268  031420  004537  035132                JSR     R5,STFFCK
5269  031424  000000                  86$:  0
5270  031426  000001                        1
5271  031430  110237  031424                MOVB    R2,86$            ;SHIFT FOR NEXT STUFF CHECK
5272  031434  005203                        INC     R3                ;BUMP BIT COUNTER
5273  031436  022703  000010                CMP     #10,R3            ;DONE FULL 8 BITS YET
5274  031442  001351                        BNE     79$               ;BR IF NO
5275  031444  005003                        CLR     R3                ;CLEAR BIT COUNTER
5276  031446  005300                        DEC     R0                ;DEC CHARACTER COUNT
5277  031450  001325                        BNE     13$               ;BR IF NOT DONE YET
```

```
5278
5279                                        ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
5280
5281  031452  005137  034470                COM     CALBCC              ;ADJUST BCC FOR SDLC
5282  031456  013700  034470                MOV     CALBCC,R0           ;PUT BCC IN R0
5283  031462  010037  031524                MOV     R0,87$              ;LOAD BCC FOR STUFF CHECK
5284  031466  104413  000001      83$:      DATACLK,1                   ;SHIFT HARDWARE BCC
5285  031472  006000                        ROR     R0                  ;SHIFT SOFT BCC
5286  031474  103005                        BCC     84$                 ;BR IF CARRY CLEAR
5287  031476  004737  033450                JSR     PC,GETSI            ;LOOK AT BIT WINDOW
5288  031502  103406                        BCS     85$                 ;BR IF OK (MARK)
5289  031504  104014                        ERROR   14                  ;ERROR, CRC WRONG (SPACE)
5290  031506  000404                        BR      85$                 ;CONTINUE
5291  031510  004737  033450      84$:      JSR     PC,GETSI            ;LOOK AT BIT WINDOW
5292  031514  103001                        BCC     85$                 ;BR IF OK (SPACE)
5293  031516  104014                        ERROR   14                  ;ERROR, CRC WRONG (MARK)
5294  031520                      85$:
5295  031520  004537  035132                JSR     R5,STFFCK           ;CHECK BCC CHAR FOR ZERO STUFFS
5296  031524  000000      87$:              0                           ;CHARACTER
5297  031526  000001                        1                           ;SHIFT COUNT
5298  031530  010037  031524                MOV     R0,87$              ;SHIFT SOFTBCC ONCE
5299  031534  005203                        INC     R3                  ;BUMP BIT COUNTER
5300  031536  022703  000020                CMP     #20,R3              ;FINISHED BCC YET?
5301  031542  001351                        BNE     83$                 ;BR IF NO
5302  031544  005003                        CLR     R3                  ;CLEAR BIT COUNTER
5303
5304                                        ;CHECK FOR FLAG TO FOLLOW BCC
5305
5306  031546  012737  000176  001302        MOV     #^B<01111110>,$TMP2 ;PUT FLAG CHARACTER IN $TMP2
5307  031554  104413  000001      88$:      DATACLK,        1           ;CLOCK FLAG ONCE
5308  031560  106037  001302                RORB    $TMP2               ;SHIFT SOFT FLAG
5309  031564  103405                        BCS     89$                 ;BR IF BIT IS MARK
5310  031566  004737  033450                JSR     PC,GETSI            ;LOOK AT BIT WINDOW
5311  031572  103006                        BCC     90$                 ;BR IF OK
5312  031574  104026                        ERROR   26                  ;ERROR IN FLAG CHAR
5313  031576  000404                        BR      90$
5314  031600  004737  033450      89$:      JSR     PC,GETSI            ;LOOK AT BIT WINDOW
5315  031604  103401                        BCS     90$                 ;BR IF OK
5316  031606  104026                        ERROR   26                  ;ERROR IN FLAG CHAR
5317  031610  005203      90$:              INC     R3                  ;INC BIT COUNT
5318  031612  022703  000010                CMP     #10,R3              ;FLAG DONE YET?
5319  031616  001356                        BNE     88$                 ;BR IF NO
5320  031620  005003                        CLR     R3                  ;CLEAR BIT COUNT
5321
5322                                        ;CHECK TO SEE IF TRANSMITTER IS MARKING
5323
5324  031622  104413  000001        2$:     DATACLK,        1           ;CLOCK TRANSMITTER
5325  031626  004737  033450                JSR     PC,GETSI            ;LOOK AT WINDOW
5326  031632  103401                        BCS     3$                  ;IT SHOULD BE MARKING
5327  031634  104024                        ERROR   24                  ;ERROR, BIT WAS A SPACE
5328  031636  005203        3$:             INC     R3                  ;BUMP BIT COUNTER
5329  031640  022703  000007                CMP     #7,R3               ;DONE YET
5330  031644  001366                        BNE     2$                  ;BR IF NO
5331  031646  104413  000010                DATACLK,        10          ;GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
5332  031652  005003                        CLR     R3                  ;CLEAR BIT COUNTER
5333  031654  104413  000001        4$:     DATACLK,        1           ;SHIFT OUT NEXT BIT
```

```
5334  031660  004737  033450              JSR     PC,GETSI        ;LOOK AT BIT WINDOW
5335  031664  103401                      BCS     .+4             ;BR IF IT IS A MARK
5336  031666  104024                      ERROR   24              ;ERROR, TRANSMITTER IS NOT MARKING
5337  031670  005203                      INC     R3              ;INC BIT COUNT
5338  031672  022703  000020              CMP     #20,R3          ;DONE YET?
5339  031676  001366                      BNE     4$              ;BR IF NO
5340
5341                                      ;CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
5342                                      ;WAS RECEIVED CORRECTLY (0,125,252,377)
5343
5344  031700  104413  000001              DATACLK,        1       ;GET LAST BIT IN RECEIVER
5345  031704  012703  000004              MOV     #4,R3           ;R3=CHARACTER COUNT
5346  031710  012702  035252              MOV     #MESDAT,R2      ;LOAD MESSAGE POINTER IN R2
5347  031714  004737  034310      40$:    JSR     PC,INRDY        ;WAIT FOR INRDY
5348  031720  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5349  031722  021204                      021204
5350  031724  016104  000004              MOV     4(R1),R4        ;PUT "FOUND" IN R4
5351  031730  112205                      MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
5352  031732  120504                      CMPB    R5,R4           ;IS RECEIVED DATA CORRECT?
5353  031734  001401                      BEQ     41$             ;BR IF YES
5354  031736  104010                      ERROR   10              ;RECEIVE DATA ERROR
5355  031740  005303              41$:    DEC     R3              ;DEC CHARACTER COUNT
5356  031742  001364                      BNE     40$             ;BR IF NOT DONE YET
5357
5358                                      ;CHECK TO SEE THAT IN BCC MATCH IS SET
5359                                      ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5360
5361  031744  004737  034310              JSR     PC,INRDY        ;WAIT FOR INRDY
5362  031750  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5363  031752  021204                      021204                  ;GET FIRST HALF OF CRC
5364  031754  116137  000004  001302      MOVB    4(R1),$TMP2     ;PUT IN $TMP2
5365  031762  042737  177400  001302      BIC     #177400,$TMP2   ;CLEAR HI BYTE
5366  031770  004737  034310              JSR     PC,INRDY        ;WAIT FOR INRDY
5367  031774  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5368  031776  021244                      021244
5369  032000  016104  000004              MOV     4(R1),R4        ;PUT "FOUND" IN R4
5370  032004  042704  000374              BIC     #374,R4         ;CLEAR UNWANTED BITS
5371  032010  012705  000003              MOV     #3,R5           ;PUT "EXPECTED" IN R5
5372  032014  120504                      CMPB    R5,R4           ;ARE IN BCC MATCH AND BLOCK END SET?
5373  032016  001401                      BEQ     50$
5374  032020  104042                      ERROR   42              ;IN BCC MATCH ERROR
5375  032022                      50$:
5376  032022  104412                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5377  032024  021204                      021204                  ;GET LAST HALF
5378  032026  116137  000004  001301      MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
5379  032034  042737  000377  001300      BIC     #377,$TMP1      ;CLEAR LO BYTE
5380  032042  053737  001300  001302      BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
5381  032050  023737  034470  001302      CMP     CALBCC,$TMP2    ;IS IT CORRECT?
5382  032056  001401                      BEQ     42$             ;BR IF OK
5383  032060  104027                      ERROR   27
5384
5385                                      ;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
5386                                      ;WAS RECEIVED CORRECTLY (0,125,252,377)
5387
5388  032062  012703  000004      42$:    MOV     #4,R3           ;R3=CHARACTER COUNT
5389  032066  012702  035252              MOV     #MESDAT,R2      ;LOAD MESSAGE POINTER IN R2
```

```
5390  032072  004737  034310          43$:    JSR     PC,INRDY        ;WAIT FOR INRDY
5391  032076  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5392  032100  021204                          021204
5393  032102  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
5394  032106  112205                          MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
5395  032110  120504                          CMPB    R5,R4           ;IS RECEIVED DATA CORRECT?
5396  032112  001401                          BEQ     44$             ;BR IF YES
5397  032114  104010                          ERROR   10              ;RECEIVE DATA ERROR
5398  032116  005303                  44$:    DEC     R3              ;DEC CHARACTER COUNT
5399  032120  001364                          BNE     43$             ;BR IF NOT DONE YET
5400
5401                                          ;CHECK TO SEE THAT IN BCC MATCH IS SET
5402                                          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5403
5404  032122  004737  034310                  JSR     PC,INRDY        ;WAIT FOR INRDY
5405  032126  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5406  032130  021204                          021204                  ;GET FIRST HALF OF CRC
5407  032132  116137  000004  001302          MOVB    4(R1),$TMP2     ;PUT IN $TMP2
5408  032140  042737  177400  001302          BIC     #177400,$TMP2   ;CLEAR HI BYTE
5409  032146  004737  034310                  JSR     PC,INRDY        ;WAIT FOR INRDY
5410  032152  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5411  032154  021244                          021244
5412  032156  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
5413  032162  042704  000374                  BIC     #374,R4         ;CLEAR UNWANTED BITS
5414  032166  012705  000003                  MOV     #3,R5           ;PUT "EXPECTED" IN R5
5415  032172  120504                          CMPB    R5,R4           ;ARE IN BCC MATCH AND BLOCK END SET?
5416  032174  001401                          BEQ     51$
5417  032176  104042                          ERROR   42              ;IN BCC MATCH ERROR
5418  032200                          51$:
5419  032200  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5420  032202  021204                          021204                  ;GET LAST HALF
5421  032204  116137  000004  001301          MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
5422  032212  042737  000377  001300          BIC     #377,$TMP1      ;CLEAR LO BYTE
5423  032220  053737  001300  001302          BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
5424  032226  023737  034470  001302          CMP     CALBCC,$TMP2    ;IS IT CORRECT?
5425  032234  001401                          BEQ     5$              ;BR IF OK
5426  032236  104027                          ERROR   27
5427  032240                          5$:
5428
5429
5430                                          ;*************************** TEST 57 ****************************
5431                                          ;*EMPTY SILO TEST
5432                                          ;*LOAD SILO WITH 2 SYNCS, 4 CHAR MESSAGE, SINGLE CLOCK
5433                                          ;*UNTIL THE SILO IS EMPTY, LOAD 4 MORE CHARACTERS IN THE
5434                                          ;*SILO. GIVE MORE TICKS, AND VERIFY THAT ONLY THE FIRST
5435                                          ;*4 CHARACTERS  AND A BLOCK END WERE RECEIVED, AND IN ACTIVE IS CLEAR
5436                                          ;;*************************************************************
5437
5438                                          ;  TEST 57
5439                                          ;-------------
5440                                  ;;*************************************************************
5441  032240  000004                  TST57:  SCOPE
5442  032242  012737  000057  001202          MOV     #57,$TSTNM              ; LOAD THE NO. OF THIS TEST
5443  032250  012737  032460  001442          MOV     #TST60,NEXT             ; POINT TO THE START OF NEXT TEST.
5444                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5445  032256  104410                          MSTCLR                  ;MASTER CLEAR KMC11
```

```
5446  032260  005061  000004                CLR     4(R1)           ;CLEAR PORT4
5447  032264  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5448  032266  122117                        122117                  ;PUT LU IN BITSTUFF MODE
5449  032270  004737  035032                JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
5450  032274  012711  004000                MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
5451  032300  012702  035252                MOV     #MESDAT,R2      ;R2 POINTS TO MESSAGE
5452  032304  012700  000003                MOV     #3,R0           ;R0 = CHAR COUNT
5453  032310  004737  034634                JSR     PC,SYNLD        ;LOAD SILO WITH TWO FLAGS
5454  032314  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5455  032320  004537  034770                JSR     R5,MESLD        ;LOAD MESSAGE IN SILO
5456  032324  035252                        MESDAT                  ;START OF MESSAGE
5457  032326  000004                        4                       ;CHARACTER COUNT
5458  032330  004737  033502                JSR     PC,OCOR         ;WAIT FOR OCOR
5459  032334  104413  000065                DATACLK,        65      ;CLOCK DATA (EMPTY SILO)
5460  032340  004537  034770                JSR     R5,MESLD        ;PUT MORE CHARACTERS IN SILO
5461  032344  035252                        MESDAT
5462  032346  000004                        4
5463  032350  004737  033502                JSR     PC,OCOR
5464  032354  104413  000006                DATACLK,        6       ;CLOCK UNTIL RTS IS CLEARED
5465  032360  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5466  032362  021264                        021264                  ;GET RTS
5467  032364  032761  000040  000004        BIT     #BIT5,4(R1)     ;IS IT CLEAR?
5468  032372  001401                        BEQ     5$              ;BR IF YES
5469  032374  104034                        ERROR   34              ;ERROR, RTS NOT CLEAR
5470  032376  104413  000041          5$:   DATACLK,        41      ;CLOCK XMITTER SOME MORE
5471  032402  004737  034310          1$:   JSR     PC,INRDY        ;OK LETS CHECK WHAT WAS RECEIVED
5472  032406  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5473  032410  021204                        021204                  ;GET RECEIVE DATA
5474  032412  016104  000004                MOV     4(R1),R4        ;PUT IT IN R4
5475  032416  112205                        MOVB    (R2)+,R5        ;R5 = "EXPECTED"
5476  032420  120504                        CMPB    R5,R4           ;IS DATA CORRECT?
5477  032422  001401                        BEQ     2$              ;BR IF OK
5478  032424  104010                        ERROR   10              ;DATA ERROR
5479  032426  005300          2$:   DEC     R0                      ;DEC CHAR COUNT
5480  032430  001364                        BNE     1$              ;BR IF NOT DONE YET
5481  032432  004737  034310                JSR     PC,INRDY        ;WAIT FOR INRDY
5482  032436  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5483  032440  021244                        021244                  ;READ LU-12
5484  032442  016104  000004                MOV     4(R1),R4        ;PUT "FOUND" IN R4
5485  032446  012705  000022                MOV     #22,R5          ;PUT "EXPECTED" IN R5
5486  032452  120504                        CMPB    R5,R4           ;ARE BLOCK END AND IN RDY SET?
5487                                                                ;AND IN ACTIVE AND IN BCC MATCH CLEAR?
5488  032454  001401                        BEQ     6$              ;BR IF YES
5489  032456  104032                        ERROR   32              ;ERROR, BLOCK END NOT SET
5490                                                                ;OR IN BCC MATCH NOT CLEAR
5491                                                                ;OR IN ACTIVE NOT CLEAR
5492  032460                          6$:
5493
5494
5495                                ;*************************** TEST 60 ****************************
5496                                ;*BITSTUFF CABLE DATA TEST
5497                                ;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
5498                                ;*2 FLAGS,16 CHAR,EOM,16 CHAR,EOM,16 CHAR,EOM
5499                                ;*THE 16 CHARACTERS INCLUDE A FLOATING ONE AND ZERO
5500                                ;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
5501                                ;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
```

```
                                        ;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
                                        ;:*****************************************************************
                                        ;
                                        ;  TEST 60
                                        ;-------------
                                    ;;*****************************************************************
032460  000004                    TST60:  SCOPE
032462  012737  000060  001202            MOV     #60,$TSTNM              ; LOAD THE NO. OF THIS TEST
032470  012737  033104  001442            MOV     #TST61,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
032476  104410                            MSTCLR                  ;MASTER CLEAR KMC11
032500  032737  040000  002050            BIT     #BIT14,STAT1    ;SKIP TEST IF NO
032506  001575                            BEQ     3$              ;LOOPBACK CONNECTOR ON
032510  005061  000004                    CLR     4(R1)           ;CLEAR PORT4
032514  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
032516  122117                            122117                  ;PUT LINE UNIT IN BITSTUFF MODE
032520  004737  035032                    JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
032524  012711  004000                    MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
032530  004737  034634                    JSR     PC,SYNLD        ;LOAD TWO FLAGS
032534  012737  102010  034466            MOV     #CRC.CCITT,XPOLY;LOAD POLYNOMIAL FOR SOFT CRC CALC
032542  005037  032576                    CLR     6$              ;CLEAR OLD BCC
032546  005137  032576                    COM     6$              ;ADJUST TO -1 FOR SDLC
032552  012703  000020                    MOV     #16.,R3         ;CHARACTER COUNT
032556  012702  035256                    MOV     #FLTDAT,R2      ;R2= POINTER
032562  112237  032574            7$:     MOVB    (R2)+,5$        ;LOAD CHAR FOR SOFT BCC CALC.
032566  004537  034344                    JSR     R5,SIMBCC       ;CALC SOFT BCC
032572  000010                            10                      ;SHIFT COUNT
032574  000000                    5$:     0                       ;CHARACTER
032576  000000                    6$:     0                       ;OLD BCC
032600  013737  034470  032576            MOV     CALBCC,6$       ;LOAD OLD BCC
032606  005303                            DEC     R3              ;DEC COUNT
032610  001364                            BNE     7$              ;BR IF NOT DONE YET
032612  005137  034470                    COM     CALBCC          ;ADJUST CALBCC FOR SDLC
032616  004537  034770                    JSR     R5,MESLD        ;LOAD SILO
032622  035256                            FLTDAT                  ;MESSAGE ADDRESS
032624  000020                            16.                     ;CHARACTER COUNT
032626  004737  034744                    JSR     PC,EOM          ;LOAD AN EOM
032632  004737  034744                    JSR     PC,EOM
032636  004537  034770                    JSR     R5,MESLD        ;LOAD SILO
032642  035256                            FLTDAT                  ;MESSAGE ADDRESS
032644  000020                            16.                     ;CHARACTER COUNT
032646  004737  034744                    JSR     PC,EOM          ;LOAD AN EOM
032652  004737  034744                    JSR     PC,EOM
032656  004537  034770                    JSR     R5,MESLD        ;LOAD SILO
032662  035256                            FLTDAT                  ;MESSAGE ADDRESS
032664  000020                            16.                     ;CHARACTER COUNT
032666  004737  034744                    JSR     PC,EOM          ;LOAD AN EOM
032672  004737  034744                    JSR     PC,EOM
032676  004737  033502                    JSR     PC,OCOR         ;WAIT FOR OCOR
032702  005011                            CLR     (R1)            ;CLEAR LINE UNIT LOOP
032704  012700  000003                    MOV     #3,R0           ;R0 = MESSAGE COUNT
032710  012703  000020                    MOV     #16.,R3         ;R3= CHARACTER COUNT
032714  012702  035256                    MOV     #FLTDAT,R2      ;LOAD MESSAGE POINTER IN R2
032720  004737  034310            1$:     JSR     PC,INRDY        ;WAIT FOR INRDY
032724  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
032726  021204                            021204                  ;GET DATA FROM IN SILO
```

```
5558  032730  016104  000004                  MOV     4(R1),R4        ;PUT CHARACTER IN "FOUND"
5559  032734  112205                          MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
5560  032736  120504                          CMPB    R5,R4           ;IS RECEIVED DATA CORRECT
5561  032740  001401                          BEQ     2$              ;BR IF OK
5562  032742  104025                          ERROR   25              ;DATA ERROR
5563  032744                          2$:
5564  032744  005303                          DEC     R3              ;DEC CHARACTER COUNT
5565  032746  001364                          BNE     1$              ;BR IF NOT DONE THIS MESSAGE
5566  032750  012703  000020                  MOV     #16.,R3         ;RESET CHARACTER COUNT
5567
5568                                          ;CHECK TO SEE THAT IN BCC MATCH IS SET
5569                                          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5570
5571  032754  004737  034310                  JSR     PC,INRDY        ;WAIT FOR INRDY
5572  032760  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5573  032762  021204                          021204                  ;GET FIRST HALF OF CRC
5574  032764  116137  000004  001302          MOVB    4(R1),$TMP2     ;PUT IN $TMP2
5575  032772  042737  177400  001302          BIC     #177400,$TMP2   ;CLEAR HI BYTE
5576  033000  004737  034310                  JSR     PC,INRDY        ;WAIT FOR INRDY
5577  033004  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5578  033006  021244                          021244
5579  033010  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
5580  033014  042704  000374                  BIC     #374,R4         ;CLEAR UNWANTED BITS
5581  033020  012705  000003                  MOV     #3,R5           ;PUT "EXPECTED" IN R5
5582  033024  120504                          CMPB    R5,R4           ;ARE IN BCC MATCH AND BLOCK END SET?
5583  033026  001401                          BEQ     25$
5584  033030  104042                          ERROR   42              ;IN BCC MATCH ERROR
5585  033032                          25$:
5586  033032  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5587  033034  021204                          021204                  ;GET LAST HALF
5588  033036  116137  000004  001301          MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
5589  033044  042737  000377  001300          BIC     #377,$TMP1      ;CLEAR LO BYTE
5590  033052  053737  001300  001302          BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
5591  033060  023737  034470  001302          CMP     CALBCC,$TMP2    ;IS IT CORRECT?
5592  033066  001401                          BEQ     4$              ;BR IF OK
5593  033070  104027                          ERROR   27
5594  033072  012702  035256          4$:     MOV     #FLTDAT,R2      ;RESET MESSAGE POINTER
5595  033076  005300                          DEC     R0              ;DECREMENT COUNTER
5596  033100  001307                          BNE     1$              ;BR IF NOT DONE
5597  033102  104420                  3$:     ADVANCE         ; ADVANCE TO NEXT TEST
5598
5599
5600                                          ;*************************** TEST 61 ****************************
5601                                          ;*BITSTUFF CABLE DATA TEST
5602                                          ;*THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
5603                                          ;*2 FLAGS,59 DATA CHARACTERS,EOM WITH GARBAGE CHARACTER
5604                                          ;*THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
5605                                          ;*RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
5606                                          ;*LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
5607                                          ;:*************************************************************
5608
5609                                          ;  TEST 61
5610                                          ;--------------
5611                                  ;;*************************************************************
5612  033104  000004                  TST61:  SCOPE
5613  033106  012737  000061  001202          MOV     #61,$TSTNM              ; LOAD THE NO. OF THIS TEST
```

```
5614  033114  012737  003662  001442          MOV     #$EOP,NEXT              ; POINT TO THE END OF PASS HANDLER.
5615                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
5616  033122  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5617  033124  032737  040000  002050          BIT     #BIT14,STAT1    ;SKIP TEST IF NO
5618  033132  001545                          BEQ     3$              ;LOOPBACK CONNECTOR ON
5619  033134  005061  000004                  CLR     4(R1)           ;CLEAR PORT4
5620  033140  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5621  033142  122117                          122117                  ;PUT LINE UNIT IN BITSTUFF MODE
5622  033144  004737  035032                  JSR     PC,CLRIO        ;DO THIS AFTER MODE IS SET
5623  033150  012711  004000                  MOV     #BIT11,(R1)     ;SET LINE UNIT LOOP
5624  033154  004737  034634                  JSR     PC,SYNLD        ;LOAD TWO FLAGS
5625  033160  012737  102010  034466          MOV     #CRC.CCITT,XPOLY;LOAD POLYNOMIAL FOR SOFT CRC CALC
5626  033166  005037  033222                  CLR     6$              ;CLEAR OLD BCC
5627  033172  005137  033222                  COM     6$              ;ADJUST TO -1 FOR SDLC
5628  033176  012703  000073                  MOV     #59.,R3         ;CHARACTER COUNT
5629  033202  012702  035252                  MOV     #MESDAT,R2      ;R2= POINTER
5630  033206  112237  033220          7$:     MOVB    (R2)+,5$        ;LOAD CHAR FOR SOFT BCC CALC.
5631  033212  004537  034344                  JSR     R5,SIMBCC       ;CALC SOFT BCC
5632  033216  000010                          10                      ;SHIFT COUNT
5633  033220  000000                  5$:     0                       ;CHARACTER
5634  033222  000000                  6$:     0                       ;OLD BCC
5635  033224  013737  034470  033222          MOV     CALBCC,6$       ;LOAD OLD BCC
5636  033232  005303                          DEC     R3              ;DEC COUNT
5637  033234  001364                          BNE     7$              ;BR IF NOT DONE YET
5638  033236  005137  034470                  COM     CALBCC          ;ADJUST CALBCC FOR SDLC
5639  033242  004537  034770                  JSR     R5,MESLD        ;LOAD SILO
5640  033246  035252                          MESDAT                  ;MESSAGE ADDRESS
5641  033250  000073                          59.                     ;CHARACTER COUNT
5642  033252  004737  034744                  JSR     PC,EOM          ;LOAD AN EOM
5643  033256  004737  034744                  JSR     PC,EOM
5644  033262  004737  033502                  JSR     PC,OCOR         ;WAIT FOR OCOR
5645  033266  005011                          CLR     (R1)            ;CLEAR LINE UNIT LOOP
5646  033270  012700  000073                  MOV     #59.,R0         ;R0= CHARACTER COUNT
5647  033274  012702  035252                  MOV     #MESDAT,R2      ;LOAD MESSAGE POINTER IN R2
5648  033300  004737  034310          1$:     JSR     PC,INRDY        ;WAIT FOR INRDY
5649  033304  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5650  033306  021204                          021204                  ;GET DATA FROM IN SILO
5651  033310  016104  000004                  MOV     4(R1),R4        ;PUT CHARACTER IN "FOUND"
5652  033314  112205                          MOVB    (R2)+,R5        ;PUT "EXPECTED" IN R5
5653  033316  120504                          CMPB    R5,R4           ;IS RECEIVED DATA CORRECT
5654  033320  001401                          BEQ     2$              ;BR IF OK
5655  033322  104025                          ERROR   25              ;DATA ERROR
5656  033324                          2$:
5657  033324  005300                          DEC     R0              ;DECREMENT COUNTER
5658  033326  001364                          BNE     1$              ;BR IF NOT DONE
5659
5660                                          ;CHECK TO SEE THAT IN BCC MATCH IS SET
5661                                          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5662
5663  033330  004737  034310                  JSR     PC,INRDY        ;WAIT FOR INRDY
5664  033334  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5665  033336  021204                          021204                  ;GET FIRST HALF OF CRC
5666  033340  116137  000004  001302          MOVB    4(R1),$TMP2     ;PUT IN $TMP2
5667  033346  042737  177400  001302          BIC     #177400,$TMP2   ;CLEAR HI BYTE
5668  033354  004737  034310                  JSR     PC,INRDY        ;WAIT FOR INRDY
5669  033360  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
                                                    021244
5670  033362  021244                                MOV     4(R1),R4        ;PUT "FOUND" IN R4
5671  033364  016104  000004                        BIC     #374,R4         ;CLEAR UNWANTED BITS
5672  033370  042704  000374                        MOV     #3,R5           ;PUT "EXPECTED" IN R5
5673  033374  012705  000003                        CMPB    R5,R4           ;ARE IN BCC MATCH AND BLOCK END SET?
5674  033400  120504                                BEQ     25$
5675  033402  001401                                ERROR   42              ;IN BCC MATCH ERROR
5676  033404  104042                        25$:
5677  033406                                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5678  033406  104412                                021204                  ;GET LAST HALF
5679  033410  021204                                MOVB    4(R1),$TMP1+1   ;PUT IN $TMP1
5680  033412  116137  000004  001301                BIC     #377,$TMP1      ;CLEAR LO BYTE
5681  033420  042737  000377  001300                BIS     $TMP1,$TMP2     ;16 BIT BCC NOW IN $TMP2
5682  033426  053737  001300  001302                CMP     CALBCC,$TMP2    ;IS IT CORRECT?
5683  033434  023737  034470  001302                BEQ     3$              ;BR IF OK
5684  033442  001401                                ERROR   27
5685  033444  104027                        3$:     ADVANCE         ; ADVANCE TO NEXT TEST
5686  033446  104420
5687
5688
5689                                        ;SUBROUTINES
5690                                        ;-----------
5691
5692  033450                                GETSI:
5693                                                ;THIS SUBROUTINE READS LU 17, AND PUTS IT INTO NITCH.
5694                                                ;NITCH IS ROTATED LEFT UNTILL THE SI BIT IS IN CARRY
5695
5696  033450  104412                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5697  033452  021364                                021364                  ;PORT4←LU 17
5698  033454  017737  146414  033500                MOV     @KMP04,NITCH    ;STORE LU 17
5699  033462  106137  033500                        ROLB    NITCH
5700  033466  106137  033500                        ROLB    NITCH
5701  033472  106137  033500                        ROLB    NITCH           ;PUT SI IN THE CARRY BIT
5702  033476  000207                                RTS     PC
5703  033500  000000                        NITCH:  0
5704
5705
5706  033502                                OCOR:
5707                                                ;THIS SUBROUTINE SPINS ON OCOR
5708
5709  033502  104412                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5710  033504  021364                                021364                  ;PORT4←LU 17
5711  033506  032777  000020  146360                BIT     #BIT4,@KMP04    ;IS OCOR SET?
5712  033514  001772                                BEQ     OCOR            ;BR IF NO
5713  033516  000207                                RTS     PC              ;OK OCOR IS SET, GO BACK
5714
5715
5716  033520                                SYNC:
5717                                                ;THIS SUBROUNTINE LOADS THE SILO WITH THE NUMBER OF SYNC
5718                                                ;CHARACTERS PASSED TO IT IN THE WORD AFTER THE JSR CALL
5719                                                ;AND A NON-SYNC CHARACTER (301)
5720
5721  033520  013637  001276                        MOV     @(SP)+,$TMP0    ;GET COUNT
5722  033524  062746  000002                        ADD     #2,-(SP)        ;ADJUST STACK
5723  033530  012761  000026  000004                MOV     #26,4(R1)       ;LOAD PORT4
5724  033536  104412                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5725  033540  122114                                122114                  ;LOAD SYNC REGISTER
```

```
5726  033542  004737  033634          1$:   JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5727  033546  012761  000001  000004        MOV     #1,4(R1)        ;LOAD PORT4
5728  033554  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5729  033556  122111                        122111                  ;SET SOM
5730  033560  012761  000026  000004        MOV     #26,4(R1)       ;LOAD PORT4
5731  033566  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5732  033570  122110                        122110                  ;LOAD OUT DATA
5733  033572  005337  001276                DEC     $TMP0           ;ALL DONE?
5734  033576  001361                        BNE     1$              ;BR IF NOT
5735  033600  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5736  033604  005061  000004                CLR     4(R1)           ;LOAD PORT4
5737  033610  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5738  033612  122111                        122111                  ;SET SOM
5739  033614  012761  000301  000004        MOV     #301,4(R1)      ;LOAD PORT4
5740  033622  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5741  033624  122110                        122110                  ;LOAD OUT DATA
5742  033626  004737  033502                JSR     PC,OCOR         ;WAIT FOR OCOR
5743  033632  000207                        RTS     PC
5744
5745
5746  033634                          OUTRDY:
5747                                        ;THIS SUBROUTINE SPINS ON OUT READY
5748
5749  033634  005037  001306                CLR     $TMP4           ;CLEAR TIMER
5750  033640                          1$:
5751  033640  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5752  033642  021224                        021224                  ;PORT4+LU11
5753  033644  032777  000020  146222        BIT     #BIT4,@KMP04    ;IS OUT RDY SET?
5754  033652  001004                        BNE     2$              ;BR IF YES
5755  033654  005237  001306                INC     $TMP4           ;INC TIMER
5756  033660  001367                        BNE     1$              ;KEEP CHECKING IF NOT DONE
5757  033662  104036                        ERROR   36              ;ERROR, OUT READY NOT SET
5758  033664  000207                  2$:   RTS     PC
5759
5760
5761  033666                          CHAR:
5762                                        ;THIS SUBROUTINE LOADS THE SILO WITH 3 SYNCS
5763                                        ;AND THE CHARACTER PASSED TO IT.
5764
5765  033666  013637  001300                MOV     @(SP)+,$TMP1    ;GET CHARACTER
5766  033672  062746  000002                ADD     #2,-(SP)        ;ADJUST STACK
5767  033676  012737  000003  001276        MOV     #3,$TMP0        ;SET FOR 3 SYNCS
5768  033704  012761  000026  000004        MOV     #26,4(R1)       ;LOAD PORT4
5769  033712  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5770  033714  122114                        122114                  ;LOAD SYNC REGISTER
5771  033716  004737  033634          1$:   JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5772  033722  012761  000001  000004        MOV     #1,4(R1)        ;LOAD PORT4
5773  033730  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5774  033732  122111                        122111                  ;SET SOM
5775  033734  012761  000026  000004        MOV     #26,4(R1)       ;LOAD PORT4
5776  033742  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5777  033744  122110                        122110                  ;LOAD OUT DATA
5778  033746  005337  001276                DEC     $TMP0           ;ALL DONE?
5779  033752  001361                        BNE     1$              ;BR IF NOT
5780  033754  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5781  033760  013761  001300  000004        MOV     $TMP1,4(R1)     ;LOAD PORT4
```

```
5782  033766  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5783  033770  122110                        122110                  ;LOAD OUT DATA
5784  033772  004737  033502                JSR     PC,OCOR         ;WAIT FOR OCOR
5785  033776  000207                        RTS     PC
5786
5787
5788  034000                        CHARSD:
5789                                        ;THIS SUBROUTINE LOADS THE SILO WITH THE CHARACTER PASSED TO IT.
5790
5791  034000  013637  001300                MOV     @(SP)+,$TMP1    ;GET CHARACTER
5792  034004  062746  000002                ADD     #2,-(SP)        ;ADJUST STACK
5793  034010  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5794  034014  013761  001300  000004        MOV     $TMP1,4(R1)     ;LOAD PORT4
5795  034022  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5796  034024  122110                        122110                  ;LOAD OUT DATA
5797  034026  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5798  034032  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5799  034034  122110                        122110                  ;LOAD GARBAGE CHAR
5800  034036  004737  033502                JSR     PC,OCOR         ;WAIT FOR OCOR
5801  034042  000207                        RTS     PC
5802
5803
5804  034044                        SILOLD:
5805                                        ;THIS SUBROUTINE FILLS THE OUT SILO
5806                                        ; WITH A BINARY COUNT PATTERN
5807
5808  034044  012737  000073  001300        MOV     #73,$TMP1       ;LOAD COUNT
5809  034052  005737  034304                TST     SCHAR           ;FIRST TIME HERE?
5810  034056  100470                        BMI     4$              ;BR IF BITSTUFF
5811  034060  001032                        BNE     2$              ;BR IF NO
5812  034062  062737  000002  001300        ADD     #2,$TMP1        ;ADD 2 TO CHARACTER COUNT
5813  034070  012737  000003  001276        MOV     #3,$TMP0        ;SET FOR 3 SYNCS
5814  034076  012761  000026  000004        MOV     #26,4(R1)       ;LOAD PORT4
5815  034104  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5816  034106  122114                        122114                  ;LOAD SYNC REGISTER
5817  034110  004737  033634        1$:     JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5818  034114  012761  000001  000004        MOV     #1,4(R1)        ;LOAD PORT4
5819  034122  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5820  034124  122111                        122111                  ;SET SOM
5821  034126  012761  000026  000004        MOV     #26,4(R1)       ;LOAD PORT4
5822  034134  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5823  034136  122110                        122110                  ;LOAD OUT DATA
5824  034140  005337  001276                DEC     $TMP0           ;ALL DONE?
5825  034144  001361                        BNE     1$              ;BR IF NOT
5826  034146  004737  033634        2$:     JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5827  034152  013761  034304  000004        MOV     SCHAR,4(R1)     ;LOAD PORT4
5828  034160  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5829  034162  122110                        122110                  ;LOAD OUT DATA
5830  034164  005737  034306                TST     STUFLG          ;BITSTUFF???
5831  034170  001407                        BEQ     6$              ;BR IF NO
5832  034172  013737  034304  034204        MOV     SCHAR,5$        ;IT IS SDLD SO CHECK BITSTUFFING
5833  034200  004537  035052                JSR     R5,STFFCL       ;ADD ANY BIT STUFF CLOCK TICKS
5834  034204  000000                5$:     0                       ;CHARACTER
5835  034206  000010                        10                      ;CHIFT COUNT
5836  034210  005237  034304        6$:     INC     SCHAR           ;NEXT CHARACTER
5837  034214  022737  000400  034304        CMP     #400,SCHAR      ;ALL DONE?
```

```
 5838  034222  001403                        BEQ     3$
 5839  034224  005337  001300                DEC     $TMP1           ;DECREMENT COUNT
 5840  034230  001346                        BNE     2$              ;BR IF NOT DONE
 5841  034232  004737  033502        3$:     JSR     PC,OCOR         ;WAIT FOR OCOR
 5842  034236  000207                        RTS     PC
 5843  034240  005037  034304        4$:     CLR     SCHAR           ;START PATTERN AT ZERO
 5844  034244  012737  177777  034306        MOV     #-1,STUFLG      ;SET BITSTUFF FLAG
 5845  034252  005037  035250                CLR     BITCON          ;CLEAR STUFF COUNT
 5846  034256  062737  000002  001300        ADD     #2,$TMP1        ;ADD 2 TO CHARACTER COUNT
 5847  034264  012761  000001  000004        MOV     #1,4(R1)        ;SET BIT0 IN PORT4
 5848  034272  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 5849  034274  122111                        122111                  ;SET SOM!
 5850  034276  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 5851  034300  122110                        122110                  ;LOAD GARBAGE CHAR
 5852  034302  000721                        BR      2$              ;GO LOAD SILO
 5853  034304  000000                SCHAR:  0
 5854  034306  000000                STUFLG: 0
 5855
 5856
 5857  034310                        INRDY:
 5858                                        ;THIS SUBROUTINE SPINS ON INRDY
 5859                                        ;IF INRDY FAILS TO SET THE DELAY TIMES OUT AND AN
 5860                                        ;ERROR IS REPORTED. FOR BETTER SCOPE LOOPS THIS
 5861                                        ;DELAY CAN BE MADE SHORTER BY ALTERING THE NUMBER
 5862                                        ;INITIALLY LOADED INTO $TMP0, THE SMALLER THE NUMBER
 5863                                        ;THE SHORTER THE DELAY. 0 IS THE LONGEST DELAY.
 5864
 5865  034310  012737  000000  001276        MOV     #0,$TMP0        ;SET UP DELAY COUNTER
 5866  034316                        1$:
 5867  034316  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 5868  034320  021244                        021244                  ;PORT4+LU12
 5869  034322  032777  000020  145544        BIT     #BIT4,@KMP04    ;IS INRDY SET?
 5870  034330  001004                        BNE     2$              ;BR IF YES
 5871  034332  005237  001276                INC     $TMP0           ;INC DELAY
 5872  034336  001367                        BNE     1$              ;TRY AGAIN
 5873  034340  104037                        ERROR   37              ;ERROR,NO INRDY
 5874  034342  000207                2$:     RTS     PC              ;RETURN
 5875
 5876
 5877  034344                        SIMBCC:
 5878                                        ;THIS SUBROUTINE CALCULATES THE CRC USING POLYNOMIAL GIVEN
 5879                                        ;IN XPOLY. THE CORRECT CRC IS $LPADRED IN CALBCC, AND THE
 5880                                        ;STATE OF THE LSB OF THE BCC IS $LPADRED IN THE C BIT.
 5881
 5882  034344  010046                        MOV     R0,-(SP)        ;SAVE R0 ON STACK
 5883  034346  012537  001276                MOV     (R5)+,$TMP0     ;$TMP0 = SHIFT COUNT
 5884  034352  012537  001300                MOV     (R5)+,$TMP1     ;$TMP1 = CHARACTER
 5885  034356  012537  034470                MOV     (R5)+,CALBCC    ;CALBCC = OLD BCC
 5886  034362  013700  034470        1$:     MOV     CALBCC,R0       ;PUT OLD BCC IN R0
 5887  034366  000241                        CLC
 5888  034370  006037  034470                ROR     CALBCC          ;SHIFT OLD BCC
 5889  034374  006037  001300                ROR     $TMP1           ;SHIFT CHARACTER
 5890  034400  005500                        ADC     R0              ;ADD CHAR CARRY TO OLD BCC
 5891  034402  006000                        ROR     R0              ;PUT BIT0 TO CARRY BIT
 5892  034404  103011                        BCC     2$              ;CARRY IS FEEDBACK BIT
 5893  034406  013700  034466                MOV     XPOLY,R0        ;IF FEEDBACK = 1
```

```
5894  034412  043700  034470          BIC     CALBCC,R0       ;EXCLUSIVLY OR XPOLY TO CALBCC
5895  034416  043737  034466  034470  BIC     XPOLY,CALBCC
5896  034424  050037  034470          BIS     R0,CALBCC
5897  034430  005337  001276  2$:     DEC     $TMP0           ;DEC SHIFT COUNT
5898  034434  001352                  BNE     1$              ;BR IF NOT DONE
5899  034436  012737  000001  001276  MOV     #1,$TMP0        ;GET SET TO INVERT BIT0
5900  034444  013700  034470          MOV     CALBCC,R0       ;PUT RESULT IN R0
5901  034450  006000                  ROR     R0              ;SHIFT BIT0 TO CARRY
5902  034452  005537  001276          ADC     $TMP0           ;INVERT CARRY TO BIT0 OF $TMP0
5903  034456  006037  001276          ROR     $TMP0           ;PUT INVERTED BIT IN CARRY
5904  034462  012600                  MOV     (SP)+,R0        ;RESTORE R0
5905  034464  000205                  RTS     R5              ;$LPADR
5906  034466  000000          XPOLY:  0
5907  034470  000000          CALBCC: 0
5908          000200          LRC8=200
5909          120001          CRC16=120001
5910          102010          CRC.CCITT=102010
5911
5912
5913  034472                  BCCLD:
5914                          ;THIS SUBROUTINE LOADS THE OUT SILO WITH 2 SYNCS
5915                          ;WITH SOM SET, AND ONE CHARACTER PASSED TO IT
5916                          ;WITH THE SOM BIT CLEAR (ENABLE CRC)
5917
5918  034472  013637  001300          MOV     @(SP)+,$TMP1    ;GET CHARACTER
5919  034476  062746  000002          ADD     #2,-(SP)        ;ADJUST STACK
5920  034502  012737  000002  001276  MOV     #2,$TMP0        ;SET FOR 2 SYNCS
5921  034510  012761  000026  000004  MOV     #26,4(R1)       ;LOAD PORT4
5922  034516  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5923  034520  122114                  122114                  ;LOAD SYNC REGISTER
5924  034522  004737  033634  1$:     JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5925  034526  012761  000001  000004  MOV     #1,4(R1)        ;LOAD PORT4
5926  034534  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5927  034536  122111                  122111                  ;SET SOM
5928  034540  012761  000026  000004  MOV     #26,4(R1)       ;LOAD PORT4
5929  034546  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5930  034550  122110                  122110                  ;LOAD OUT DATA
5931  034552  005337  001276          DEC     $TMP0           ;ALL DONE?
5932  034556  001361                  BNE     1$              ;BR IF NOT
5933  034560  004737  033634          JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5934  034564  013761  001300  000004  MOV     $TMP1,4(R1)     ;LOAD PORT4
5935  034572  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5936  034574  122110                  122110                  ;LOAD OUT DATA
5937  034576  004737  033502          JSR     PC,OCOR         ;WAIT FOR OCOR
5938  034602  000207                  RTS     PC
5939
5940
5941  034604                  GETQO:
5942                          ;THIS SUBROUTINE READS THE STATE OF THE TRANSMIT
5943                          ;BCC LSB AND PUTS IT IN THE CARRY BIT
5944
5945  034604  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5946  034606  021364                  021364                  ;PORT4←LU-17
5947  034610  106177  145260          ROLB    @KMP04          ;PUT Q0 IN CARRY
5948  034614  000207                  RTS     PC              ;RETURN
5949
```

```
5950
5951  034616                          GETQI:
5952                                          ;THIS SUBROUTINE READS THE STATE OF THE RECEIVE
5953                                          ;BCC LSB AND PUTS IT IN THE CARRY BIT
5954
5955  034616  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5956  034620  021364                        021364                  ;PORT4+LU-17
5957  034622  106177  145246                ROLB    @KMP04          ;PUT Q0 IN CARRY
5958  034626  106177  145242                ROLB    @KMP04          ;PUT QI IN CARRY
5959  034632  000207                        RTS     PC              ;RETURN
5960
5961
5962  034634                          SYNLD:
5963                                          ;THIS SUBROUTINE LOADS OUT SILO WITH
5964                                          ;2 SYNC CHARACTERS WITH SOM SET
5965
5966  034634  012737  000002  001276        MOV     #2,$TMP0        ;LOAD COUNTER FOR 2 SYNCS
5967  034642  012761  000026  000004        MOV     #26,4(R1)       ;PORT4+26
5968  034650  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5969  034652  122114                        122114                  ;LOAD SYNC REG
5970  034654  004737  033634          1$:   JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5971  034660  012761  000001  000004        MOV     #1,4(R1)        ;LOAD PORT4
5972  034666  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5973  034670  122111                        122111                  ;SET SOM
5974  034672  012761  000026  000004        MOV     #26,4(R1)       ;PORT+26
5975  034700  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5976  034702  122110                        122110                  ;LOAD OUT DATA WITH SYNC
5977  034704  005337  001276                DEC     $TMP0           ;DECREMENT COUNTER
5978  034710  001361                        BNE     1$              ;BR IF NOT DONE
5979  034712  000207                        RTS     PC              ;RETURN
5980
5981
5982  034714                          SOM:
5983                                          ;THIS SUBROUTINE LOADS SOM AND OUT DATA WITH A
5984                                          ;GARBAGE CHARACTER (0)
5985
5986  034714  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
5987  034720  012761  000001  000004        MOV     #1,4(R1)        ;PORT4+1
5988  034726  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5989  034730  122111                        122111                  ;SET SOM
5990  034732  005061  000004                CLR     4(R1)           ;CLEAR DATA CHAR
5991  034736  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5992  034740  122110                        122110                  ;LOAD GARBAGE CHARACTER
5993  034742  000207                        RTS     PC              ;RETURN
5994
5995
5996  034744                          EOM:
5997                                          ;THIS SUBROUTINE LOADS EOM AND OUT DATA WITH A
5998                                          ;GARBAGE CHARACTER (2) TO ENABLE TRANSMISSION OF BCC
5999
6000  034744  004737  033634                JSR     PC,OUTRDY       ;WAIT FOR OUTRDY
6001  034750  012761  000002  000004        MOV     #2,4(R1)        ;PORT4+2
6002  034756  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6003  034760  122111                        122111                  ;SET EOM
6004  034762  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6005  034764  122110                        122110                  ;LOAD GARBAGE CHARACTER
```

```
 6006  034766  000207                        RTS     PC              ;RETURN
 6007
 6008
 6009  034770                          MESLD:
 6010                                          ;THIS SUBROUTINE LOADS SILO WITH MESSAGE
 6011                                          ;THE FIRST ARUGUMENT IS THE ADDRESS OF THE MESSAGE
 6012                                          ;THE SECOND ARGUMENT IS THE NUMBER OF CHARACTERS IN THE MESSAGE
 6013
 6014  034770  010046                        MOV     R0,-(SP)        ;SAVE R0
 6015  034772  012500                        MOV     (R5)+,R0        ;R0=MESSAGE POINTER
 6016  034774  012537  001276                MOV     (R5)+,$TMP0     ;$TMP0=CHARACTER COUNT
 6017  035000  004737  033634          1$:   JSR     PC,OUTRDY       ;WAIT FOR OUT RDY
 6018  035004  112061  000004                MOVB    (R0)+,4(R1)     ;LOAD PORT4 WITH CHARACTER
 6019  035010  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 6020  035012  122110                        122110                  ;LOAD OUT DATA SILO
 6021  035014  005337  001276                DEC     $TMP0           ;DEC CHAR COUNT
 6022  035020  001367                        BNE     1$              ;BR IF NOT DONE
 6023  035022  004737  033502                JSR     PC,OCOR         ;WAIT FOR OCOR
 6024  035026  012600                        MOV     (SP)+,R0        ;RESTORE R0
 6025  035030  000205                        RTS     R5              ;RETURN
 6026
 6027
 6028  035032                          CLRIO:
 6029                                          ;THIS SUBROUTINE SETS IN CLR AND OUT CLR TO
 6030                                          ;CLEAR THE TRANSMIT AND RECEIVE BCC REGISTERS
 6031
 6032  035032  012761  000200  000004        MOV     #BIT7,4(R1)     ;LOAD PORT4
 6033  035040  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 6034  035042  122112                        122112                  ;SET IN CLR!
 6035  035044  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 6036  035046  122111                        122111                  ;SET OUT CLR!
 6037  035050  000207                        RTS     PC              ;RETURN
 6038
 6039
 6040  035052                          STFFCL:
 6041                                          ;THIS SUBROUTINE ADDS ANY NECESSSARY BIT STUFF CLOCK TICKS
 6042                                          ;FIRST ARGUMENT IS CHAR, SECOND ARGUMENT IS SHIFT COUNT.
 6043
 6044  035052  010046                        MOV     R0,-(SP)        ;SAVE R0
 6045  035054  012500                        MOV     (R5)+,R0        ;PUT CHAR IN R0
 6046  035056  012537  001302                MOV     (R5)+,$TMP2     ;PUT SHIFT COUNT IN $TMP2
 6047  035062  106000                  1$:   RORB    R0              ;LOOK AT NEXT BIT
 6048  035064  103403                        BCS     2$              ;BR IF A MARK
 6049  035066  005037  035250                CLR     BITCON          ;IT WAS A SPACE, CLEAR 1'S COUNTER
 6050  035072  000412                        BR      3$              ;CONTINUE
 6051  035074  005237  035250          2$:   INC     BITCON          ;INC CONSECUTIVE 1'S COUNTER
 6052  035100  022737  000005  035250        CMP     #5,BITCON       ;IS IT 5 YET?
 6053  035106  001004                        BNE     3$              ;BR IF NO
 6054  035110  005037  035250                CLR     BITCON          ;YES!  SO START AGAIN
 6055  035114  104413  000001                DATACLK,        1       ;GIVE EXTRA TICK TO STUFF ZERO
 6056  035120  005337  001302          3$:   DEC     $TMP2           ;DEC SHIFT COUNT
 6057  035124  001356                        BNE     1$              ;BR IF NOT DONE
 6058  035126  012600                        MOV     (SP)+,R0        ;RESTORE R0
 6059  035130  000205                        RTS     R5              ;RETURN
 6060
 6061
```

```
6062  035132                          STFFCK:
6063                                          ;THIS SUBROUTINE CHECKS TO SEE IF TRANSMITTER
6064                                          ;IS STUFFING ZEROS WHEN IT SHOULD. FIRST ARGUMENT
6065                                          ;IS THE CHARACTER, SECOND ARGUMENT IS SHIFT COUNT.
6066
6067  035132  010046                          MOV     R0,-(SP)        ;SAVE R0
6068  035134  012500                          MOV     (R5)+,R0        ;PUT CHAR IN R0
6069  035136  012537  001302                  MOV     (R5)+,$TMP2     ;PUT SHIFT COUNT IN $TMP2
6070  035142  106000                  1$:     RORB    R0              ;SHIFT OUT NEXT BIT
6071  035144  103403                          BCS     2$              ;BR IF IT IS A MARK
6072  035146  005037  035250                  CLR     BITCON          ;IT WAS A SPACE, CLEAR 1'S COUNTER
6073  035152  000416                          BR      3$              ;CONTINUE
6074  035154  005237  035250          2$:     INC     BITCON          ;INC CONSECUTIVE I'S COUNTER
6075  035160  022737  000005  035250          CMP     #5,BITCON       ;5 IN A ROW YET?
6076  035166  001010                          BNE     3$              ;BR IF NO
6077  035170  005037  035250                  CLR     BITCON          ;YES, SO START OVER
6078  035174  104413  000001                  DATACLK,        1       ;EXTRA TICK TO STUFF ZERO
6079  035200  004737  033450                  JSR     PC,GETSI        ;LOOK AT WINDOW
6080  035204  103001                          BCC     3$              ;IS IT A ZERO, BR IF YES
6081  035206  104030                          ERROR   30              ;NO, ERROR ZERO WAS NOT STUFFED
6082  035210  005337  001302          3$:     DEC     $TMP2           ;DEC SHIFT COUNT
6083  035214  001352                          BNE     1$              ;BR IF NOT DONE
6084  035216  012600                          MOV     (SP)+,R0        ;RESTORE R0
6085  035220  000205                          RTS     R5              ;RETURN
6086
6087
6088  035222                          CTSDLY:
6089                                          ;THIS SUBROUTINE WASTES TIME UNTIL CTS SETS,
6090                                          ;BUT HOPEFULLY NOT SO LONG THAT THE SILO RUNS OUT
6091
6092  035222  010046                          MOV     R0,-(SP)        ;SAVE R0
6093  035224  012700  000032                  MOV     #32,R0          ;LOAD R0 WITH COUNT
6094  035230  027777  144010  144006  1$:     CMP     @$TKS,@$TKS     ;WASTE TIME
6095  035236  005300                          DEC     R0              ;DECREMENT COUNTER
6096  035240  001373                          BNE     1$              ;DO IT AGAIN IF NOT = 0
6097  035242  012600                          MOV     (SP)+,R0        ;RESTORE R0
6098  035244  000207                          RTS     PC              ;RETURN
6099
6100
6101  035246  000176                  FLAG:   ^B<01111110>            ;FLAG CHARACTER
6102  035250  000000                  BITCON: 0
6103  035252     000     125     252  MESDAT: .BYTE   0,125,252,377
6104  035255     377
6105  035256     001     002     004  FLTDAT: .BYTE   1,2,4,10,20,40,100,200,376,375,373,367,357,337,277,177
6106  035261     010     020     040
6107  035264     100     200     376
6108  035267     375     373     367
6109  035272     357     337     277
6110  035275     177
6111  035276     100     140     160  STUFDT: .BYTE   100,140,160,170,3,300,174,176,177,1
6112  035301     170     003     300
6113  035304     174     176     177
6114  035307     001
6115  035310     363     347     317          .BYTE   363,347,317,200,0,377,377,377,200,37
6116  035313     200     000     377
6117  035316     377     377     200
```

```
 6118  035321     037
 6119                                .EVEN
 6120  035322  046200  047111  020105  EM1:    .ASCIZ  <200>/LINE UNIT INITALIZATION TEST/
       035360  046200  047111  020105  EM2:    .ASCIZ  <200>↑LINE UNIT REGISTER READ/ONLY TEST↑
       035423     200  044514  042516  EM3:    .ASCIZ  <200>↑LINE UNIT REGISTER WRITE/READ TEST↑
       035467     200  044514  042516  EM4:    .ASCIZ  <200>/LINE UNIT INTERNAL CLOCK FAILURE/
       035531     200  051124  047101  EM5:    .ASCIZ  <200>/TRANSMITTER DATA ERROR/
       035561     200  042522  042503  EM6:    .ASCIZ  <200>/RECEIVER TEST/
       035600  051200  041505  044505  EM7:    .ASCIZ  <200>/RECEIVER DATA ERROR/
       035625     200  047515  042504  EM10:   .ASCIZ  <200>/MODEM SIGNAL ERROR/
       035651     200  051124  047101  EM11:   .ASCIZ  <200>/TRANSMITTER CRC ERROR/
       035700  051200  041505  044505  EM12:   .ASCIZ  <200>/RECEIVER CRC ERROR/
       035724  044600  020116  041502  EM13:   .ASCIZ  <200>/IN BCC MATCH ERROR (LU REG 12)/
       035764  052200  040522  051516  EM14:   .ASCIZ  <200>/TRANSMITTER FAILED TO GO TO MARK STATE/
       036034  041600  041101  042514  EM15:   .ASCIZ  <200>/CABLE DATA TEST/
       036055     200  046106  043501  EM16:   .ASCIZ  <200>/FLAG ERROR/
       036071     200  051124  047101  EM17:   .ASCIZ  <200>/TRANSMITTER FAILED TO STUFF A ZERO/
       036135     200  053523  052111  EM20:   .ASCIZ  <200>/SWITCH PAC TEST/
       036156  040600  047502  052122  EM21:   .ASCIZ  <200>/ABORT ERROR/
       036173     200  051124  047101  EM22:   .ASCIZ  <200>/TRANSMITTER ERROR/
       036216  044200  046101  020106  EM23:   .ASCIZ  <200>/HALF DUPLEX TEST/
       036240  047600  052125  051040  EM24:   .ASCIZ  <200>/OUT READY NOT SET/
       036263     200  047111  051040  EM25:   .ASCIZ  <200>/IN READY NOT SET/

       036305     200  054105  042520  DH1:    .ASCIZ  <200>/EXPECTED  FOUND/
       036326  042600  050130  041505  DH2:    .ASCIZ  <200>/EXPECTED  FOUND  LU-REGISTER/
       036364  041600  040510  040522  DH3:    .ASCIZ  <200>/CHARACTER    BIT THAT FAILED/
       036422  041600  051117  042522  DH4:    .ASCIZ  <200>/CORRECT CRC    BIT THAT FAILED/
       036462  042600  050130  041505  DH5:    .ASCIZ  <200>/EXPECTED   FOUND   SHIFT/
       036514  042600  050130  041505  DH6:    .ASCIZ  <200>/EXPECTED   FOUND   CHARACTER   SHIFT/
       036562  041200  047514  045503  DH7:    .ASCIZ  <200>/BLOCK END NOT SET/
       036605     200  052122  020123  DH10:   .ASCIZ  <200>/RTS DID NOT CLEAR/
                                       .EVEN

       036630  000002                  DT1:    2
       036632     003     007                  .BYTE   3,7
       036634  001274                          $REG5
       036636     003     002                  .BYTE   3,2
       036640  001272                          $REG4
       036642  000003                  DT2:    3
       036644     003     007                  .BYTE   3,7
       036646  001274                          $REG5
       036650     003     010                  .BYTE   3,10
       036652  001272                          $REG4
       036654     003     002                  .BYTE   3,2
       036656  001266                          $REG2
       036660  000002                  DT3:    2
       036662     003     017                  .BYTE   3,17
       036664  001274                          $REG5
       036666     002     002                  .BYTE   2,2
       036670  001270                          $REG3
       036672  000002                  DT4:    2
       036674     006     021                  .BYTE   6,21
       036676  034470                          CALBCC
       036700     002     002                  .BYTE   2,2
       036702  001270                          $REG3
```

```
036704  000003                  DT5:    3
036706      001         011             .BYTE   1,11
036710  001462                          ZERO
036712      001         011             .BYTE   1,11
036714  001464                          ONE
036716      002         002             .BYTE   2,2
036720  001262                          $REGO
036722  000003                  DT6:    3
036724      001         011             .BYTE   1,11
036726  001464                          ONE
036730      001         011             .BYTE   1,11
036732  001462                          ZERO
036734      002         002             .BYTE   2,2
036736  001262                          $REGO
036740  000004                  DT7:    4
036742      001         011             .BYTE   1,11
036744  001462                          ZERO
036746      001         011             .BYTE   1,11
036750  001464                          ONE
036752      003         007             .BYTE   3,7
036754  001274                          $REG5
036756      002         001             .BYTE   2,1
036760  001270                          $REG3
036762  000004                  DT10:   4
036764      001         011             .BYTE   1,11
036766  001464                          ONE
036770      001         011             .BYTE   1,11
036772  001462                          ZERO
036774      003         007             .BYTE   3,7
036776  001274                          $REG5
037000      002         001             .BYTE   2,1
037002  001270                          $REG3
037004  000002                  DT11:   2
037006      003         007             .BYTE   3,7
037010  035246                          FLAG
037012      002         002             .BYTE   2,2
037014  001270                          $REG3
037016  000002                  DT12:   2
037020      006         004             .BYTE   6,4
037022  034470                          CALBCC
037024      006         002             .BYTE   6,2
037026  001302                          $TMP2

037030                          CORMAX:
        000001                  .END
```

```
ABASE = 000000            266     307
ACDW1 = 000000            266     309
ACDW2 = 000000            256     310
ACPUOP= 000000            266     281
ADDW0 = 000000            266     311
ADDW1 = 000000            266     312
ADDW10= 000000            266     321
ADDW11= 000000            266     322
ADDW12= 000000            266     323
ADDW13= 000000            266     324
ADDW14= 000000            266     325
ADDW15= 000000            266     326
ADDW2 = 000000            266     313
ADDW3 = 000000            266     314
ADDW4 = 000000            266     315
ADDW5 = 000000            266     316
ADDW6 = 000000            266     317
ADDW7 = 000000            266     318
ADDW8 = 000000            266     319
ADDW9 = 000000            266     320
ADEVCT= 000000            266     272
ADEVM = 000000            266     308
ADRCNT  006057           1410*   1425*   1434#
ADVANC= 104420           1579#   5597    5686
AENV  = 000002              1#    266     277
AENVM = 000000            266     278
AFATAL  000000            266     269
AMADR1= 000000            266     294
AMADR2= 000000            266     298
AMADR3= 000000            266     301
AMADR4= 000000            266     304
AMAMS1= 000000            266     288
AMAMS2= 000000            266     296
AMAMS3= 000000            266     299
AMAMS4= 000000            266     302
AMSGAD= 000000            266     274
AMSGLG= 000000            266     275
AMSGTY= 000000            266     268
AMTYP1= 000000            266     289
AMTYP2= 000000            266     297
AMTYP3= 000000            266     300
AMTYP4= 000000            266     303
APASS = 000000            266     271
APRIOR= 000000            266
APTCSU= 000040           1135    1240#
APTENV= 000001           1128    1196    1238#   1640
APTSIZ= 000200           1237#
APTSPO= 000100           1130    1198    1239#
APT.SI  013510            803    2214#
ASWREG= 000000            266     279
ATESTN= 000000            266     270
AUDONE  003354            840     861     900#
AUNIT = 000000            266     273
AUSTRT  003126            839#
AUSWR = 000000            266     280
AUTO.S  012110            801    1958#
```

```
AVECT1= 000000   266   305
AVECT2= 000000   266   306
BASE    013726  2225* 2236  2250* 2260#
BCCLD   034472  4090  4120  4174  4210  4264  4294  4348  4378  5913#
BINWRD  006412  1482* 1485* 1486  1523#
BITCUR  035250  2823* 2899* 2975* 3078* 3084* 3089* 3093  3095* 3130* 3163* 3212* 3249* 3835*
                3884* 4082* 4166* 4177* 4256* 4340* 4422* 4507* 4597* 4803* 5119* 5845* 6049*
                6051* 6052  6054* 6072* 6074* 6075  6077* 6102#
BIT0   = 000001  129#  1747  1748
BIT00  = 000001  119#  129
BIT01  = 000002  118#  128
BIT02  = 000004  117#  127
BIT03  = 000010  116#  126
BIT04  = 000020  115#  125
BIT05  = 000040  114#  124
BIT06  = 000100  113#  123
BIT07  = 000200  112#  122
BIT08  = 000400  111#  121
BIT09  = 001000  110#  120
BIT1   = 000002  128#   808  1741  1747  1748  2095
BIT10  = 002000  109#  2076
BIT11  = 004000  108#  1071  2672  2710  2759  2824  2900  2976  3048  3131  3213  3302  3377
                3437  3471  3505  3547  3583  3642  3690  3738  3786  3839  3934  4032  4083
                4167  4257  4341  4423  4508  4601  4735  4807  5123  5450  5519  5623
BIT12  = 010000  107#  2030  2093
BIT13  = 020000  106#  2033  2079  2097  2362  3976  3990  4005  4028  4051
BIT14  = 040000  105#  1053  2044  2046  2097  2108  3978  4030  5513  5617
BIT15  = 100000  104#   748
BIT2   = 000004  127#   808   949  1748
BIT3   = 000010  126#  2099  2106  4007
BIT4   = 000020  125#  1731  1754  1756  2795  3597  3656  3704  3752  3800  5711  5753  5869
BIT5   = 000040  124#  2211  2365  2733  2872  2948  3024  3185  3889  5467
BIT6   = 000100  123#  1736  1737  2101  3480  3522  3556
BIT7   = 000200  122#  1737  1856  2211  2767  3602  3992  4053  6032
BIT8   = 000400  121#  2088  2090  2103  2105  2111  2114  2171
BIT9   = 001000  120#  2085  2111  2114  2169  2171
BM      010631 1779#  2057
BPTVEC= 000014   136#
BRLVL   013442  2166  2176  2184  2197#
BRW     004360   954  1088#
CALBCC  034470  4109  4139  4199  4229  4283  4313  4367  4397  4472  4560  4629  4654* 4655
                4840  4865* 4866  4921  4946* 4947  5046  5089  5157  5182* 5183  5256  5281*
                5282  5381  5424  5531  5534* 5591  5635  5638* 5683  5885* 5886  5888* 5894
                5895* 5896* 5900  5907# 6120
CHAR    033666  3935  5761#
CHARSD  034000  3589  3648  3696  3744  3792  5788#
CHRCNT  006410 1480* 1483  1487  1503* 1521#  1522
CKSWR   011212   779  1094  1586  1804#
CKSWR1  011266 1816#  1828
CKSWR2  011300 1819#
CKSWR3  011304 1821#
CKSWR4  011310 1822#  1830  1837
CKSWR5  011414 1805   1807  1811  1846#
CLKX    001452   341#
CLRIO   035032  2671  2709  2758  2822  2898  2974  3047  3129  3211  3301  3376  3436  3470
                3504  3546  3581  3640  3688  3736  3784  3832  3881  3933  4081  4084  4114
```

```
                    4165    4168    4204    4255    4258    4288    4339    4342    4372    4421    4506    4596    4734
                    4802    5118    5449    5518    5622    6028#
CNERR   011027       873    1779#
CNT.MA  002302       355     628#    747     749     751    1884
CNVRT = 104417       870     979     981     983     985    1578#   1621    1623    1698    1819
CONERR  010762       868    1779#
CONN    010671      1779#   2035
CONTAB  003330       877     885#
CONVRT= 104416       820     876    1577#   1637    1980
CORMAX  037030      6120#
CR    = 000015       44#    1174    1184
CRC.CC= 102010      4086    4116    4170    4206    4260    4290    4344    4374    4429    4514    4623    4834    4915
                    5151    5250    5521    5625    5910#
CRC16 = 120001      5909#
CREAM   001502       354#    746*   1880*   1881    1883*   1887
CRLF  = 000200        45#    1145    1184
CSR     010363      1779#   1984
CSRMAP  012112      1960#
CTSDLY  035222      6088#
CYCLE   011460       955    1016    1871#
DATABP  006764      1610*   1613    1635    1638#
DATACL= 104413      1574#   2717    2766    2770    2839    2841    2855    2869    2915    2917    2931    2945    2991
                    2993    3007    3021    3061    3063    3081    3096    3147    3149    3164    3182    3233    3235
                    3253    3319    3322    3340    3349    3393    3394    3403    3441    3475    3517    3551    3591
                    3650    3698    3746    3794    3841    3842    3893    3894    3940    4092    4093    4122    4123
                    4176    4179    4212    4213    4266    4267    4296    4297    4350    4351    4380    4381    4443
                    4447    4528    4533    4615    4630    4657    4680    4697    4704    4706    4746    4826    4841
                    4868    4891    4922    4949    4972    4989    4996    4998    5009    5143    5158    5185    5208
                    5226    5257    5284    5307    5324    5331    5333    5344    5459    5464    5470    6055    6078
DATAHD  006752      1609*   1631    1634#
DDISP = 177570        51#     207#    236
DELAY = 104411      1572#
DEVTAB  003342       843     890#
DH1     036305       404     407     410     446     452     458     485    6120#
DH10    036605       467    6120#
DH2     036326       386     389     392     398     470     479    6120#
DH3     036364       401     449     464    6120#
DH4     036422       419    6120#
DH5     036462       413     416     425     428     482    6120#
DH6     036514       431     434     437     440    6120#
DH7     036562       461    6120#
DISPLA  001242       236#    765*    771*   1083*
DISPRE  000174       197#    771
DSWR  = 177570        50#     206#    235
DT1     036630       405     408     411     447     459     486    6120#
DT10    036762       435     441    6120#
DT11    037004       450    6120#
DT12    037016       453    6120#
DT2     036642       387     390     393     399     471     480    6120#
DT3     036660       402     465    6120#
DT4     036672       420    6120#
DT5     036704       414     417     483    6120#
DT6     036722       426     429    6120#
DT7     036740       432     438    6120#
DZDME = 000000       382
DZDMG = ****** U     382     487
```

```
EMTVEC= 000030  139#
EM1     035322  385   6120#
EM10    035625  409   6120#
EM11    035651  412   418   424   430   433   6120#
EM12    035700  415   427   436   439   451   6120#
EM13    035724  421   484   6120#
EM14    035764  442   6120#
EM15    036034  445   6120#
EM16    036055  448   6120#
EM17    036071  454   6120#
EM2     035360  388   6120#
EM20    036135  457   6120#
EM21    036156  460   463   6120#
EM22    036173  466   6120#
EM23    036216  469   6120#
EM24    036240  472   6120#
EM25    036263  475   6120#
EM3     035423  391   6120#
EM4     035467  394   6120#
EM5     035531  397   400   481   6120#
EM6     035561  403   478   6120#
EM7     035600  406   6120#
EOM     034744  4611  4612  4743  4744  4817  4818  4822  4823  5133  5134  5139  5140  5538
                5539  5543  5544  5548  5549  5642  5643  5996#
ERCT00  002304  630#
ERCT01  002310  633#
ERCT02  002314  636#
ERCT03  002320  639#
ERCT04  002324  642#
ERCT05  002330  645#
ERCT06  002334  648#
ERCT07  002340  651#
ERCT10  002344  654#
ERCT11  002350  657#
ERCT12  002354  660#
ERCT13  002360  663#
ERCT14  002364  666#
ERCT15  002370  669#
ERCT16  002374  672#
ERCT17  002400  675#
ERR     003244  852   865#   869
ERRMSG  006740  1608* 1626   1629#
ERRPC   003322  871   882#
ERRVEC= 000004  132#  1058   1059*  1061*  1064*
ERTABO  007112  1623  1663#
EXIT  = 000205  159#
EXITER  007046  1649  1654#
FLAG    035246  6101# 6120
FLOAT   003156  845#  851
FLTDAT  035256  5525  5536  5541  5546  5554  5594  6105#
FY      003202  853#  857   862
GETQI   034616  4130  4134  4220  4224  4304  4308  4388  4392  4542  4546  5951#
GETQO   034604  4100  4104  4189  4193  4274  4278  4358  4362  4456  4460  5941#
GETSI   033450  2844  2848  2858  2862  2920  2924  2934  2938  2996  3000  3010  3014  3066
                3070  3085  3090  3097  3152  3156  3167  3171  3238  3242  3256  3260  3325
                3329  3341  3350  3395  3406  3410  4633  4637  4660  4664  4683  4687  4698
```

```
                    4707    4844    4848    4871    4875    4894    4898    4925    4929    4952    4956    4975    4979
                    4990    4999    5161    5165    5188    5192    5211    5215    5229    5233    5260    5264    5287
                    5291    5310    5314    5325    5334    5692#   6079
GNS    = ****** U   1562    1565    1566    1567    1568    1569    1570    1571    1572    1573    1574    1575    1576
                    1577    1578    1579
HALTS   006770      1593    1640#
HILIM   006052      1407*   1415    1431#
HT     = 000011       42#   1143    1184
INCHAR  011420      1822    1850#
INIFLG  001506       359#    774     804     811*
INLPI   005766      1412#   1421
INPUT  = 104415     1576#   1920    1971    1983    1991    2048    2056
INRDY   034310      3843    3895    3941    3949    4747    4760    4765    5012    5026    5031    5055    5069    5074
                    5347    5361    5366    5390    5404    5409    5471    5481    5555    5571    5576    5648    5663
                    5668    5857#
INTTY   013456      2001    2020    2036    2205#
IOTVEC= 000020       137#
KMACTV  001470       348#    788     913*    914     1871    1885    1966*   2126*   2132*   2133*   2137    2162    2228*
                    2229
KMCM    011050       879    1779#
KMCR00  002100       543#
KMCR01  002110       548#
KMCR02  002120       553#
KMCR03  002130       558#
KMCR04  002140       563#
KMCR05  002150       568#
KMCR06  002160       573#
KMCR07  002170       578#
KMCR10  002200       583#
KMCR11  002210       588#
KMCR12  002220       593#
KMCR13  002230       598#
KMCR14  002240       603#
KMCR15  002250       608#
KMCR16  002260       613#
KMCR17  002270       618#
KMCSR   002066       526#    839*    854     860*    889     1019    1086    1701    1755    1889*   1898    1946    2278*
KMCSRH  002070       527#   1736*   1737*   1741*   1747*   1748*   1754*   1756*   1898*   1899*   1900
KMCTL   002072       528#   1900*   1901*   1902
KMNUM   001472       349#    742     996    1964*   1978*   2118*   2119    2127    2129
KMP04   002074       529#   1725*   1731    1768    1773    1902*   1903*   1904    5698    5711    5753    5869    5947*
                    5957*   5958*
KMP06   002076       530#   1742*   1904*   1905*
KMRLVL  002060       523#    989     990*   1907*   1908*   1909
KMRVEC  002056       522#    989*   1022    1890*   1891*   1907
KMS100  002102       544#
KMS101  002112       549#
KMS102  002122       554#
KMS103  002132       559#
KMS104  002142       564#
KMS105  002152       569#
KMS106  002162       574#
KMS107  002172       579#
KMS110  002202       584#
KMS111  002212       589#
KMS112  002222       594#
```

```
KMS113  002232            599#
KMS114  002242            604#
KMS115  002252            609#
KMS116  002262            614#
KMS117  002272            619#
KMS200  002104            545#
KMS201  002114            550#
KMS202  002124            555#
KMS203  002134            560#
KMS204  002144            565#
KMS205  002154            570#
KMS206  002164            575#
KMS207  002174            580#
KMS210  002204            585#
KMS211  002214            590#
KMS212  002224            595#
KMS213  002234            600#
KMS214  002244            605#
KMS215  002254            610#
KMS216  002264            615#
KMS217  002274            620#
KMS300  002106            546#
KMS301  002116            551#
KMS302  002126            556#
KMS303  002136            561#
KMS304  002146            566#
KMS305  002156            571#
KMS306  002166            576#
KMS307  002176            581#
KMS310  002206            586#
KMS311  002216            591#
KMS312  002226            596#
KMS313  002236            601#
KMS314  002246            606#
KMS315  002256            611#
KMS316  002266            616#
KMS317  002276            621#
KMTLVL  002064            525#    991     992*   1911*  1912*
KMTVEC  002062            524#    991*   1909*   1910*  1911
KM.END  002300            623#   1962    2233
KM.MAP  002100            354     542#    746     813    823    1881   1883   1960   1965   2154   2231   2235
LF    =  000012             43#   1178    1184
LINE    010573           1779#   2049
LOBITS  006056           1409*   1433#
LOCK    001444            335#   1085*   1097    1099   1440*  1617   2384*  2400*  2427*  2443*  2470*  2493*  2529*
                         2549*   4075*   4113*   4159*  4203*  4249*  4287*  4333*  4371*
LOKFLG  001510            361#
LOLIM   006050           1406*   1417    1430#
LRCB    000200           5908#
LUTYPE= 000000              1    5877
MASKX   001454            342#
MASTEK  010015           1619    1779#
MCSRX   007745            978    1779#
MDATA   011150           1501    1511    1796#
MEMLIM  001466            347#    937*
MEPASS  007620            977    1779#
```

```
MERRPC  010072          1622    1779#
MERRX   007772           984    1779#
MERR2   007645          1779#   2139
MERR3   007672           910    1779#
MESDAT  035252          4041    4602    4609    4736    4741    4808    4815    4820    4906    5011    5054    5124    5131
                        5137    5241    5346    5389    5451    5456    5461    5629    5640    5647    6103#
MESLD   034770          3055    4040    4608    4740    4814    4819    5130    5136    5455    5460    5535    5540    5545
                        5639    6009#
MILK    001504           355#    747*    986    1879*   1884*   1888
MLOCK   007716           951    1779#
MNEW    010017           905    1779#
MODU    010461          1779#   2019
MPASSX  007761           982    1779#
MPFAIL  007562          1697    1714    1779#
MR      007642           958    1779#   1936
MRESET= 004000           159#
MSTCLR= 104410          1571#   1703    2329    2355    2386    2429    2472    2531    2583    2607    2631    2667    2705
                        2754    2818    2894    2970    3043    3125    3207    3297    3372    3432    3466    3500    3542
                        3577    3636    3684    3732    3780    3828    3877    3929    3975    4027    4077    4161    4251
                        4335    4417    4502    4592    4730    4798    5114    5445    5512    5616
MTITLE  001000           205#    778
MTSTN   010003          1620    1779#   1921
MVECX   007753           980    1779#
NEXT    001442           334#    1439    1659    2276*   2302*   2327*   2353*   2383*   2426*   2469*   2528*   2581*   2605*
                        2629*   2665*   2703*   2752*   2816*   2892*   2968*   3041*   3123*   3205*   3295*   3370*   3430*
                        3464*   3498*   3540*   3575*   3634*   3682*   3730*   3778*   3826*   3875*   3927*   3973*   4025*
                        4074*   4158*   4248*   4332*   4415*   4500*   4590*   4728*   4796*   5112*   5443*   5510*   5614*
NITCH   033500          5698*   5699*   5700*   5701*   5703#
NOACT   010731           790    1779#   1873
NODEV   003240           841     863#
NUM     010323          1779#   1972
OCOR    033502          2716    2765    2836    2912    2988    3144    3232    3317    3391    3516    4442    4527    4613
                        4745    4824    5141    5458    5463    5550    5644    5706#   5712    5742    5784    5800    5841
                        5937    6023
OK      003220           855     858#    881
ONE     001464           346#   6120
OUTRDY  033634          2825    2832    2901    2908    2977    2984    3049    3132    3140    3217    3223    3228    3272
                        3305    3313    3378    3507    3937    4607    4739    4813    5129    5454    5726    5735    5746#
                        5771    5780    5793    5797    5817    5826    5924    5933    5970    5986    6000    6017
PACT00  002302           629#
PACT01  002306           632#
PACT02  002312           635#
PACT03  002316           638#
PACT04  002322           641#
PACT05  002326           644#
PACT06  002332           647#
PACT07  002336           650#
PACT10  002342           653#
PACT11  002346           656#
PACT12  002352           659#
PACT13  002356           662#
PACT14  002362           665#
PACT15  002366           668#
PACT16  002372           671#
PACT17  002376           674#
PARBIT= 040000           159#
```

```
PERFOR= 004537  159#
PFTAB   007324  1698    1720#
PIRQ    = 177772  49#
PIRQVE= 000240  143#
POPRO   = 012600  154#  1653
POP1SP= 005726  152#
POP2SP= 022626  156#
PRIO    010422  1779#  2000
PRIRTY  013730  2221*  2227*  2240    2261#
PRO     = 000000  66#  2197    2198
PR1     = 000040  67#
PR2     = 000100  68#
PR3     = 000140  69#
PR4     = 000200  70#  2199
PR5     = 000240  71#  2200
PR6     = 000300  72#  2201
PR7     = 000340  73#  178     180     182     184     2202
PS      = 177776  46#  47      739*    948*    2166*   2176*
PSW     = 177776  47#
PUSHRO= 010046  153#  1650
PUSH1S= 005746  151#
PUSH2S= 024646  155#
PWRVEC= 000024  138#  1676*  1677*  1686*  1692*  1711*  1712*
QV.FLG  001511  362#  745*    995*
RDCHR   = 104402  1292    1565#
RDLIN   = 104403  1365    1566#
RDOCT   = 104404  1414    1567#
RESREG  006766  1636    1639#
RESVEC= 000010  133#
RESOS   = 104407  1570#  1639
ROMCLK= 104412  1573#  1726    1729    1766    1771    2280    2305    2331    2357    2389    2391    2402    2404
                2432    2434    2445    2447    2478    2480    2499    2501    2536    2538    2554    2556    2584
                2608    2634    2643    2669    2674    2676    2680    2707    2712    2714    2719    2729    2756
                2761    2763    2768    2772    2782    2792    2820    2827    2829    2834    2870    2896    2903
                2905    2910    2946    2972    2979    2981    2986    3022    3045    3051    3053    3127    3134
                3136    3142    3183    3209    3219    3221    3225    3230    3274    3299    3307    3309    3315
                3374    3380    3382    3385    3387    3389    3434    3442    3468    3476    3502    3509    3511
                3514    3518    3544    3552    3579    3585    3587    3593    3603    3605    3638    3644    3646
                3652    3661    3686    3692    3694    3700    3709    3734    3740    3742    3748    3757    3782
                3788    3790    3796    3805    3830    3844    3879    3890    3896    3931    3938    3942    3950
                3982    3985    3997    4000    4034    4038    4046    4079    4163    4253    4337    4419    4432
                4436    4440    4481    4504    4517    4521    4525    4569    4594    4732    4748    4761    4766
                4775    4800    5013    5027    5032    5041    5056    5070    5075    5084    5116    5348    5362
                5367    5376    5391    5405    5410    5419    5447    5465    5472    5482    5516    5556    5572
                5577    5586    5620    5649    5664    5669    5678    5696    5709    5724    5728    5731    5737
                5740    5751    5769    5773    5776    5782    5795    5798    5815    5819    5822    5828    5848
                5850    5867    5922    5926    5929    5935    5945    5955    5968    5972    5975    5988    5991
                6002    6004    6019    6033    6035
RUN     001500  352#  748*  1877*  1878*  1885
SAVACT  001474  350#  908  2137*  2229*
SAVNUM  001476  351#  742*  993*  996*  2130*  2223*
SAVPC   001460  344#  869*  884  1446*  1665
SAVSP   001456  343#
SAVOS   = 104406  1569#  1598
SCHAR   034304  3833*  3834*  3882*  3883*  5809  5827  5832  5836*  5837  5843*  5853#
SCOP1   = 104405  1568#  2399  2412  2442  2455  2489  2510  2545  2563  4112  4142  4202  4232
```

```
                    4286    4316    4370    4400
SILOLD  034044      3840    3856    3892    3908    5804#
SIMBCC  034344      4095    4125    4184    4215    4269    4299    4353    4383    4451    4537    4625    4836    4917
                    5153    5252    5527    5631    5877#
SOFTSW  011452      1820    1859#
SOM     034714      5135    5982#
SPACNT= 006411      1481*   1505    1508*   1522#
STACK = 001200        37#    740     927    1660
STAT    001450       340#
STAT1   002050       515#   1892*   2362    3976    3978    3990    4005    4028    4030    4051    5513    5617
STAT2   002052       516#   1893*   2587    2611
STAT3   002054       517#   1894*
STFFCK  035132      3175    3264    4180    4448    4534    4641    4668    4852    4879    4933    4960    5169    5196
                    5268    5295    6062#
STFFCL  035052      5833    6040#
STKLMT= 177774        48#
STRTSW  001446       339#    780*    783*    784     786     800*    806     808     903     949     956    1914    1937*
                    1967    2150
STUFDT  035276      3056    3058    6111#
STUFLG  034306      3836*   3885*   5830    5844*   5854#
SVOS    006100      1450#
SWFLG   011416       743*   1815*   1842*   1848#
SWR     001240       235#    764*    766     770*    780     908     913    1053    1071    1095    1587    1592    1648
                    1655    1657    1684    1704*   1744    1804    1841*
SWREG   000176       198#    759*    770     1804    1861
SW0     = 000001     101#
SW00    = 000001      91#     101     784     800    1967    2150
SW01    = 000002      90#     100     956    1914    1937
SW02    = 000004      89#      99
SW03    = 000010      88#      98     903
SW04    = 000020      87#      97
SW05    = 000040      86#      96
SW06    = 000100      85#      95    1744
SW07    = 000200      84#      94
SW08    = 000400      83#      93    1655
SW09    = 001000      82#      92    1095
SW1     = 000002     100#
SW10    = 002000      81#    1657
SW11    = 004000      80#
SW12    = 010000      79#    1587
SW13    = 020000      78#    1592
SW14    = 040000      77#
SW15    = 100000      76#
SW2     = 000004      99#
SW3     = 000010      98#
SW4     = 000020      97#
SW5     = 000040      96#
SW6     = 000100      95#
SW7     = 000200      94#
SW8     = 000400      93#
SW9     = 001000      92#
SYNC    033520      3439    3473    3549    5716#
SYNLD   034634      4430    4515    4606    4738    4812    5128    5453    5520    5624    5962#
TBITVE= 000014       134#
TEMP    011106      1488    1752*   1757*   1763*   1775*   1794#   2632*   2638*   2641*   2647*
TIMER = 104414      1575#   2678    3592    3651    3699    3747    3795    3984    3999    4036
```

```
TKVEC = 000060            141#
TLAST = 033104           1940    5687#
TPVEC = 000064            142#
TRAPVE= 000034            140#
TRTVEC= 000014            135#
TST1    013732           1047    1927    1945    2274#
TST10   014730           2469    2526#
TST11   015104           2528    2579#
TST12   015146           2581    2603#
TST13   015210           2605    2627#
TST14   015310           2629    2663#
TST15   015424           2665    2701#
TST16   015576           2703    2750#
TST17   016010           2752    2814#
TST2    014006           2276    2300#
TST20   016272           2816    2890#
TST21   016554           2892    2966#
TST22   017036           2968    3039#
TST23   017346           3041    3121#
TST24   017654           3123    3203#
TST25   020206           3205    3293#
TST26   020474           3295    3368#
TST27   020714           3370    3428#
TST3    014054           2302    2325#
TST30   021016           3430    3462#
TST31   021122           3464    3496#
TST32   021260           3498    3538#
TST33   021364           3540    3573#
TST34   021570           3575    3632#
TST35   021736           3634    3680#
TST36   022106           3682    3728#
TST37   022256           3730    3776#
TST4    014126           2327    2351#
TST40   022426           3778    3824#
TST41   022602           3826    3873#
TST42   022766           3875    3925#
TST43   023130           3927    3971#
TST44   023326           3973    4023#
TST45   023506           4025    4072#
TST46   024052           4074    4156#
TST47   024444           4158    4246#
TST5    014220           2353    2381#
TST50   025010           4248    4330#
TST51   025354           4332    4413#
TST52   025676           4415    4498#
TST53   026234           4500    4588#
TST54   026736           4590    4726#
TST55   027160           4728    4794#
TST56   030500           4796    5110#
TST57   032240           5112    5441#
TST6    014360           2383    2424#
TST60   032460           5443    5508#
TST61   033104           5510    5612#   5687
TST62 = ****** U         5614
TST7    014520           2426    2467#
TTST    004146            952*    954*   1049#
```

F12


```
TWOSYN= 010000          1596
TYPDAT  006754          1614    1632    1635#
TYPE  = 104401           778     790     812     867     872     878     905     910     951     958     977     978     980
                         982     984    1148    1299    1305    1310    1314    1319    1320    1322    1325    1329    1394
                        1396    1412    1419    1420    1471    1511    1562#   1615    1616    1619    1620    1622    1624
                        1628    1633    1697    1713    1818    1821    1873    1919    1936    1942    1979    1999    2013
                        2018    2027    2034    2041    2139
TYPMSG  006654          1612    1615#
VEC     010401          1779#   1992
VECMAP  013172          2138    2150#
VECTR   013724          2220*   2226*   2245    2251*   2259#
WHAT    005770          1405*   1413#
WHERE   006054          1408*   1422    1432#
WHICH   013164          1981    2146#
WRDCNT  006406          1479*   1512*   1520#
WRKO.F  006742          1627    1630#
XBX     006542          1588    1590    1592#
XCSR    004104           979    1017#
XERR    004126           985    1026#
XHEAD   010077           812    1779#
XPASS   004120           983    1023#
XPOLY   034466          4086*   4116*   4170*   4206*   4260*   4290*   4344*   4374*   4429*   4514*   4623*   4834*   4915*
                        5151*   5250*   5521*   5625*   5893    5895    5906#
XSTATQ  011060           821    1779#
XTSTN   007120          1621    1666#
XVEC    004112           981    1020#
ZERO    001462           345#   6120
$APTHD  002034           497     503#
$ASTAT= ****** U        1218    1233
$ATYC   004722          1189    1191#
$ATY1   004676          1187#
$ATY3   004704          1133    1188#
$ATY4   004714          1190#   1643
$AUTOB  001234           232#
$BASE   001372           307#   2225
$BOADR  001222           227#
$BOOAT  001226           229#
$CDW1   001376           309#   2222
$CDW2   001400           310#
$CHARC  004672          1150*   1160*   1167    1176*   1181#
$CKSWR= ****** U        1565
$CMTAG  001200           215#
$CM1  = 000006           247#    248#    249#    250#    251#    252#    253#
$CM2  = 000014           247#    248#    249#    250#    251#    252#    253#
$CM3  = 000006           245#    247
$CM4  = 000005           253#    254#    255#    256#    257#    258#
$CNTLG  005534          1340#
$CNTLU  005527          1314    1339#
$COD  = ****** U           1
$CPUOP  001344           281#
$CRAP = 177777             1#   2263#   2266    2269#   2289#   2292    2295#   2314#   2317    2320#   2340#   2343    2346#
                        2370#   2373    2376#   2413#   2416    2419#   2456#   2459    2462#   2515#   2518    2521#   2568#
                        2571    2574#   2592#   2595    2598#   2616#   2619    2622#   2651#   2654    2658#   2689#   2692
                        2696#   2738#   2741    2745#   2800#   2803    2809#   2876#   2879    2885#   2952#   2955    2961#
                        3028#   3031    3034#   3107#   3110    3116#   3189#   3192    3198#   3280#   3283    3288#   3356#
                        3359    3363#   3417#   3420    3423#   3451#   3454    3457#   3485#   3488    3491#   3527#   3530
```

```
                    3533#  3561#  3564   3568#  3621#  3624   3627#  3669#  3672   3675#  3717#  3720   3723#
                    3765#  3768   3771#  3813#  3816   3819#  3860#  3863   3868#  3912#  3915   3920#  3959#
                    3962   3966#  4012#  4015   4018#  4060#  4063   4067#  4144#  4147   4151#  4234#  4237
                    4241#  4318#  4321   4325#  4402#  4405   4408#  4487#  4490   4493#  4575#  4578   4583#
                    4714#  4717   4721#  4778#  4781   4789#  5093#  5096   5105#  5428#  5431   5436#  5493#
                    5496   5503#  5598#  5601   5607#
$CRLF   001313       260#  1149   1184   1319   1339   1399   1420   1471   1615   1616   1624   1919   1979
$DDH0   001402       311#  2230
$DDH1   001404       312#
$DDH10  001426       321#
$DDH11  001430       322#
$DDH12  001432       323#
$DDH13  001434       324#
$DDH14  001436       325#
$DDH15  001440       326#
$DDH2   001406       313#
$DDH3   001410       314#
$DDH4   001412       315#
$DDH5   001414       316#
$DDH6   001416       317#
$DDH7   001420       318#
$DDH8   001422       319#
$DDH9   001424       320#
$DEVCT  001326       272#
$DEVM   001374       308#  2228
$DOAGN  004100       994   1003   1008   1014#
$ENDAD  004070       191    776   1010#  1646
$ENDCT  004054      1005#
$ENV    001336       277#   757    793    795   1128   1196   1220   1640   1806
$ENVM   001337       278#   798   1130   1135   1198
$EOP    003662       973#  5614
$EOPCT  004046      1002#  1006
$ERFLG  001203       218#   744*   976*  1038   1067   1069*  1090   1597*  1611   1625*  1699*
$ERMAX  001215       224#  1090
$ERROR  006512       181   1586#
$ERRPC  001216       225#   753*   997*  1046*  1594   1596*  1700*
$ERRTB  001512       379#  1607
$ERTTL  001212       222#   988   1028   1654*  1896*
$ETABL  001336       276#
$ETEND  001442       329#   509
$FATAL  001320       269#  1224*
$FFLG   005142      1187*  1190*  1218   1227*  1235#
$FILLC  001256       243#  1153   1184
$FILLS  001255       242#  1184
$GOADR  001220       226#
$GODAT  001224       228#
$GET42  004060      1007#
$GTSWR= ****** U    1564
$HD   = 000000        11
$HIBTS  002034       504#
$HIOCT  005722      1387*  1398#
$ICNT   001204       219#  1075*  1076   1078*  1089
$ILLUP  007316      1676   1692   1716#
$INPUT  005724      1403#  1576
$INTAG  001235       233#
$ITEMB  001214       223#  1602*  1642
```

```
SLF     001314          261#   1184    1329    1339    1399
SLFLG   005141         1228*   1234#
SLPADR  001206          220#    755*    955*    959    1082*   1084    1089    1659*   1661    1935*   1945*   1947
SLPERR  001210          221#
SMADR1  001350          294#
SMADR2  001354          298#
SMADR3  001360          301#
SMADR4  001364          304#
SMAIL   001316          267#    505     509    1081    1128
SMAMS1  001346          288#   2224
SMAMS2  001352          296#
SMAMS3  001356          299#
SMAMS4  001362          302#
SMBADR  002036          505#
SMFLG   005140         1188*   1194    1229*   1233#
SMNEW   005552         1343#   1821
SMSGAD  001332          274#   1204*   1207
SMSGLG  001334          275#   1209*
SMSGTY  001316          268#   1202    1210*   1222    1226*
SMSWR   005541         1341#   1818
SMTYP1  001347          289#
SMTYP2  001353          297#
SMTYP3  001357          300#
SMTYP4  001363          303#
SMXCNT  004362         1079    1089#
$N      = 000061          1#   2263    2269    2271    2278#   2289    2295    2297    2304#   2314    2320    2322    2329
                       2330#   2340    2346    2348    2355    2356#   2370    2376    2378    2386    2387#   2413    2419
                       2421    2429    2430#   2456    2462    2464    2472    2473#   2515    2521    2523    2531    2532#
                       2568    2574    2576    2583    2584#   2592    2598    2600    2607    2608#   2616    2622    2624
                       2631    2632#   2651    2658    2660    2667    2668#   2689    2696    2698    2705    2706#   2738
                       2745    2747    2754    2755#   2800    2809    2811    2818    2819#   2876    2885    2887    2894
                       2895#   2952    2961    2963    2970    2971#   3028    3034    3036    3043    3044#   3107    3116
                       3118    3125    3126#   3189    3198    3200    3207    3208#   3280    3288    3290    3297    3298#
                       3356    3363    3365    3372    3373#   3417    3423    3425    3432    3433#   3451    3457    3459
                       3466    3467#   3485    3491    3493    3500    3501#   3527    3533    3535    3542    3543#   3561
                       3568    3570    3577    3578#   3621    3627    3629    3636    3637#   3669    3675    3677    3684
                       3685#   3717    3723    3725    3732    3733#   3765    3771    3773    3780    3781#   3813    3819
                       3821    3828    3829#   3860    3868    3870    3877    3878#   3912    3920    3922    3929    3930#
                       3959    3966    3968    3975    3976#   4012    4018    4020    4027    4028#   4060    4067    4069
                       4077    4078#   4144    4151    4153    4161    4162#   4234    4241    4243    4251    4252#   4318
                       4325    4327    4335    4336#   4402    4408    4410    4417    4418#   4487    4493    4495    4502
                       4503#   4575    4583    4585    4592    4593#   4714    4721    4723    4730    4731#   4778    4789
                       4791    4798    4799#   5093    5105    5107    5114    5115#   5428    5436    5438    5445    5446#
                       5493    5503    5505    5512    5513#   5598    5607    5609    5616    5617#   5687#
$NULL   001254          241#   1155    1184
$NWTST= 000000         2273#   2299#   2324#   2350#   2380#   2423#   2466#   2525#   2578#   2602#   2626#   2662#   2700#
                       2749#   2813#   2889#   2965#   3038#   3120#   3202#   3292#   3367#   3427#   3461#   3495#   3537#
                       3572#   3631#   3679#   3727#   3775#   3823#   3872#   3924#   3970#   4022#   4071#   4155#   4245#
                       4329#   4412#   4497#   4587#   4725#   4793#   5109#   5440#   5507#   5611#
$OVER   004334         1051    1054    1065    1077    1083#
$PASS   001324          271#    975*    987     999*   1000*   1017    1025    1073    1090    1895*
$PASTM  002042          507#
$PWRDN  007126          179     741    1676#   1711
$PWRMG  007312         1714#
$PWRUP  007200         1686    1692#
$QUES   001312          259#   1184    1322    1339    1396    1399    1419    1942    2014    2028    2042
```

```
$RDCHR  005144          1259#   1565
$RDDEC= ****** U        1568
$RDLIN  005264          1287#   1566
$RDOCT  005564          1360#   1567
$RDSZ = 000007          1280#
$REGAD  001260           245#
$REG0   001262           247#   1455*   1460    6120
$REG1   001264           248#    875*    887    1454*   1461
$REG2   001266           249#   1453*   1462    6120
$REG3   001270           250#   1452*   1463    6120
$REG4   001272           251#   1451*   1464    6120
$REG5   001274           252#   1450*   1465    6120
$RTNAD  004102          1016#
$R2A  = ****** U        1568
$S    = 000063             1#   2276    2278#   2302    2304#   2327    2330#   2353    2356#   2383    2387#   2426    2430#
                        2469    2473#   2528    2532#   2581    2584#   2605    2608#   2629    2632#   2665    2668#   2703
                        2706#   2752    2755#   2816    2819#   2892    2895#   2968    2971#   3041    3044#   3123    3126#
                        3205    3208#   3295    3298#   3370    3373#   3430    3433#   3464    3467#   3498    3501#   3540
                        3543#   3575    3578#   3634    3637#   3682    3685#   3730    3733#   3778    3781#   3826    3829#
                        3875    3878#   3927    3930#   3973    3976#   4025    4028#   4074    4078#   4158    4162#   4248
                        4252#   4332    4336#   4415    4418#   4500    4503#   4590    4593#   4728    4731#   4796    4799#
                        5112    5115#   5443    5446#   5510    5513#   5614    5617#
$SAVRE= ****** U        1568
$SAVR6  007322          1685*   1693    1694*   1695*   1718#
$SCOPE  004134           177    1045#   2194
$SETUP= 000000           998    1046    1248    1345
$SVLAD  004316          1062    1080#
$SVPC = 000040           189#    194
$SWR  = 164000             1#      11     258     259     970     998    1009    1015    1017    1039    1040    1041    1042
                        1053    1065    1067    1068    1069    1070    1071    1083    1089    1715    2275    2301    2326
                        2352    2382    2425    2468    2527    2580    2604    2628    2664    2702    2751    2815    2891
                        2967    3040    3122    3204    3294    3369    3429    3463    3497    3539    3574    3633    3681
                        3729    3777    3825    3874    3926    3972    4024    4073    4157    4247    4331    4414    4499
                        4589    4727    4795    5111    5442    5509    5613
$SWREG  001340           279#    759
$SWRMK= 000000          1042
$TESTN  001322           270#   1081*
$TIMES  001310           258#    998*   1070*   1076    1079*   1089
$TKB    001246           238#   1052    1246    1263    1269    1808    1810    1852    2207
$TKS    001244           237#   1050    1246    1261    1267    1850    2205    6094
$TMP0   001276           253#    814*   1781    2162*   2163*   5721*   5733*   5767*   5778*   5813*   5824*   5865*   5871*
                        5883*   5897*   5899*   5902*   5903*   5920*   5931*   5966*   5977*   6016*   6021*
$TMP1   001300           254#    815*   1783    5043*   5044*   5045    5086*   5087*   5088    5378*   5379*   5380    5421*
                        5422*   5423    5588*   5589*   5590    5680*   5681*   5682    5765*   5781    5791*   5794    5808*
                        5812*   5839*   5846*   5884*   5889*   5918*   5934
$TMP2   001302           255#    817*    838*    865     874*   1785    1975    1978    2065*   2840*   2842*   2916*   2918*
                        2992*   2994*   3062*   3064*   3148*   3150*   3234*   3236*   3321*   3323*   3402*   3404*   4679*
                        4681*   4763*   4764*   4890*   4892*   4971*   4973*   5029*   5030*   5045*   5046    5072*   5073*
                        5088*   5089    5207*   5209*   5225*   5227*   5306*   5308*   5364*   5365*   5380*   5381    5407*
                        5408*   5423*   5424    5574*   5575*   5590*   5591    5666*   5667*   5682*   5683    6046*   6056*
                        6069*   6082*   6120
$TMP3   001304           256#    818*   1787    1987    1990    1995    1998    2052    2055    2060    2063
$TMP4   001306           257#    819*   1789    1970*   1982*   2148    5749*   5755*
$TN   = 000062             1#      11   2273    2275#   2299    2301#   2324    2326#   2350    2352#   2380    2382#   2423
                        2425#   2466    2468#   2525    2527#   2578    2580#   2602    2604#   2626    2628#   2662    2664#
                        2700    2702#   2743    2751#   2813    2815#   2889    2891#   2965    2967#   3038    3040#   3120
```

```
                    3122#  3202   3204#  3292   3294#  3367   3369#  3427   3429#  3461   3463#  3495   3497#
                    3537   3539#  3572   3574#  3631   3633#  3679   3681#  3727   3729#  3775   3777#  3823
                    3825#  3872   3874#  3924   3926#  3970   3972#  4022   4024#  4071   4073#  4155   4157#
                    4245   4247#  4329   4331#  4412   4414#  4497   4499#  4587   4589#  4725   4727#  4793
                    4795#  5109   5111#  5440   5442#  5507   5509#  5611   5613#
$TPB    001252       240#  1173*  1184   1591*  1855*  2210*
$TPFLG  001257       244#  1122   1184
$TPS    001250       239#  1171   1184   1589   1853   2208
$TRAP   006414       183   1539#
$TRAP2  006436      1550#  1561
$TRP  = 000021      1554#  1563#  1565   1566#  1567#  1568#  1569#  1570#  1571#  1572#  1573#  1574#  1575#
                    1576#  1577#  1578#  1579#  1580#
$TRPAD  006450      1544   1561#
$TSTM   002040       506#
$TSTNM  001202       217#   754*  1038   1080*  1081   1083   1090   1668   1722   1924   1931   1933   2275*
                    2301*  2326*  2352*  2382*  2425*  2468*  2527*  2580*  2604*  2628*  2664*  2702*  2751*
                    2815*  2891*  2967*  3040*  3122*  3204*  3294*  3369*  3429*  3463*  3497*  3539*  3574*
                    3633*  3681*  3729*  3777*  3825*  3874*  3926*  3972*  4024*  4073*  4157*  4247*  4331*
                    4414*  4499*  4589*  4727*  4795*  5111*  5442*  5509*  5613*
$TTYIN  005520      1289   1290   1302   1320   1334   1338#
$TYPBN= ****** U    1563
$TYPDS= ****** U    1563
$TYPE   004414      1122#  1215   1554   1562
$TYPEC  004626      1152   1159   1166   1171#  1172
$TYPEX  004674      1177   1179   1182#
$TYPOC= ****** U    1563
$UNIT   001330       273#
$UNITM  002044       508#
$USWR   001342       260#
$VECT1  001366       305#  2226   2227
$VECT2  001370       306#
$XTSTR  004174      1048   1056#
$Y    = 000000         1#   510#
$$GET4= 000000      1009#
$40CAT= ****** U    1053
.     = 037030       171#   173    176#   189    190#   192#   194#   196#   200#   204#   214#   262    348#
                     349#   350#   351#   360#   487#   493    494#   496#   498#   541#   543#   544#   545#
                     546#   548#   549#   550#   551#   553#   554#   555#   556#   558#   559#   560#   561#
                     563#   564#   565#   566#   568#   569#   570#   571#   573#   574#   575#   576#   578#
                     579#   580#   581#   583#   584#   585#   586#   588#   589#   590#   591#   593#   594#
                     595#   596#   598#   599#   600#   601#   603#   604#   605#   606#   608#   609#   610#
                     611#   613#   614#   615#   616#   618#   619#   620#   621#   782    792    899#   912
                    1017   1089   1090   1184   1236#  1246   1338#  1339   1345#  1399   1688   1717   1760#
                    1795#  1797#  1851   1854   1875   1968   2009   2092   2096   2142   2173   2206   2209
                    2363   3396   3977   3991   4006   4029   4052   4708   5000   5335
.ADVAN  006060      1439#  1579
.BEGIN  003464       927#
.CNVRT  006170      1472#  1578
.CONVR  006164      1471#  1577
.DATAC  007446      1574   1751#
.DELAY  007332      1572   1724#
.MSTCL  007362      1571   1735#
.RESOS  006132      1460#  1570
.ROMCL  007400      1573   1740#
.SAVOS  006072      1446#  1569
.SCOP1  004364      1094#  1568
```

K12


```
.START  002402            201    739#    755    1838
.TIMER  007512           1575   1762#
.$ASTA= ****** U         1188   1191
.$X   = 002034            493#   498
```

```
COMMEN    144#
ENDCOM    144#
ERROR      38#  2287  2312  2338  2368  2398  2411  2441  2454  2488  2509  2544  2562  2590  2614
              2640  2649  2687  2726  2736  2779  2789  2798  2846  2850  2860  2864  2874  2922  2926
              2936  2940  2950  2998  3002  3012  3016  3026  3068  3072  3087  3092  3099  3154  3158
              3169  3173  3187  3240  3244  3258  3262  3327  3331  3343  3352  3397  3408  3412  3449
              3483  3525  3559  3600  3612  3619  3659  3667  3707  3715  3755  3763  3803  3811  3850
              3902  3948  3957  3995  4010  4058  4102  4106  4132  4136  4191  4195  4222  4226  4276
              4280  4306  4310  4360  4364  4390  4394  4448  4462  4544  4548  4635  4639  4662  4666
              4685  4689  4700  4709  4754  4773  4846  4850  4873  4877  4896  4900  4927  4931  4954
              4958  4977  4981  4992  5001  5019  5039  5048  5062  5082  5091  5163  5167  5190  5194
              5213  5217  5231  5235  5262  5266  5289  5293  5312  5316  5327  5336  5354  5374  5383
              5397  5417  5426  5469  5478  5489  5562  5584  5593  5655  5676  5685  5757  5873  6081
ESCAPE    144#
GETPRI    144#
GETSWR    144#
HLT       157#
KMEND       1#  960
KMFRNT      1#
MULT      144#
NEWTST    144#  2273  2299  2324  2350  2380  2423  2466  2525  2578  2602  2626  2662  2700  2749
              2813  2889  2965  3038  3120  3202  3292  3367  3427  3461  3495  3537  3572  3631  3679
              3727  3775  3823  3872  3924  3970  4022  4071  4155  4245  4329  4412  4497  4587  4725
              4793  5109  5440  5507  5611
POP       144#  1230  1231  1388  1704  1705  2253
PUSH      144#  1191  1193  1214  1362  1678  1684  2216
REPORT      1#  144#
SCOPE      39#  2274  2300  2325  2351  2381  2424  2467  2526  2579  2603  2627  2663  2701  2750
              2814  2890  2966  3039  3121  3203  3293  3368  3428  3462  3496  3538  3573  3632  3680
              3728  3776  3824  3873  3925  3971  4023  4072  4156  4246  4330  4413  4498  4588  4726
              4794  5110  5441  5508  5612
SETPRI    144#
SETTRA   1554#  1565  1566  1567  1568  1569  1570  1571  1572  1573  1574  1575  1576  1577  1578
              1579
SETUP     144#
SKIP      144#
SLASH     144#
SPACE     144#
STARS     144#   187   210   262   265   490   492   499   968  1035  1107  1186  1245  1251  1280
             1348  1533  1674  1690  2273  2299  2324  2350  2380  2423  2466  2525  2578  2602  2626
             2662  2700  2749  2813  2889  2965  3038  3120  3202  3292  3367  3427  3461  3495  3537
             3572  3631  3679  3727  3775  3823  3872  3924  3970  4022  4071  4155  4245  4329  4412
             4497  4587  4725  4793  5109  5440  5507  5611
SWRSU     144#
TRMTRP   1554#
TYPBIN    144#
TYPDEC    144#
TYPNAM    144#
TYPNUM    144#
TYPOCS    144#
TYPOCT    144#
TYPTXT    144#
$ABORT      1#  3912
$AUTO       1#   824
$BCC        1#  4060  4144  4234  4318
$BINCR      1#  4402  4487
```

```
SBINWI      1#  3189
SBUFFE      1#  1791
SCDATA      1#  5493  5598
SCLOCK      1#  2616
SCOMP       1#  3520  3595  3607  3614  3654  3663  3702  3711  3750  3759  3798  3807  3944  3987
          4002  4768  5034  5077  5369  5412  5579  5671
SCRC        1#  4084  4114  4168  4204  4258  4288  4342  4372
SCRCSH      1#  4423  4508
SCYCLE      1#  1862
SEMPTY      1#  5428
SEOP        1#   960
SERTBL      1#   381
SFINI       1#  5687
SFLAG       1#  2840  2916  2992  3062  3148  3234  3321  3402  4679  4890  4971  5207  5225  5306
SFLOAT      1#  2474  2494  2533  2550
SGETPA      1#
SHALF       1#
SHEADE      1#    11
SINACT      1#  3417  3451  3527
SINIT       1#
SLINE1      1#  2456  2515
SLU1        1#  2263  2289  2314  2340
SLU12       1#  2370
SLU17       1#  2413
SMARHI      1#
SMARK       1#
SMATCH      1#  4757  5022  5065  5357  5400  5567  5659
SMOCK       1#
SMODEM      1#  3959
SMSG        1#  1779
SMULT       1#  3280
SPASEN      1#   974
SPATTE      1#  3813  3860
SPFAIL      1#  1669
SQQQI       1#  5941  5951
SQUEST      1#  1971  1983  1991  2048  2056
SRAMCL      1#  1723
SRCLK       1#  1726  1729  1766  1771  2280  2305  2331  2357  2389  2391  2402  2404  2432  2434
          2445  2447  2478  2480  2499  2501  2536  2538  2554  2556  2584  2608  2634  2643  2669
          2674  2676  2680  2707  2712  2714  2719  2729  2756  2761  2763  2768  2772  2782  2792
          2820  2827  2829  2834  2870  2896  2903  2905  2910  2946  2972  2979  2981  2986  3022
          3045  3051  3053  3127  3134  3136  3142  3183  3209  3219  3221  3225  3230  3274  3299
          3307  3309  3315  3374  3380  3382  3385  3387  3389  3434  3442  3468  3476  3502  3509
          3511  3514  3518  3544  3552  3579  3585  3587  3593  3603  3605  3638  3644  3646  3652
          3661  3686  3692  3694  3700  3709  3734  3740  3742  3748  3757  3782  3788  3790  3796
          3805  3830  3844  3879  3890  3896  3931  3938  3942  3950  3982  3985  3997  4000  4034
          4038  4045  4079  4163  4253  4337  4419  4432  4436  4440  4481  4504  4517  4521  4525
          4569  4594  4732  4748  4761  4766  4775  4800  5013  5027  5032  5041  5056  5070  5075
          5084  5116  5348  5362  5367  5376  5391  5405  5410  5419  5447  5465  5472  5482  5516
          5556  5572  5577  5586  5620  5649  5664  5669  5678  5696  5709  5724  5728  5731  5737
          5740  5751  5769  5773  5776  5782  5795  5798  5815  5819  5822  5828  5848  5850  5867
          5922  5926  5929  5935  5945  5955  5968  5972  5975  5988  5991  6002  6004  6019  6033
          6035
SRCRC       1#  4714
SREC        1#  3561  3621  3669  3717  3765
SROVAR      1#   331
```

```
$SCADD      1#   1046
$SCAD1      1#   1085
$SIMBC      1#   5877
$SINAC      1#   3485
$SOFTC      1#   1799
$STUFF      1#   3028
$SWPAC      1#   2568   2592
$TCHAR      1#   4601   4735   4807   5123
$TCRC       1#   4575   4778   5093
$TRANW      1#   4622   4833   4914   5150   5249
$TRAN1      1#   2651   2689   2738
$TSTN       1#   2271   2297   2322   2348   2378   2421   2464   2523   2576   2600   2624   2660   2698   2747
            2811   2887   2963   3036   3118   3200   3290   3365   3425   3459   3493   3535   3570   3629   3677
            3725   3773   3821   3870   3922   3968   4020   4069   4153   4243   4327   4410   4495   4585   4723
            4791   5107   5438   5505   5609
$UPADD      1#   1697
$VARIA      1#    203
$WINDO      1#   2800   2876   2952   3107
$XZ         1#   2263   2269   2289   2295   2314   2320   2340   2346   2370   2376   2413   2419   2456   2462
            2515   2521   2568   2574   2592   2598   2616   2622   2651   2658   2689   2696   2738   2745   2800
            2809   2876   2885   2952   2961   3028   3034   3107   3116   3189   3198   3280   3288   3356   3363
            3417   3423   3451   3457   3485   3491   3527   3533   3561   3568   3621   3627   3669   3675   3717
            3723   3765   3771   3813   3819   3860   3868   3912   3920   3959   3966   4012   4018   4060   4067
            4144   4151   4234   4241   4318   4325   4402   4408   4487   4493   4575   4583   4714   4721   4778
            4789   5093   5105   5428   5436   5493   5503   5598   5607
$ZEROS      1#   3356
$$CMRE    208#    247    248    249    250    251    252
$$CMTM    208#    253    254    255    256    257
$$ESCA    144#
$$NEWT    144#   2273   2299   2324   2350   2380   2423   2466   2525   2578   2602   2626   2662   2700   2749
          2813   2889   2965   3038   3120   3202   3292   3367   3427   3461   3495   3537   3572   3631   3679
          3727   3775   3823   3872   3924   3970   4022   4071   4155   4245   4329   4412   4497   4587   4725
          4793   5109   5440   5507   5611
$$SCOP      1#   1029
$$SET     1554#  1565   1566   1567   1568   1569   1570   1571   1572   1573   1574   1575   1576   1577   1578
          1579
$$SKIP    144#
.EQUAT      1#     34
.HEADE      1#
.SETUP      1#
.$ACT1      1#    185
.$APTB      1#    263#
.$APTH      1#    488
.$APTY      1#   1184
.$CATC      1#
.$CMTA      1#    208
.$EOP       1#    966
.$ERRO      1#
.$ERRT      1#
.$POWE      1#   1672
.$RDOC      1#   1346
.$READ      1#   1243
.$SCOP      1#   1033
.$TRAP      1#   1531
.$TYPE      1#   1105
.$TYPO      1#
```

. ABS. 037030 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DZKCF,DZKCF/SOL/CRF←DZKCF.MAC,DZKCF.P11/EQ:DZDME/EQ:LUTYPE
RUN-TIME: 31 28 2 SECONDS
RUN-TIME RATIO: 724/62=11.5
CORE USED: 53K (106 PAGES)